

<http://researchcommons.waikato.ac.nz/>

Research Commons at the University of Waikato

Copyright Statement:

The digital copy of this thesis is protected by the Copyright Act 1994 (New Zealand).

The thesis may be consulted by you, provided you comply with the provisions of the Act and the following conditions of use:

- Any use you make of these documents or images must be for research or private study purposes only, and you may not make them available to any other person.
- Authors control the copyright of their thesis. You will recognise the author's right to be identified as the author of the thesis, and due acknowledgement will be made to the author where appropriate.
- You will obtain the author's permission before publishing any material from the thesis.

PROBLEM SOLVING IN PHYSICS: AN INTERACTIVE COMPUTER- ASSISTED ENVIRONMENT

A thesis submitted in fulfilment of the requirements for the degree

of

Doctor of Philosophy

by

Amarjit Singh s/o Gurnam Singh

University of Waikato

1993

ABSTRACT

The thesis reports the development and trial of an interactive computer-assisted environment program. A rotational dynamics problem of considerable difficulty was chosen for program development involving a selection process using students and lecturers. Six year one undergraduates, 6 graduates, and a lecturer from the Physics department performed think-aloud problem solving. A high school Physics teacher and 7 year one undergraduates took part in pen and paper problem solving. Data collected focussed on: activities performed at commencement; question clarification and assistance sought; examples referred to for help; general mistakes made; conceptions portrayed; major difficulties encountered; and knowledge used. The problem solving has been categorised into 14 groups. These show the differences between expert and novice styles, highlight the strategies used, and depict the complexity of the problem-solving task through individual flow charts.

The problem-solving was performed in an idiosyncratic manner. This has informed the program development to represent knowledge so that it can be accessed at various stages of problem solving when needed. This suggests a constructivist environment with the provision of information through transmission as appropriate.

Solutions by the lecturer and the 4 high school teachers were used to determine the knowledge and explanations normally presented. A general solution using the fundamental physical quantities was developed to highlight the complexity of the solution and the vast amount of related knowledge required. This knowledge is usually assumed and normally not presented to students.

The findings informed the development of a problem-solving program. The program has been used by 18 and 14 students respectively to solve the problem for which it was developed and a related problem. Data collected highlighted the assistance students seek, the extent of student control allowed, program features used and limitations encountered. Data was also examined to ascertain transfer of learning, and the effectiveness of the program in compiling student conceptions, knowledge and problem-solving styles.

Analysis indicates students will use a program if it is helpful in serving their needs and is easy to use. Students were not very effective in transforming knowledge to a new situation. Data collected by the program showed its effectiveness in documenting information and features used, but it was less useful in compiling alternative conceptions.

ACKNOWLEDGMENTS

I am grateful to my supervisors:

Associate Professor Malcolm Carr for his continual guidance, encouragement and assistance throughout the research; and

Professor Bruce Liley for his help, support and guidance.

I would also like to thank:

Alistair Steyn-Ross for his help and guidance in the initial research phase and during program development.

The year 1 undergraduates (1991 and 1993), graduate students and a visiting lecturer from the Physics department; the high school Physics teachers, and the residents of Student Village who kindly participated and helped in the data collection.

Craig Lynch-Blosse and Lawrence D'Oliveiro from the Computer Services Department for their help on programming matters.

Friends who helped to proof read the thesis.

The University of Waikato for the grant of the Postgraduate Scholarship.

All others who in some way or other helped towards the production of this thesis.

Special thanks to:

John and Sharon Pearson for their friendship, total support and invaluable help in making available facilities to enhance the completion of this thesis.

Garth and Teresa Fernandez Ritchie for their friendship, encouragement and help.

The Science and Mathematics Education Research Centre for the support and provision of facilities.

My family for their support, continual encouragement and sponsorship. This provided comforting solace and peace of mind during difficult times.

I wish to dedicate this thesis to:

my mother Satwant Kaur, for instilling the awareness and value of lifelong education and achievement;

and

my eldest brother Rajdeb Singh for his support and sponsorship of my first degree which has led to this eventual accomplishment.

ਮੇਰਾ ਇਹ ਲੇਖ ਮੇਰੀ ਮਾਤਾ ਸਤਵੰਤ ਕੌਰ
ਦੀ ਭੇਟਾ ਵਿਚ। ਜਿਸਨੇ ਮੈਨੂੰ ਵੱਧ ਤੋਂ ਵੱਧ
ਸਿਖਿਆ ਵੱਲ ਪ੍ਰੇਰਿਆ। ਉਨ੍ਹਾਂ ਦੀ ਪ੍ਰੇਰਣਾ
ਦੇ ਜਹਾਜ਼ੇ ਹੀ ਅੱਜ ਇਹ ਗਿਆਨ ਸਿਰੇ
ਚੜ੍ਹਿਆ ਹੈ।

CONTENTS

CHAPTER 1: INTRODUCTION AND OVERVIEW

1.1 INTRODUCTION	1
1.2 OVERVIEW	2
1.3 RESEARCH QUESTIONS	5

CHAPTER 2: THEORETICAL BACKGROUND ON LEARNING AND PROBLEM SOLVING

2.1 INTRODUCTION	6
2.2 LEARNING THEORIES AND MODELS	8
2.3 LEARNING ENVIRONMENTS	12
2.3:1 Transmission	13
2.3:2 Constructivism	15
2.4 PROBLEM SOLVING	18
2.4:1 General problem-solving steps	19
2.4:2 Problem-solving strategies	20
2.4:3 Problem-solving knowledge	23
2.5 TEACHER'S ROLE IN THE LEARNING ENVIRONMENT	24
2.6 CONCLUSION	25

CHAPTER 3: RESEARCH INTO COMPUTER-ASSISTED LEARNING

3.1 INTRODUCTION	26
3.2 HISTORICAL PERSPECTIVE	27
3.3 ARCHITECTURE OF ICAI SYSTEMS	30
3.3:1 Expertise module	30
3.3:2 Student model	31
3.3:3 Tutoring module	31
3.3:4 Interface	33
3.4 KNOWLEDGE REPRESENTATION AND ACCESS	34
3.4:1 Production systems	34
3.4:2 Frames, Scripts and Schemata	35
3.4:3 Accessing knowledge	38
3.5 LEARNING THEORIES EMPLOYED IN ICAI SYSTEMS	39
3.6 PROGRAM FEATURES FOR A CONSTRUCTIVIST ICAI SYSTEM	47
3.7 CONCLUSION	50

CHAPTER 4: RESEARCH DESIGN AND METHODOLOGY

4.1 INTRODUCTION	51
4.2 REVIEW OF DATA COLLECTION METHODS	52
4.3 RESEARCH SCHEDULE	56
4.4 PHASE 1: INITIAL DATA COLLECTION AND ANALYSIS	57
4.4:1 Target group and topic	57
4.4:2 Choice of questions	57
4.4:3 Think-aloud sessions	60
4.5 PHASE 2: PROBLEM SELECTION AND ASSOCIATED DATA COLLECTION	62
4.5:1 Problem selection	62
4.5:2 Pen and paper data collection and written feedback	64
4.5:2a Novices	64
4.5:2b Expert	65
4.5:2c Teachers	65
4.5:3 The general and alternative problem solution	66
4.6 PHASE 3: THE PROGRAM DEVELOPMENT	66
4.6:1 The Hyper-X shell	66
4.6:2 The programming	67
4.6:3 Program	67
4.7 PHASE 4: PROGRAM USE AND EVALUATION	68
4.7:1 Stage 1	69
4.7:2 Stage 2	71
4.8 SUMMARY	72

CHAPTER 5: PROBLEM-SOLVING PROTOCOLS ANALYSIS

5.1 INTRODUCTION	73
5.2 DATA ANALYSIS LEADING TO THE PROBLEM SELECTION	74
5.3 THE PROBLEM SOLVING PROCESS OF THE PARTICIPANTS	76
5.3:1 Activities performed at commencement of problem solving	76
5.3:2 Question clarifications and assistance sought	80
5.3:3 Examples referred to for help	84
5.3:4 General mistakes made while problem solving	87
5.3:5 Conceptions portrayed	89
5.3:6 Major difficulties encountered while problem solving	94
5.3:7 Knowledge used to solve the problem	96
5.4 ACTIVITIES PERFORMED WHILE PROBLEM SOLVING	99
5.5 GENERAL DISCUSSION OF INFORMATION ON ACTIVITIES PERFORMED	109
5.6 DIFFERENCES BETWEEN EXPERT AND NOVICE USE OF THE ACTIVITIES	111
5.7 PROBLEM-SOLVING FLOW CHARTS	112

5.8 GENERAL AND SPECIFIC PROBLEM-SOLVING STRATEGIES USED	115
5.8:1 General strategies	115
5.8:2 Specific problem-solving strategies	119
5.9 PROBLEM-SOLVING PERCEPTIONS AND EXPECTATIONS	124
5.10 THE LEARNING ENVIRONMENT	125
5.11 CONCLUSION	126
 CHAPTER 6: THE PROBLEM SOLUTION	
6.1 INTRODUCTION	128
6.2 THE EXPERT'S SOLUTION	128
6.3 THE TEACHERS SOLUTIONS	130
6.4 THE COMPLETE SOLUTION	139
6.5 ALTERNATIVE SOLUTION FOR ROLLING	148
6.6 CONCLUSION	150
 CHAPTER 7: THE PROGRAM	
7.1 INTRODUCTION	152
7.2 MAIN PROGRAM COMPONENTS	153
7.2:1 The start panel and general access	154
7.2:2 Force Equations and Check Force Equations	156
7.2:3 Energy Expressions and Check Energy Expressions	161
7.2:4 Envision the motion	166
7.2:5 Perform Solution and Perform Solution Check	166
7.2:6 Perform Simulation	170
7.2:7 Answer Check	172
7.2:8 Other general facilities	172
7.3 THE INTERNAL PROGRAM STRUCTURE	173
7.3:1 Expertise module	175
7.3:2 Student module	175
7.3:3 Tutoring module	176
7.3:4 Interface	178
7.3:5 Knowledge representation and access	178
7.4 INCORPORATING THE PROBLEM-SOLVING STYLES	181
7.4:1 The activities performed	181
7.4:2 The problem-solving strategies	184
7.5 LEARNING ENVIRONMENT PROVIDED BY THE PROGRAM	184
7.6 CONCLUSION	186

CHAPTER 8: ANALYSIS AND EVALUATION OF DATA ON PROGRAM USE

8.1 INTRODUCTION	188
8.2 GENERAL INFORMATION ON PROGRAM USE	189
8.3 ANALYSIS OF THE PROGRAM EVALUATION QUESTIONNAIRE	189
8.3:1 General student opinion of the program features	190
8.3:2 Difficulties encountered while using the program	192
8.3:3 Features which the students liked	193
8.3:4 Purposes for which the program might be consulted	195
8.3:5 Program limitations encountered	196
8.4 ANALYSIS OF THE PROBLEM EVALUATION QUESTIONNAIRE	198
8.4:1 Difficulty in solving the problem	199
8.4:2 Help accorded by the program	202
8.4:3 Similarities between the two problems	204
8.5 INTERACTION RECORDS COMPILED	207
8.5:1 Features used to help solve the problem	207
8.5:2 Information used and recalled	209
8.5:3 Difficulties encountered and conceptions portrayed	211
8.5:4 Differences in transcripts compiled	214
8.6 EXTENT TO WHICH RESEARCH AIMS HAVE BEEN MET	217
8.7 CONCLUSION	221

CHAPTER 9: CONCLUSION

9.1 SUMMARY OF FINDINGS	222
9.2 IMPLICATIONS FOR LEARNING	223
9.3 RECOMMENDATIONS FOR FUTURE WORK	225

APPENDIX A1: INSTRUCTIONS TO PARTICIPANTS FOR THE THINK- ALoud SESSIONS

227

APPENDIX A2: INSTRUCTIONS TO PARTICIPANTS FOR THE PEN AND PAPER SESSION

228

APPENDIX A3: INSTRUCTIONS TO TEACHERS

229

APPENDIX A4: INSTRUCTIONS TO PARTICIPANTS PRIOR TO PILOT USE OF PROGRAM

230

APPENDIX A5: INSTRUCTIONS TO PARTICIPANTS PRIOR TO USING THE PROGRAM FOR SOLVING THE PROBLEM

231

APPENDIX A6: PROGRAM EVALUATION QUESTIONNAIRE

232

APPENDIX A7: INSTRUCTIONS TO PARTICIPANTS SOLVING THE RELATED PROBLEM

237

APPENDIX A8: PROBLEM EVALUATION QUESTIONNAIRE	238
APPENDIX B1: THINK-ALOUD TRANSCRIPTS	241
APPENDIX B2: FLOW CHARTS	269
APPENDIX C: RELATED INFORMATION FOR THE COMPLETE SOLUTION OF THE PROBLEM	273
APPENDIX D1: THE USER MANUAL	313
APPENDIX D2: PROGRAM CODE	337
REFERENCES	408

LIST OF FIGURES AND CHARTS

Figure 3.1: An example of a production rule with a single assertion
and conclusion 34

Figure 3.2: A production rule with five assertions and a single conclusion 34

Figure 3.3: A production rule with a single assertion leading to three
conclusions 35

Figure 3.4: Frame for a rolling ball at the bottom of the loop 36

Figure 3.5: The completed frame for a rolling ball at the bottom of the loop. 37

Figure 4.1: The methodology used for data collection. 52

Figure 4.2: Warm-up questions used. 61

Figure 4.3: The modified version of question 6. 63

Figure 4.4: The similar question administered. 71

Figure 5.1: Examples of diagrams drawn by N3 to depict torque or angular
momentum. (a) For both propellers on vertical axes. (b) For one
propeller on a horizontal axis. 75

Figure 5.2: Example referred by the participants. 85

Figure 5.3: Ball oscillating between two points. 91

Figure 5.4: The forces and acceleration for the top of the loop. 92

Figure 5.5: Energy treated as forces and given direction. 93

Figure 5.6: The sequential problem-solving flow chart of expert E2. 113

Figure 5.7: The frequency flow chart of expert E1. 114

Figure 5.8: Problem solution concept map of E2. 123

Figure 6.1: Force diagram for the top of the loop. 129

Figure 6.2: Diagram drawn to show the forces acting on the ball at the top of
the loop. 132

Figure 6.3: Forces at a general point and at the top of the loop. 134

Figure 6.4: Solution steps on a transparency. 135

Figure 6.5: Diagram showing the 2 components of F_{grav} . 136

Figure 6.6: The forces acting on the ball at a general point in the loop. 140

Figure 6.7: The radial (normal) and tangential directions. 141

Figure 6.8: The resolved forces and the tangential and radial accelerations. 141

Figure 6.9: The direction of the forces in the top right quadrant of the loop. 143

Figure 6.10: The forces at the top of the loop. 143

Figure 6.11: The forces acting on the rolling ball (f is the frictional force). 146

Figure 6.12: Applying the parallel axis theorem. 149

Figure 7.1: The start panel. 153

Figure 7.2: The question window providing the problem wording and access

to question clarification.	154
Figure 7.3: The question clarifications available and the information provided when rolling vs sliding is selected.	155
Figure 7.4: The contents of the Window menu indicating the open windows during interaction.	155
Figure 7.5: The Start menu with the start panel closed, open and in the background, and open and currently active.	156
Figure 7.6: The two interaction windows for writing the force equations.	157
Figure 7.7: The final force expression obtained for a general point in the loop.	158
Figure 7.8: The force diagram for the lower left quadrant of the loop.	159
Figure 7.9: Information provided through a dialogue box during interaction.	160
Figure 7.10: The Force Diagrams window showing the resolved forces and accelerations for a rolling ball in the top right quadrant of the loop.	160
Figure 7.11: Writing energy expressions interaction windows.	162
Figure 7.12: The final energy expression for the top of the loop and the equation for the start and the top of the loop.	163
Figure 7.13: Two examples of information provided through a dialogue box during interaction for writing energy expressions.	164
Figure 7.14: A completed check for the start and the top of the loop for a rolling ball.	165
Figure 7.15: The envisioning window showing the description of the motion for <i>Insufficient starting height</i> .	165
Figure 7.16: The two windows which can be used to perform the solution to the problem.	167
Figure 7.17: A dialogue box giving the final expression for the minimum starting height for rolling motion.	168
Figure 7.18: Dialogue boxes giving information on the positions at which to write the force equation and energy expressions.	169
Figure 7.19: The simulation window showing a simulation for rolling motion of the ball.	170
Figure 7.20: Pop-up menus for the simulation window.	171
Figure 7.21: The <i>Answer Check</i> window.	172
Figure 7.22: The untitled worksheet.	173
Figure 7.23: Rules used and knowledge supplied by the expertise module.	174
Figure 7.24: Possible interactions for writing the force equation for the top of the loop.	177
Figure 7.25: Information for commencing the <i>Check Force Equations</i> interaction.	179
Figure 7.26: Dialogue box presented when performing a force equation check.	179

Figure 7.27: A rule and associated dialogue box to prompt the student.	181
Figure 8.1: Marble loses contact with the sphere when it has travelled a quarter of a circle along the sphere.	201
Figure 8.2: The forces in the radial direction for the ball moving in the track and the marble on the sphere.	201
Figure 8.3: Measurement of the angle from the radius vector passing through the top and the bottom.	212
Figure 8.4: Conceptions of points at which the marble would lose contact with the sphere.	212
Figure 8.5: The horizontal motion of the ball at the top of the loop.	216
Chart 8.1: Student choices of purposes for which they would consult the program.	196
Chart 8.2: Features of the program used by the students to solve the related problem.	208

LIST OF TABLES

Table 2.1: Groupings of related learning theories.	7
Table 3.1: Differences between CBI and ICAI systems.	28
Table 3.2: Explanations highlighting the usage of the learning theories within the computer-assisted instructional systems.	45
Table 3.3: Learning theories and operating environments on which the ICAI systems are based.	46
Table 4.1: Participants in the initial data collection.	60
Table 4.2: The information sought by the questionnaire in stage 1 related to the research questions.	70
Table 5.1: Success of participants at solving the 6 problems.	74
Table 5.2: The first activities of the participants when problem solving.	77
Table 5.3: Summary of information used by participants of the think-aloud session.	97
Table 5.4: Summary of information used by participants of the pen and paper session.	98
Table 5.5: Frequency of activities performed by the think-aloud participants.	100
Table 5.6: Number of participants using the different strategies.	115
Table 7.1: Frame for the rolling motion of the ball at the bottom of the loop.	180
Table 8.1: The number of students selecting a response for each statement.	190
Table 8.2: Equating the energies at the top and a general point and adapting it.	209
Table 8.3: The information supplied by the students for the solution of the related problem.	210
Table 8.4: An excerpt from a program compiled transcript.	213
Table 8.5: The information collection aspects addressed by each method.	214

CHAPTER 1

INTRODUCTION AND OVERVIEW

1.1 INTRODUCTION

The research reported in this thesis aimed at providing students with a computer-assisted instructional system which takes their prior knowledge into account, and allows them to be the initiators and controllers of their learning in the interactive process. In practically all the instructional systems developed, the prior knowledge of students is ignored, and the student is not in control of the interaction with the system. This prompted the development of the computer-assisted instructional system to assist and complement student learning. The system here is associated with problem solving. The interactive program provides opportunities not only to learn to solve the problem but to obtain hints, to discuss/check a proposed solution, to provide help in performing diagrammatic or graphical representation, to provide graphical envisioning through simulation, and to provide qualitative envisioning to develop insight into the problem. Students might require further information, recommended readings or further similar or graded problems on the topic. Unlike a human tutor, the system can provide students with as much interaction as they wish. In the process students at a loss on how to pursue the interaction are provided with declarative knowledge on definitions, principles, equation formats or units. This transmission is usually followed by an interaction.

Textbook problem solving is a major component of Physics learning at the first year undergraduate level. Large classes and limited resources often mean that limited help is available to assist students in difficulty. Furthermore, a computer-assisted system provides an opportunity to seek clarification and help when it is needed, without being concerned about the opinion of peers and teachers. At this first year undergraduate level, lecturers pass on declarative knowledge to a few hundred students at a time. When time permits worked examples are solved by the lecturer to help students apply their knowledge. Students are then assigned textbook problems for practice and to gain further insight and understanding. Tutorial sessions (usually conducted once a week) provide an opportunity to clarify lecture material as well as help to solve assigned problems. In many cases students burdened by other subject commitments do not attempt to solve the problems prior to the tutorial session and are reluctant to ask questions in the presence of their peers.

In this study of a computer-assisted instructional system which is interactive the topic of rotational dynamics was chosen. This is often the first topic in a Physics course exploring the motion of bodies where the geometry of the bodies is taken into account. Prior to this, objects are treated as particles. The concept of moment of inertia or rotational inertia is now introduced replacing mass as the analogous rotational equivalent. Students have difficulties with this transition from particle to extended body considerations. The problem used was chosen for the following reasons:

- It cannot be solved by applying an algorithm. A successful solution requires a number of theorems, principles, constraints and assumptions.
- It involves a combination of translational and rotational motions.
- It involves sliding motion, whereby a rigid body may be considered a particle.
- It involves rolling motion, whereby the dimensions of the rigid body have to be considered.
- For rolling motion, there are two solution paths which may be used to successfully solve the problem.

1.2 OVERVIEW

The thesis which follows discusses the literature review giving the theoretical background on learning and problem solving, and the research into computer-assisted learning. The development of the rotational dynamics problem used is detailed through data collection using think aloud protocols of experts, teachers and novices. The knowledge required to solve the problem has been determined by analysing these protocols. The expert solution, a teaching solution, a general solution encompassing all the knowledge required to solve the problem and an alternative solution path to the problem are presented. The knowledge required and the various solutions have been used to develop a pilot interactive computer-assisted program using the Think C Class Library, an object oriented version of the C programming language. Data collected through students using the program to assist them in solving the problem has been used to further develop the program and to evaluate it. A related problem has been used to determine the transfer of learning enabled by the use of the program. The evaluation and interviews have been used to determine the strengths and limitations of the program, and to make suggestions for further work on the program.

The thesis is organised into nine chapters and an appendix. It commences with an abstract providing a summary of the research and findings included in this thesis. Chapter 1 provides a rationale for the research performed. This follows an overview of the thesis and concludes with the research questions explored by the thesis.

Chapter 2 gives the literature review on learning and problem solving providing the theoretical background for a constructivist learning environment coupled with transmission. Learning is broadly defined and 12 learning theories or models are briefly outlined. A discussion of the transmission and constructivist learning environments and the interplay of the various learning theories and models within each of the environments is highlighted. The implementation and role of interactive discussion and problem solving within each environment is pinpointed. Ten general problem-solving strategies are detailed. The chapter concludes with a discussion of the teacher's role in helping students in a constructivist learning environment coupled with transmission.

Chapter 3 gives the literature review on computer-assisted learning, from traditional computer-based systems (CBI) to artificial intelligence techniques and expert systems resulting in intelligent computer-assisted systems (ICAI). The improvements in computer-assisted learning are highlighted by analysing the differences between CBI and ICAI systems. Insight is provided into the workings of ICAI systems through a discussion of the main components, the knowledge representation models used and the techniques used to access knowledge and provide inferences. Thirty systems are summarised and reviewed to determine the learning theories and environments inherently or explicitly used. The program features necessary to realise a constructivist based problem-solving environment coupled with transmission are mentioned. A discussion of the extent to which these features may be realised concludes the chapter.

Chapter 4 details the research design and methodology used for data collection. The D. Phil. research schedule is given and the following are detailed:

- (a) The methodology and problems used for the initial data collection.
- (b) The instructions given to students during the different data collection phases.
- (c) The methodology used to collect data during and subsequent to program usage.
- (d) The methodology used to evaluate the program.

The chapter concludes with a discussion of a related problem and the questionnaire used to determine the transfer of learning to the solution of that problem.

In Chapter 5 the think-aloud, and pen and paper data collected in phase one and two of the research is analysed. The data is used to justify the selection of the question

used for further research and eventual program development. The initial activities performed upon commencement of problem solving, the question clarifications sought, examples and information referred to for assistance, the general mistakes made, the conceptions portrayed, the major difficulties encountered and the knowledge needed and used by participants for solving the problem are highlighted. Excerpts from transcripts reinforce the analysis. The activities performed by the participants during problem-solving are identified and categorised into fourteen groups which are used to describe the problem-solving processes used by participants. The general and specific problem-solving strategies used are inferred, and the participants expectations and perceptions of the problem-solving function are mentioned. The learning theories and environments which would suit the participants styles are inferred. The use of a constructivist learning environment with the provision of information through transmission when required, is justified.

Chapter 6 provides the solutions to the problem obtained using pen and paper in phase two of the research. Solutions by an expert and two teachers provide the information generally mentioned when students are being helped to solve the problem. A general solution to the problem using the fundamental concepts of time, mass and displacement highlights the vast amount of relevant knowledge which is assumed to be known by the students. It also shows the tediousness if every relationship and concept used in solving a problem is justified. An alternative solution for the rolling case which was developed when compiling the general solution is also provided. These solutions outline the wide knowledge base from which selected information is used to form the knowledge base of the program.

Chapter 7 presents the structure of the program. The major changes to the program after its pilot use are highlighted. The external features of the program and the purposes for which it can be consulted are detailed. The internal workings of the program are described using the expertise module, the student module, the interface and the representation and access of knowledge. The use of frames and rules to provide interaction is detailed. The structure of the program is related to the activities performed during problem solving. The incorporation of the problem-solving strategies within the program is made explicit. The chapter concludes with a description of the extent to which it provides a constructivist learning environment.

Chapter 8 presents the analysis and evaluation of the research aims. The data collected using questionnaires and interviews, and transcripts of student interactions compiled through observations and by the program are analysed. These are used to evaluate the program and discuss the extent to which the research aims of the study have been realised.

Chapter 9 concludes the research report. A summary of the findings of the research are presented. Implications of the findings are discussed and recommendations made for future work.

This chapter is concluded by detailing the research aims of the study.

1.3 RESEARCH QUESTIONS

In describing the development of a computer-assisted instructional system exploring the topic of Rotational Dynamics, this study aims to answer the following research questions:

- What assistance do students seek in interacting with a computer-assisted system?
- To what extent does the system allow students to control the interactions ?
- Do students find it encouraging and worthwhile to consult such a computer-assisted system ?
- Which features of the system are deemed good by the students ?
- What system limitations do students encounter ?
- Can students transfer their learning to the solution of related problems ?
- How effective is a computer-assisted system in interactively collecting data on individual student knowledge, conceptions and problem-solving styles ?

In the next chapter the literature review on learning and problem solving is presented.

CHAPTER 2

THEORETICAL BACKGROUND ON LEARNING AND PROBLEM SOLVING

2.1 INTRODUCTION

Learning may be broadly termed as the gaining of knowledge, comprehension or mastery through experience. It always involves the interaction between the learner's present understanding of the world and the knowledge input. Knowledge may be termed the awareness or understanding gained through experience. It is the sum range of what has been perceived, discovered, comprehended or mastered by the person. Experience may be termed the accumulation of knowledge through the active participation in events or activities formally as well as informally outside the classroom. Carbonell (1986) defines learning to take place when a system (biological or mechanical) can modify its behaviour after a set of experiences such that it can perform a task more accurately or more efficiently than before or perform a new task beyond its previous capabilities. A broad view of cognition and learning put forward by George (1970) encompasses at least the properties of perception, recognition, recall, conceptualising, inference making, problem solving, thinking and the use of language. The complexity of learning is reiterated by factor analysts who claim to have discovered more than fifty different intellectual factors (Harmon, 1960). Learning then denotes a wide range of processes and outcomes. Vitale (1983) postulates that the brain is compartmentalised into a left and right hemisphere of consciousness to handle the various aspects of learning. The left hemisphere handles linear, symbolic, sequential, logic, verbal, reality based, temporal and abstract aspects of learning while the right hemisphere handles holistic, concrete, random, intuitive, non-verbal, fantasy oriented, non-temporal and analogic aspects.

Learning being an internal process cannot be directly measured and evaluated. Inferences about learning are made from student behaviour and actions. This is usually done through information obtained from:

- performance in some assigned task;
- application to a novel situation; and
- explication of learning in various contexts.

Group	Related Learning Theories	Learning Procedures Implied
A	Skinner's Operant Conditioning. Thorndike's S-R Connectionism.	Stimulus-Response reinforcement through feedback.
B	Piaget's Cognitive Development. Kelly's Personal Constructs. Schemata Acquisition. Hierarchical Organisation.	The gradual formation in stages, of inter-related hierarchical network of ideas through discrimination and refinement.
C	Phenomenological Approach. Maslow's Development Levels.	Self actualisation seen as the motivating force for personal enrichment.
D	Ausubel's Meaningful Learning.	Meaningful incorporation of information through reception and discovery.
E	Generative Learning model.	Student-initiated learning bringing about the development of meaning and reorganisation of conceptions through experience.
F	Gestalt Learning	Response to whole patterns or situations, interaction with total experience.
G	Information Processing.	Modelling of how people think, handle, store, and retrieve information over time.

Table 2.1: Groupings of related learning theories.

Since Thorndike proposed his theory of connectionism in the late 1890's (Thorndike, 1949), a number of theories of learning and of processes for improving learning have been proposed. The early learning theories may be classified as Associationist-Behaviourist and Gestaltist. Associationist-behaviourists believe that learning starts with irreducible elements and the process of learning is one of combining these elements through stimulus-response behavioural actions. The Gestaltists believe that learning is closely related to perception, and insight is obtained by the learner observing a total conceptual pattern. Present day learning theories however address the mental constructions of the mind and how these propagate learning. Learning theories have affected views of the teaching and learning process. In earlier years the application of the associationist theories brought about rote learning encouraging the ability to recall and repeat facts through drill and reinforcement. The Gestaltists encouraged

reflective learning by obtaining insight about relationships between concepts within whole patterns. The present day learning theories encourage using the conceptual framework of the learner to facilitate learning. Table 1 shows 12 learning theories categorised into 7 groups by their implied learning procedures.

The next section outlines these 12 learning theories in more detail and discusses how they have informed the teaching and learning process, either consciously or subconsciously. Then follows a description of two commonly employed learning environments, transmission and constructivism. The extent to which each or a combination of both is employed varies with the teacher, the subject matter being taught and the task under consideration. The role of rote learning, discovery, interactive discussions and problem solving are highlighted within each of these environments. This follows a detailed discussion on general problem-solving steps, strategies and knowledge as both environments use problem solving to provide students with the opportunity and assistance to apply the correct principles. In any learning environment the role of the teacher is an important factor. This is the focus of attention in the final section of this chapter.

2.2 LEARNING THEORIES AND MODELS

All the learning theories and models discussed basically propose cognitive change through knowledge compilation from smaller to larger functional units which may be easily modified and adapted. These are theories developed for separate purposes and may be broadly classified by their origins. Laboratory studies using controlled conditions with animals as subjects led to the theories of Skinner and Thorndike. The study of the maturation and motivational processes of human beings from infant to adulthood led to the Phenomenological and Gestaltist theories, and those of Maslow and Piaget. The theories of Ausubel, Kelly and the generative model evolved from teaching, learning and curriculum development research in the classroom. Piaget's work has also been applied to the construction and teaching of the school curriculum. The information processing system, schema acquirement and hierarchical organisation have originated from research on trying to understand and model the cognitive processing in human beings. These 12 learning theories are each briefly discussed below.

Skinner's (1969) operant conditioning: is the learning process whereby a response is made more probable or frequent through reinforcement or feedback. Reinforcements may be positive or negative and are used to improve behaviour. Complex responses are seen as the result of simple responses being built into more

complicated behaviour patterns. Transfer of learning is maximised by building a very large repertoire of responses.

Thorndike's (1949) stimulus-response theory of connectionism:

considers learning as a process of linking physical (stimulus or response) and mental (something sensed or perceived) units or events in various combinations. A response is strengthened if it is followed by pleasure and weakened if followed by displeasure.

Reward is generally used as a motivating factor. Transfer of learning takes place when stimulus-response (S-R) connections learned in one situation may be used for another situation.

The Phenomenological approach: (Coombs and Snygg, 1959) sees a person as an organism forever seeking greater personal adequacy. The search for adequacy or self actualisation is the driving force motivating all behaviour. Learning is a function of need which is perceived as maintaining and enhancing the phenomenal self. Factors that become more self-related are more readily learned and retained than those that are less self-related. A person learns what is perceived as a need to know in order to maintain the self. Simple perceptions are made more complex through a process of differentiation.

Gestalt learning theory: (Bigge, 1982) considers learning phenomena to be closely related to perception of a whole, which is seen as conferring configuration of meaning on an object and trying to organise or integrate what is seen. Gestaltists believe that to understand a thing, a study of its totality is required. A study of its constituent parts is not considered sufficient. The major premise of Gestaltists is that people respond to whole patterns or situations. Purposeful learning is a dynamic process of organising perceptions to reduce ambiguity, which acts as a motivating factor. Development of insight is seen as the manner in which learning takes place.

Maslow's (1987) skill development: theory of personality states that a person possesses both lower (animalistic) and higher (human) needs. Lower needs include those for physiological survival, safety, belongingness and esteem. Higher needs include those for self actualisation and cognitive understanding. All needs can be conscious or unconscious. The goal is to try and move through the levels from unconscious incompetence, to conscious incompetence, through to conscious competence, to attain and achieve unconscious competence. For some skills the development through the levels might be quite rapid but for others it might be much slower. One might not be able to reach the final level for every skill aspect but the goal might be to achieve that.

Piaget's cognitive development theory: (1956, 1960 cited in Pinard and Laurendeau, 1969) divides cognitive development into the four stages of pre-conceptual thought (2 to 4 years), intuitive thought (4 to 7 years), concrete operations (7 to 11 years) and formal operations (11 to 15 years). He lists five criteria attributed to the conditioned mental development of the mind in his theory of stages. These in order are: hierarchical intransitivity of stages, integration through restructuring and coordination of states, consolidation, structuring by creating inter-connections to obtain total structures, and equilibration. The final stage of equilibration involves the processes of assimilation (or association) to earlier schemata and accommodation to new requirements. Equilibration is realised in successive degrees. It does not refer to a particular stage but plays a role in the entire process.

Ausubel's (1968) meaningful learning: can take place through reception or discovery. In meaningful reception learning the potentially meaningful task is presented to the learner in final form. It is then comprehended or made meaningful in the process of internalization. In discovery learning the principal content of what is to be learned must be discovered by the learner before it can be meaningfully incorporated and internalised. Internalisation occurs through progressive differentiation of subsuming concepts and by integrative reconciliation. For the most part, large bodies of subject matter are acquired through reception learning whereas the everyday problems of living are solved through discovery learning.

Kelly's (1955) personal constructs: asserts that individuals represent knowledge using structures called constructs. A construct brings out the similarities of a set of elements, and differences of the set (objects, events, situations or individuals) from other elements. A person's past experience with similar and dissimilar elements in comparison to the current elements under consideration helps him/her to perceive similarities and differences in the formation of constructs. A person uses constructs to summarise, to give meaning to, and to anticipate events. Garg-Janardan and Salvendy (1988) mention that constructs are inter-related and may be represented as hierarchies or networks within an individual.

The Generative learning model: (Osborne and Wittrock, 1985) states that people tend to generate perceptions and meanings that are consistent with their prior learning. Learners bring prior knowledge with them to the learning situation and actively construct meaning while interacting with the environment. The sense made of any event is seen to be dependent not only on the situation itself but also on the individual's purposes and active construction of meaning. The learner is seen as an active participant in the construction of his/her own knowledge. The models constructed by individuals are continually tested against experience and if necessary modified.

The Information processing model: (Gagne, 1977) describes mental events in terms of transformations of information from input (stimulus) to output (response). Information received is held in more or less complete form for a very brief time during processing. Only a small fraction of the processed information is kept for continued representation in short term memory (STM) while the rest is lost from the system. Information in STM can persist for a limited period, generally thought to be about 10 seconds, unless it is rehearsed (silent mental repetition of the information). It is integrated in various ways with known information and enters long term memory. Retrieval of information happens through response generation using cues provided externally or by the learner.

Schemata: (Rumelhart & Norman, 1981) are large well-coordinated and organised units of knowledge. Associated with each schema are cues indicating how to use it, what is to be expected from its use and the action to pursue if it fails (Millward, 1980). Schemata are organised by other schemata into structures that represent organisation of knowledge. Learning involves not only the development of the schemata but also building of the connections to create an organised network of structures. This is done by coding new information in terms of existing schemata or through schema refinement. Once evoked, a schema begins processing data and looks for appropriate information to complete its meaning. If appropriate data is not found then the schema is replaced by some other schema. An active schema may evoke a number of inter-connected schemata. Learning is considered the process of organising the evoked schema into some higher level schema.

Hierarchical organisation: (Reif, 1981) advocates the storage of concepts and relationships by the decomposition of larger units into manageably smaller sized blocks. These are hierarchically organised into structured levels. It has been shown that concept maps organised by experts not only include more concepts but have better hierarchical structure. Hierarchical organisation and storage of concepts and relationships is predominant amongst learners who show evidence of meaningful learning (Novak, 1988).

In the models described, learning is generally seen as behavioural or conceptual change. The way in which change should be brought about leads to differing opinions amongst educators. The learning theories and models discussed are usually consciously or subconsciously employed in situations within which teaching and learning is purported to take place. For this to happen, environments should be created such that the system may be transformed from an initial state to a final state where it can now perform. We shall refer to all such situations where we try to propagate and induce learning in some way as learning environments. A learning situation involves

one or several students interacting with text(s), machine tutor(s), live teacher(s) or each other. In the next section, attention is focussed on two environments commonly employed in the classroom, transmission and constructivism. The interplay of the various learning theories within each of these environments is also discussed.

2.3 LEARNING ENVIRONMENTS

The program I have developed uses the constructivist learning environment with information supplied using transmission. In transmission information is directly provided from a source, which might be text material, mass media, a teacher, tutor or lecturer. A large amount of our knowledge is provided by way of transmission. Constructivism proposes that learners use prior knowledge or past experiences in trying to making sense of the world. Constructivism encompasses elements of discovery experiential and reflective learning. Within both these environments, interactive discussions and problem solving are used as means for learning.

Interactive discussion here means student-student, teacher-student and machine-student discussions. Interactions are well suited for feedback. Through interactive question-answer sessions a profile of the student's understanding and deep-seated apprehensions or misapprehensions may be constructed and corrective measures implemented (Webb, 1988). Examples and counter examples can easily be introduced to challenge, correct and clarify student concepts, and explanations can be provided through sharing of various viewpoints. Discussion helps refine ideas successively through interaction. Furthermore, new information can be appropriately introduced and reinforced as the opportunity arises. Interaction can be exploratory, corrective, or reflexive, depending on the focus by the parties involved in the interaction.

In a problem-solving learning environment, the student is provided with the opportunity and assistance to apply the correct principles. Appropriate examples and problems need to be selected to ensure directional movement towards complexity in conceptual attainment. Problems can be graded from the 'cook book' type of simple textbook problems requiring the application of an elementary concept, to general design-type questions requiring a complex linkage of inter-related concepts. As more problems are solved on the application of a concept, the concept is incrementally refined (Michalski, 1986). A problem-solving learning environment assumes that chances of learning are enhanced in applied or contextual settings.

Next, the transmission and constructivist learning models are each discussed in detail. In each case, the learning environment is first elaborated. This follows the learning

theories which are employed or come into play in the environment. Then the role of interactive discussion and problem solving within each environment is discussed.

2.3:1 Transmission

Carbonell et al (1986) term transmission as learning by being told. In this process knowledge is handed down from the teacher to the learner explicitly, either in spoken or in written form. The learner is considered a passive absorber of information and is usually assumed to be like an 'empty bucket' into which knowledge is to be poured. In transmitting the knowledge, links which a teacher wishes to generate in the learner are not always made due to poor processing capability and lack of appropriate knowledge of the students (Osborne and Wittrock, 1985). The entire content to be learned is presented to the learner in final form. In many cases the learners have to internally transform the knowledge received and integrate it with their existing knowledge for usage. They are left to make the expected links as the teacher cannot cover all the possibilities.

In such a learning environment the structure, organisation and inter-relations of the knowledge are stressed. The teacher is able to highlight the reasons and provide explanations as to why something is so or works in a certain way. This can encourage students to question things rather than just accept facts. Learning is hierarchically organised by decomposing the main component into sub-components e.g. the simple concepts are taught first followed by more complex concepts. Benefits of this approach are that the declarative knowledge required can be passed on; a skilled teacher can select illuminating examples to simplify the learning task; the learning can be controlled to slowly expose the learner to ideas ranging from simple to complex; exercises can be chosen and arranged to enable students to practice and master the concept; and errors may be detected as the teacher will have knowledge of patterns of errors and these may be rectified through further transmission of knowledge. Furthermore, the teacher can answer questions like *How come?*, *So what?*, and *What if?*, while passing on the knowledge and while answering student queries.

The disadvantages are that there may be encouragement of rote learning due to emphasis on recall, and the drill and practice type exercises of material taught; students are left to make or induce their own generalisations and discriminations; teaching is based on encouraging recall of facts and drilling of standard exercises; students have to learn what is chosen and considered important for them by another person; and individual learning differences may not be taken into account.

The transmission learning model is based on the traditional stimulus-response learning proposed by Thorndike and Skinner. It is assumed that the responses of students can

be conditioned through drill and practice type exercises and rote learning to enhance recall. To a limited extent this is in line with the information processing model which states that regular repetition enhances retention. The Phenomenological approach and Maslow's skill development theory assume that students will learn because the subject matter is seen by them as important for personal enrichment. Piaget's, Kelly's, hierarchical and schema acquisition learning theories hierarchically organise learning by decomposing main components into sub-components. The teaching and learning is geared towards combining small units to form larger units to create totality and connections as propagated by Gestaltists. Ausubel's theory of meaningful reception learning is employed in the belief that rote learning will be internalised through the teacher highlighting the inter-connections. The implication is that knowledge innately exists and is hierarchically organised with simpler bits combining to form more complex structures. To obtain an understanding of the knowledge that exists a person needs to comprehend the simpler prerequisite structures prior to comprehending the complex structures. Bruner (1985) summarises the implied nature of knowledge in stating that the order in the mind reflects the order that exists in the world. It is assumed that the internalisation of knowledge is dependent on the maturation of the person.

In the transmission environment, interactive discussion involves the Socratic style whereby the teacher asks questions that are intended to encourage students to recognise that which is known and to modify their own understanding (Rolston, 1988). Responses to individual student queries can be elicited and built upon. The interaction is used to determine how much and how well the student has learnt. Students are encouraged to seek clarification through questions. After the subject matter has been taught, textbook type problem solving is used to encourage learning through practice. Appropriate examples and problems may be selected to ensure directional movement towards complexity in conceptual attainment through small incremental changes in the knowledge structures of students. The belief is that if problem sets are appropriately chosen and designed, then incorrect knowledge transformation of students is minimised.

Learning which is solely dependent on the transmission environment may ignore students' prior knowledge. However, there are cases when students will need assistance while making connections or while applying their knowledge. This may be requested by the student, and may be in the form of clarification or additional knowledge to help make connections. In such cases, transmission may be an efficacious method of passing on knowledge.

2.3:2 Constructivism

The learner is seen as an active participant in the construction of his/her own knowledge in trying to make sense of the world. The sense made of any event is seen to be dependent not only on the situation itself but also on the individual's purpose and active construction of meaning (Driver and Oldham, 1986 and Gunstone, 1991). It is implied that knowledge is found or made according to a set of structural rules that are imposed on the flow of experience (Bruner, 1985). Knowledge is assumed to originate through the formation and/or evolution of mental constructs as the learner partakes in an activity. Wheatley (1991) states, constructivists believe that knowledge is intimately related to the action and experience of a learner in a contextual setting and is dependent on the learner. In such a learning environment, conceptual change is brought about by the student. Driver and Oldham (1986) suggest the five phases of orientation, elicitation, restructuring, application and review, for learning to effect conceptual change. In the orientation phase, a program of activities ideally initiated by the student is chosen to uncover the conceptions of students and to build upon them. This gives students an opportunity to develop a sense of purpose and motivation for learning. During the elicitation phase, pupils make their ideas explicit. Claxton (1985) calls these ideas mini-theories built out of experience over a long period of time in trying to make sense of the world. Elicitation may be done by having learners represent their constructions in written form or verbally. Representation in written form may be descriptions, pictures and diagrams, tabulations, headings and sub-headings for classification and concept maps showing the inter-connections and relations. In the restructuring phase clarification, exchange, construction and evaluation of alternative ideas is done through exposure to conflict situations. This is based on the premise that learning occurs when existing ideas are shown to be inadequate. White and Frederikson (1990) call this model transformation whereby increasingly more adequate models are formed in response to the demands of more complex problems or situations experienced by the student. The application phase gives students the opportunity to apply their developed ideas for consolidation and development. This allows the student further experiences which may be translated into concepts which then act as guidelines for new experiences. In the final review phase students are invited to reflect on how their ideas have changed. This enables the newly acquired concepts to fall into place and to be integrated in relation to other previous knowledge. To encourage such conceptual change, Osborne and Wittrock (1985) suggest: (a) pupils be provided with opportunities to consider, contemplate and expand their views of the world and to develop their language whereby they can describe and explain their views; (b) encouraging, challenging and/or reflecting back to pupils the views they expound, to provide better understanding of their views and those of others and the basis for the views.

Osborne and Wittrock (1985) propose the generative learning model as central to the constructivist tradition. The fundamental premise of generative learning is that people generate perceptions and meanings that are consistent with their prior learning. Osborne and Wittrock (1985) stress the importance of knowledge which learners bring to the learning situation and recognise the active construction of meaning while individuals interact with the environment. The need to generate links and to actively construct, test out and subsume meaning requires individuals to accept a major responsibility for their own learning. The Phenomenological Approach comes into play since learning is seen as student initiated. Students will learn and retain those concepts that become more self-related than those that are less self-related. The goals and values of an individual will affect his/her perceptions and provide a measure for judging self actions and those of others. Piaget may be considered a constructivist in the sense that he was interested in how children constructed ideas while learning. He saw intelligence as the way in which the mind organises reality, stating

the essential functions of intelligence consists in inventing and understanding, in other words, in building structures by structuring reality.

(Piaget, 1971 cited in Wilson, 1981)

He saw mental growth as 'substitution' where more mature ideas replace less mature ideas and 'integration' where less mature ideas are combined and refined to arrive at more complex and abstract conceptions. For Ausubel (1968) the single most important factor influencing learning is what the learner already knows. Constructivists stress the importance of ascertaining this prior knowledge and teaching accordingly by building on it. In Ausubel's meaningful learning, the principal content of what is to be learned is not given but must be discovered by the learner before it can be meaningfully incorporated into his/her cognitive structure. The learner must rearrange information, integrate it with existing cognitive structure and reorganise or transform the integrated combination in such a way as to generate a desired end product. Upon completion of discovery learning the discovered content is made meaningful through the process of internalisation. Kelly's theory of personal constructs states that a person's past experience with similar and dissimilar elements in comparison to the current elements under consideration helps him/her to perceive similarities and differences in the formation of constructs. This is in line with the constructivist belief that past experiences are used to interpret current experiences. Learning is seen as a dynamic process whereby students learn better through participation. The information processing model proposes that learning takes place through the usage of receptors and effectors e.g. eyes, arms, hands, voice apparatus, muscles and glands. Participation

increases the receptors and effectors in use during a learning session. Schema acquisition is used in constructing, generating links and subsuming concepts. Rumelhart and Norman (1981) suggest that learning through schema acquisition can happen through:

- (a) accretion which is the coding of new information in terms of existing schemata i.e. adding new information or parts to existing schemata; and
- (b) tuning or schema evolution to bring about the slow creation or evolution of new schemata.

They describe the following learning sequence:

- (i) In encountering a new situation, a person seeks to interpret it in terms of existing schema.
- (ii) If the situation is successfully understood no new schema is created.
- (iii) When no existing schema offers a satisfactory account, one or more schema are restructured or tuned, or a totally new one is created.

During discussions examples and counter examples can easily be introduced to diagnose, challenge, correct and clarify student concepts and explanations provided through sharing of various viewpoints. Discussive feedback can be used to trace and rectify sources of student errors. Champagne et al (1985) put forth the ideational confrontation strategy whereby students are first required to be explicit about their notions after which a discussion ensues through sharing of ideas. It was found that as students attempted to convince others and defend their ideas, concepts became better defined to them. Questions students ask can be used to help them to seek information to enhance their learning.

Problem solving is often carried out as part of the concept acquisition process. Problem solving provides the student with opportunities to apply their knowledge and to be assisted in applying their knowledge. Using knowledge in applied settings provides for knowledge enrichment. Larkin (1981) argues that knowledge (declarative and ancillary) accumulated through problem solving will be stored in readily useable form as procedural and heuristic knowledge, so that a concept can be easily and reliably used. Woolf (1987) defines procedural knowledge as the reasoning used to solve problems; and heuristics as the actions taken to perform transformations in the domain. Problem solving also gives the students practice in transferring their knowledge to different settings through application to different problems. Problem

solving provides opportunities to explore and develop the reasoning employed by students.

It is believed that students learn most effectively when they are actively engaged in applying their knowledge while problem solving. In this last two decades a vast amount of research has been conducted in the area of problem solving (Garrett, 1986; Staver et al, 1989; and Koballa et al, 1990) in domains such as Mathematics, Engineering, Physics, Chemistry and Genetics. Science education aims at developing these problem-solving skills of students through the meaningful application of science concepts. To improve their problem-solving skills, students need to learn and be aware of general problem-solving steps, strategies and problem-solving knowledge. These are the focus of discussion in the next section.

2.4 PROBLEM SOLVING

Problem solving is an investigative task whereby the solver investigates the solution path to reach a goal from given information. Ausubel (1968) refers to problem solving as any activity in which both the cognitive representation of prior experience and the components of a current problem situation are reorganised in order to achieve a designated objective, popularly termed the solution. He considers problem solving as learning which bridges the gap between the learner's existing knowledge and the solution to the problem through transformation of information by analysis, synthesis, hypothesis-formulation and testing, rearrangement, recombination, translation and integration.

Problem solving ability is strongly affected by the context in which the problem occurs. Students may appear as *good* problem solvers in one area of a subject domain and may be *poor* solvers in another. An important factor in successful problem solving is whether students can transfer their knowledge and apply it in solving problems. The features, mentioned in the literature, which good problem solvers need to have for problem solving include learning from further experience, reconceptualising, having general knowledge, having common sense across all dimensions, reasoning by analogy, recognising where their knowledge is inapplicable or insufficient, the ability to check whether their conclusions are reasonable and being able to explicate their reasoning process. The result is that the same problem may be solved in different ways by different individuals due to different problem-solving steps, strategies and knowledge.

2.4:1 General problem-solving steps

General problem solving techniques may not be evident in a learner's skilled behaviour when engaged in relatively familiar tasks but are likely to play an essential role whenever the person has to move into new territory and attempt new learning (Simon, 1980). The general methods proposed in the literature (Newell and Simon, 1972; Larkin, 1980; Reif and Heller, 1982; Pizzini et al, 1989) basically follow the procedures described by Polya (1971) of understanding the problem; finding the connection between data and unknown quantities, finding auxiliary problems if an immediate connection cannot be found, obtaining a plan of the solution, carrying out the plan, and examining the solution obtained. Most of the methods advocate picturing or diagram drawing as this seems to facilitate the problem solving by producing coherence between the inputs of the problem. This also helps to reduce the cognitive load during processing since a person's processing capacity is limited to about five to seven chunks.

Rubinstein (1975) has drawn together ten heuristics as a general guide to problem solving. Four of these heuristics which may be considered general steps are summarised below:

- Before attempting a solution a clear picture of the problem should be obtained by going over the details mentioned in the problem.
- Language, pictures, drawings, graphs, symbols and equations or concrete models in the form of objects can be used to obtain a simplified representation.
- The given information may be transformed so as to make the obscure solution become transparent through a new form of representation.
- Uttering or hearing the correct word or question may result in retrieval of information blocks from long term memory to assist in obtaining the solution.

All these general steps do not mention the guiding principles of when to apply a piece of knowledge. The reasons being that it is difficult to teach problem solving and there are to date no guiding principles that may be globally used to guide students in this aspect. Furthermore, it may be argued that only a limited description of procedural knowledge in a declarative form is possible. All these general methods advocate a sort of redescription of the problem as research shows that this crucially determines how

easily the problem can be solved or whether it can be solved at all. The specific strategies employed in solving problems are discussed next.

2.4:2 Problem-solving strategies

Gagne (1985) states that general problem-solving strategies are used when the solution to a problem is not obvious. Strategies like working forward and means-ends analysis may limit search while strategies like analogy and brainstorming are helpful in expanding search. She defines a strategy to be a goal-directed sequence of mental operations. General problem solving strategies apply to a wide variety of problems, independent of content. Domain specific strategies apply only within a given domain of problems. Carbonell (1986) states that a useful means of classifying different problem-solving methods is to compare them in terms of the amount and specificity of domain knowledge they require. He states that if no structuring domain knowledge is available and there is no useful past experience to draw upon then, weak methods such as means-ends analysis and heuristic search have to be used. If specific domain knowledge in the form of plans or procedures exists then these may be recursively used in solving the main problem and any sub-problems that might arise. If on the other hand only general plans apply then these can be used to reduce the problem. If no specific plans apply but the problem resembles one solved previously then analogical transformation can be applied to adapt the solution of the similar past problem to the new situation. Ten commonly used weak methods and general strategies are summarised below.

Generate and test: The problem solver simply generates solutions one at a time and tests to see their applicability. Control during search of solutions may be done by generating only those that possess at least some of the properties that define the solution (Newell and Simon, 1972). Sleeman (1987) recommends this method as a good means for students to learn how to explore all possible solutions and to systematically explore the complete solution space. Ausubel (1968) suggests this is an inevitable method when no meaningful pattern of relationships exist.

Means-ends analysis: is a powerful form of working backwards whereby assessment is made between the current state of knowledge of the problem and the desired goal state (Newell and Simon, 1972). This assessment is used to select and apply an operation or operator to reduce the difference. This is recursively done until the goal state is reached or all the operators in the problem space have been used. The method applies to problems for which there are no specific or better methods.

Problem decomposition: is the process whereby a large, complex problem is broken into a collection of smaller sub-parts (Rolston, 1988) or sub-problems. A sub-

problem is any problem whose solution facilitates the solution of the original problem (Reif, 1981). Each of the sub-problems may in turn be decomposed into further sub-problems and independently solved. The process of successive decomposition is continued until all the resulting sub-problems are readily solvable. These separate pieces are then combined to form the solution to the original problem.

Problem abstraction: is the process of concentrating on a generalised representation of the most important elements of a problem (Rolston, 1988). This allows the solver to construct a simplified representation by ignoring the minor details and focussing on the problem by retaining certain central features. This abstracted problem is first solved and its solution used as a guide to solving the original problem (Larkin, 1980). The successive refinement or modification of the abstraction of the problem by including more features in steps facilitates search and allows one to focus on a couple of features at a time.

Working backward: uses goal-directed strategy. The goal of the problem is used to construct a tree of intermediate quantities until one arrives at the quantities specific as given in the problem (van Weeren, 1983). Rules acquired as a result of previous problem-solving experiences are used to consider only those paths which might potentially lead to the solution. The problem with the backward strategy is that there are no specific guidelines as to the process of acquiring the rules or of then selecting them appropriately during problem solving.

Forward strategy: is a data driven strategy. The initial problem description is used to generate solutions using the knowledge base (Reif, 1983). In the case of well defined problems, the solution is accomplished with the application of constraint satisfaction. Such constraints are generated from information contained in the particular problem. The operations one chooses to perform on the initial data are not goal constrained as in means-ends analysis and might lead to fruitless directions (Gagne, 1985). When knowledge is available, solutions usually show forward and top down development.

Brainstorming: is a good strategy for increasing the number and quality of solutions (Osborn, 1963). The problem is first defined, and as many solutions as possible are generated without criticism. Criteria are then developed to judge the viability or applicability of the solutions and the best solution selected (Gagne, 1985). The process includes elements of both working forward and means-ends analysis.

Analogy: Carbonell (1986) defines analogical problem solving as consisting of transferring knowledge from past problem-solving episodes to new problems that

share significant aspects with the corresponding past experience. The transferred knowledge is then used to construct solutions to the new problems. Reasoning by analogy is often used when there is little knowledge in the domain of the problem (Gagne, 1985). The problem has to be represented such that the representation may be used to access knowledge in a familiar domain relevant to the current situation and to evaluate the utility of the knowledge.

Envisioning: relies on being able to visualise what is happening in the problem. It involves generating a series of sequential snapshots which describe what might or could happen (de Kleer, 1977). This provides the qualitative knowledge to roughly predict what will happen at various points in a given problem situation or state. Domain specific principles and rules are then applied to obtain a qualitative causal description of the problem situation. This knowledge is represented as transformation and analysis rules and applied to solve the problem.

Heuristic search: is the process of using operators to guide search in a problem space (Newell and Simon, 1972). The search can go depth wise or breadth wise, and can backtrack and continue in different directions. If the problem space is large then it is usually transformed into a more manageable form by first using problem decomposition. For the search to be constructive, knowledge of possibilities is necessary to generate trial solutions and knowledge of constraints is necessary to guide the search for the solution. Guessing or plausible reasoning is also used. Kolodner (1984) mentions that sometimes a search for related operators or information called alternate context search is required especially when the operator being searched is too obscure.

Most solvers use a combination of the above-mentioned problem-solving strategies. The strategies used depend on the amount of factual and procedural information available and the experience of the problem solver. Many solvers use strategies without realising the style adopted. In general, search processes dominate much of the problem-solving behaviour of novices as their understanding is both shallow and fragmented. As experience is gained and knowledge accumulated the search becomes more selective.

2.4:3 Problem-solving knowledge

Problem solving requires the integration of knowledge types which can be classified as content and structure.

Content can be declarative or procedural. Declarative knowledge consists of facts within a domain. Procedural knowledge determines when a piece of declarative knowledge is applicable and under which circumstances, encompassing the possible assumptions that can be made in solving problems. Procedural knowledge is sometimes referred to as ancillary knowledge which includes the appropriate interpretation of the concept; the applicability conditions of a concept; knowledge to prevent errors and to discriminate errors from correct situations (Reif, 1985).

Structure is the inter-relationship and connections between pieces of related information. This organisation of knowledge enables it to be related to and applied to solving problems. If the connections do not exist then *functional fixedness* occurs. This happens when a piece of knowledge is seen to have only its conventional use and not thought of for other purposes (Anderson, 1985).

The aim of exposing students to the various strategies and knowledge is to move them from the inexperienced problem solver (novice) state to the experienced (expert) mode. Davis (1989) suggests five stages of performance from novice to expert. Stage one (novice) uses context-free facts and rules to produce actions based on these facts. Stage two (advanced beginner) learns to recognise elements not defined in terms of context-free facts. Stage three (competent) learns to organise thoughts in terms of plans and goals. This involves choices and decisions. Stage four (proficient) intuitively recognises a familiar situation and the goal. Stage five (expert) has the ability to intuitively know the sense of the situation and from prior experience knows what to do.

Regardless of the development of instructional systems and technological advancements (e.g. computer-assisted instruction, films and curriculum materials) which enable the student to work independently, classroom learning will continue to be mostly spent in group instruction with a teacher. In my model I am employing a constructivist learning environment coupled with the use of transmission. To realise such an environment, teachers must be aware of their role. They should know when to use transmission sparingly as the need arises and still enable students to be in control of their own learning. The role of a teacher in enhancing a learning environment is discussed in the next section.

2.5 TEACHER'S ROLE IN THE LEARNING ENVIRONMENT

As a participant in the learning environment, the teacher or tutor acts as provider, facilitator, enhancer, enquirer and guide. In these roles, the teacher attempts to replace irrelevant ideas and to evoke ideas to which new ideas can be related. In doing this the teacher needs to have an understanding of the:

- nature and structure of the subject domain knowledge to be of useful assistance in providing help and guiding students to the host of ancillary knowledge required to interpret a concept appropriately when requested. Presently, the student is not made aware of the connections and is expected to make them. Without experience the average student fails to do so.
- student's prior knowledge. Teachers need to interact with the students to learn of their conceptions, the knowledge they bring to the classroom and to probe their understanding.
- student's strategies in acquiring knowledge and problem solving to provide explanations and help at the correct level when requested. Chi et al (1989) suggests refining and expanding the conditions of student actions, explicating the consequences of their action, providing a goal for a set of actions, relating the consequences of one action to another and explaining the meaning of a set of quantitative explanations.

The goal of a good teacher should be to make themselves continually less necessary to the learner. In a constructivist environment, the teacher must take pains to ensure that they do not dominated the learning environment and that the student should be the central figure in initiating and conducting any interaction. Anderson et al (1990) and Webb (1988) advocate that a teacher or tutor should provide suitable feedback at appropriate times to enhance learning. They stress that feedback given should take into account the history of the student and the errors made. Feedback need not only be error correction but can be in the form of:

- assigning appropriate remedial or strengthening work;
- helpful hints and suggestions on where to look for subject material for application;
- how to make relational links from text or other material.

A teacher, preferably in consultation with the student, should basically select an educational objective, make changes in teaching strategy, evaluate the effectiveness and

update the strategy to suit the student. Within a transmission-constructivist learning environment the teacher needs to keep the following in mind:

- Start from the existing knowledge of the students.
- Interact with students at a rate which is consistent with their ability.
- Enhance the learning of the students by providing hints.
- Encourage students to diagnose their learning.
- Provide opportunities for students to apply and test the knowledge acquired.
- Provide knowledge through transmission when the need arises.
- Have an open mind and be willing to learn as well.

2.6 CONCLUSION

Thirteen learning theories have been categorised into 7 groups by the learning procedures implied. The two formal learning environments have been categorised as transmission and constructivism. Both these environments use interactive discussion and problem solving. These have been discussed to lay the ground for the learning environments and learning theories on which the problem-solving program developed is based. It has been argued that while it is necessary to make students responsible for their learning through a constructivist environment, at times it is necessary to use transmission to provide information to enable the students to carry on with their learning.

The general steps used in solving problems have been discussed and the various strategies commonly used by students while problem solving have been outlined.

In the next chapter, related research in the use of computer-assisted programs in providing tutoring help to students is discussed. The advent of artificial intelligence and expert systems have brought about intelligent computer assisted-instructional systems. The type of tutoring systems produced to date and their salient features which set them apart from traditional computer-based instructional systems are highlighted.

CHAPTER 3

RESEARCH INTO COMPUTER-ASSISTED LEARNING

3.1 INTRODUCTION

Since the 1960s computer technology has been applied to education. It has helped in formal as well as informal learning in the home and the workplace. Research in artificial intelligence and the deficiencies of traditional computer-based instructional systems has brought about the development of intelligent tutoring systems (ITS) or intelligent computer-assisted instructional (ICAI) systems. Computers have helped learning by make experimentation easier and safer for the student to experience (Brown et al, 1982). In the early years of its inception computers were used more in areas such as course-scheduling, test-grading and the management of teaching aids (Simmons, 1984). In later years the application has gone from drill and practice type exercises towards programming, games, simulations and intelligent computer-assisted instruction (Barr and Feigenbaum, 1982).

Traditional pedagogic methods in the classroom provide a useful function in conveying declarative knowledge for learning but are often inadequate for teaching complex skills. Intelligent computer-assisted tutoring systems are aimed at trying to teach such complex skills. These systems are computer programs which use artificial intelligence techniques for representing knowledge and interacting with students. The aim has been to provide students a flexible learning environment. Attempts have been made to include the student's learning behaviour, and teaching and tutoring strategies. Designers of intelligent computer-assisted instructional systems have placed great effort in developing and producing ways to motivate students to interact with the systems. This reflects efforts to provide an environment that facilitates and encourages learning (Stevens and Collins, 1980). However, many of the designers of such systems have neglected the learning theories upon which the framework for a learning environment should be based.

A historical perspective of artificial intelligence, expert systems and traditional computer-based instruction leading to the development of intelligent computer-assisted instruction is outlined in the next section. The differences between a traditional computer-based instructional system and an ICAI system are highlighted. Then follows a description of the general architecture of an ICAI system. The processes of an ICAI system which are representing, searching and retrieving, and collating knowledge are explained. Thirty ICAI systems are then briefly described

and reviewed to highlight the implicit learning theories upon which each is based, and the extent to which transmission, problem solving, interactive discussion and constructivism are used. The program features of an ideal interactive ICAI system using a constructivist learning environment coupled with transmission are outlined. A discussion of the extent to which these can be realised with the present state of art of computers concludes the chapter.

3.2 HISTORICAL PERSPECTIVE

Traditionally, computer-based instruction (CBI) has used canned text for carrying a student through a question-response sequence. The most common application has been to use pre-stored materials (test items or question-answer sequences) for assessment and tutoring, the students' responses being restricted to pre-stored options that have been anticipated by the designer (Hartley, 1988). The student is given some instructional text and a question to answer in the form of choices. The student is told whether the answer is right or wrong. Remedial instruction is provided in the form of branching to an easier question, or as an explanation (Tait and Hughes, 1984). Feedback may be specific to the item itself or may make suggestions for further study. This provides questions, answers and learning materials all in a single session. Computer-based instruction has also used computer simulations which incorporate a model of a process e.g. an on-line wave simulator (Bork and Robson, 1972). Such programs allow 'scientific' investigations in a controlled, guided, safe and economical way (Hartley, 1988).

Canned text can be used to accommodate different learning rates by adding or subtracting optional components. However it is not an appropriate way of addressing misconceptions of students. Research has shown that merely providing canned text does not correct the misconceptions. The ideas of students need to be challenged before they will substitute their conceptions with alternatives put forth (Mestre and Touger, 1989). Usually there is little scope for the student's response to affect the lesson in any major way. Present applications of artificial intelligence in CBI provide some educational environment control of the learning experience by students. Such programs are termed intelligent computer-assisted instructional teaching and learning tools. The advent of artificial intelligence and the deficiencies of traditional computer-based instructional systems has brought about the development of intelligent computer-assisted systems, popularly termed intelligent tutoring systems. These are essentially expert systems using artificial intelligence techniques to assist in instruction and learning. Table 3.1 summarises the differences between traditional

computer-based instructional systems and intelligent computer-assisted instructional systems.

Aspect	CBI	ICAI
1. Learning	Uses transmission and is highly directive and geared towards mastery learning.	Uses learning by discovery and is student centred.
2. Solution	Program provides the solution steps and leads the user stepwise to the solution.	Is able to provide and talk about alternative solution strategies.
3. Dialogue and Explanation	Fixed choices of specific canned text.	Can vary the level of detail in consultation with the student model.
4. Instruction	Fixed path and style for all students e.g. drill and practice, games and simulations which are sequentially arranged.	The student model is used to determine the instructional sequence to cater to student needs.
5. Representation	Uses hierarchical arrangement of subject matter or an algorithmic approach.	Uses AI knowledge representation techniques for organising knowledge in multiple ways.
6. Responses	Tutoring material presented based on a set number of keyword responses.	Based on the student and dialogue history using a student model.
7. Control	Controlled hierarchically or algorithmically in specific paths.	Decided by the interaction between student and system.
8. Knowledge	Subject matter and teaching strategies are all chunked together.	Subject matter, teaching strategies and discourse rules are separated.
9. Teaching	Problem areas identified through question and answer sessions and corrective feedback through sequentially arranged subject matter.	Identifies the problem areas as well as the underlying misconceptions and provides interactive feedback for students to correct themselves.
10. Tutoring Style	Procedure following.	Uses problem solving.
11. Questions	Students are restricted to pre-programmed question choices using keywords.	Students can ask various questions, carry on a dialogue and even pose problems.

Table 3.1: Differences between CBI and ICAI systems

Artificial intelligence is a complex topic encompassing diverse applications in a plethora of fields. No single definition is widely accepted by every researcher in the

various fields. To convey an impression of artificial intelligence some of the references made to it are given below:

Minsky (1968): Artificial intelligence is the science of making machines do things that would require intelligence if done by men;

Gardner (1985): Artificial intelligence seeks to produce, on a computer, a pattern of output that would be considered intelligent if displayed by human beings;

Putnam (1988): The aim of artificial intelligence is to develop software (and hardware) that will enable computers to simulate or duplicate the achievements of what we intuitively recognise as “intelligence”.

Artificial intelligence has branched off into knowledge engineering and cognitive science. This is evident from the following headings used to categorise the major fields represented in the proceedings of the International Joint Conference on Artificial Intelligence (1987):

1. Natural language and architecture;
2. Representation of knowledge;
3. Aids to programming;
4. Problem solving and search techniques;
5. Theorem proving;
6. Vision and perception;
7. Robotics;
8. Tutoring in various fields;
9. Consultation, diagnosis and applications in professional fields;
10. Planning and debugging;
11. Pattern description and recognition;
12. Machine and human learning;
13. Cognitive modelling;
14. Reasoning; and
15. Logic.

Cognitive science, within the artificial intelligence field, looks at the representation and processes used by humans for acquiring, accumulating, representing, and using knowledge. Knowledge engineering endeavours to capture these within a system. Lindsay (1988) defines knowledge engineering as the development of expert systems, including design, knowledge acquisition, knowledge representation, testing and

implementation (of a system). An expert system is a software program designed to perform problem-solving functions. The essence of these functions is captured by the following description put forth by Hayes-Roth et al (1983):

The area of expert systems investigates methods and techniques for constructing man-machine systems with specialised problem-solving expertise. Expertise consists of knowledge about a particular domain, understanding of domain problems and skill at solving some of these problems.

The initial research into ICAI using artificial intelligence techniques was to test their applicability for improving instruction. In later years cognitive learning theories have been incorporated in developing computer-based instructional models. Practically all the ICAI systems provide instructional format in terms of games, simulations and tutorials. To date intelligent tutoring systems have been developed for interesting and well structured domains such as Biology, Chemistry, Computer Science, Engineering, Genetics, Geography, History, Law, Mathematics, Medicine and Physics. Many of the systems are prototype research projects and have not been extensively applied to any classroom setting. Insight into ICAI systems may be obtained through a discussion of the main components. The function of each of these components is briefly discussed next.

3.3 ARCHITECTURE OF ICAI SYSTEMS

The pioneer expert systems on intelligent computer-assisted instruction (ICAI) concentrated on using production rules to represent domain knowledge, skills and problem-solving methods (Barr and Feigenbaum, 1982). From the late 1970s, the student's learning behaviour, and teaching and tutoring strategies have been incorporated (Sleeman and Brown, 1982). The main components of most ICAI systems include the expertise module, the student model, the tutoring module and an interface. The interface is usually considered to be part of each of the other three acting as a front end.

3.3:1 Expertise module

This module includes the declarative (factual), procedural (problem-solving skills), teaching strategies and multiple representations of answering students questions (Barr and Feigenbaum, 1982). Yasdi and Ziarko (1988) subdivide the expertise module into a knowledge base, a monitor which guides the user through different levels of

interaction, an inference engine to perform the deductive function, a learning component to enable the expert to introduce and update rules, and an explainer to explain the reasoning, inferences and conclusions. Sometimes a pedagogical module is also included. This performs the function of interacting with the student for teaching problem selection and updating the student model (discussed in the next section). It includes the skills and knowledge which need to be taught and incorporates teaching and learning strategies to enhance the interaction.

3.3:2 Student model

The student model is used to represent the student's understanding of a domain at any time during a tutoring session (Rolston, 1988). It is used to identify what the student knows and understands, does not know, or knows incorrectly. It contains a repertoire of the particular learner's skills and misunderstandings (Nawrocki, 1987). Woolf (1987) reiterates that the student model should also contain common errors and misconceptions. An ideal interactive student model should trace each elementary move taken by the student and constantly update itself. Clancey et al (cited in Park et al 1987) lists four major information sources for maintaining a student-model: (a) student problem-solving behaviour (or performance progress) observed by the system; (b) direct questions asked of the student; (c) assumptions based on the student's learning experience; and (d) assumptions based on some measures of the difficulty of the subject material.

The common techniques used for structuring a student model are an Overlay model (Goldstein, 1982) and a Buggy model (Brown and Burton, 1975). In the overlay model the student's response or performance is compared to that of an expert for the same task. Each skill is then marked as understood or otherwise using the evidence from student protocols. In this model the student's knowledge is entirely represented relative to that of the expert's. In the Buggy model, the student's knowledge is modelled by noting the points at which it deviates from the expert's. A bug is considered to be a deviation from the procedure and might or might not be a flaw in the procedure. Unlike the overlay model, the buggy model assumes that to be correct the student need not reason like the expert.

3.3:3 Tutoring module

The aim of the tutoring module is to offer interaction which is tailored to the students' strengths and weaknesses and style of learning. This module integrates knowledge about teaching methods, the student model and the subject area to organise the overall tutoring session. In consultation with the student model, it decides what action is appropriate to help the student. Actions may be in the form of communication with the student; selecting problems for the student to solve, monitoring and criticising

performance of the student, providing assistance upon request, selecting remedial action (Bumbaca, 1988), volunteering assistance at the appropriate time, suggesting ideas when the student appears to have none, and anticipating future actions based on current activities performed by the student. The system should have knowledge of the multiple ways in which a problem is solvable. This means knowing how to ask the right questions, and how to focus on the appropriate issue. The system acts as a partner, rather than being passive and disinterested (Woolf, 1987), and should be designed so that the student has to think before responding to a question.

The tutoring module should also contain pedagogical knowledge. By applying different teaching styles the system may arrive at the type of style a student prefers (Bumbaca, 1988). If a student prefers to be taught through the use of examples, the system should note this from the student model and oblige. The teaching methods in most ICAI systems are generally one or more of the following:

In Diagnostic modelling: the program determines the student's level of understanding by posing tasks and obtaining and evaluating responses (Collins, 1976).

For the Socratic technique: the student is questioned and led to recognise where he/she is going wrong; self debugging of misconceptions is then possible (Carbonell, 1970).

The Coaching method: engages the student in skill acquisition and general problem -solving skills through activities such as computer games (Burton and Brown, 1982).

In Constructive debugging: the student is given information to determine why an error occurred, and can then try to correct the error (Burton and Brown, 1982).

The Adviser: system is explicitly called upon when the student requests hints, expansions, or critiques (Rolston, 1988). This is a common style when the student is in control of the situation and has sufficient knowledge to know roughly what to request. The interface needs to be sufficiently elaborate for requesting information in greater depth.

The tutoring module should be such that students feel they have gained an understanding of each situation. Recently, ICAI systems have experimented with a constructivist model of learning, whereby the learner is an active agent whose cognitive structures are undergoing change (Duchastel, 1988).

3.3:4 Interface

The interface is the module that communicates and interacts with user. The interface should be able both to understand the user, and to generate responses. Understanding requires that the system recognise alternative wordings of the same thought, since individuals are poor at paraphrasing questions and phrase questions in a variety of ways (Brown et al, 1982). The interface therefore analyses the student input, parses it and decides what the student intends to do. Generating responses requires the interface to indicate to the user how to ask for information, particularly when the user asks questions which the system does not know how to handle or does not understand. Responses are also required when the user is at a loss as to the next step in the discourse (Brown et al 1982). Clancey (1987a) in his GUIDON system allows students to specify what they know, what they want to know more about and what they want to ignore. If a question is unnecessary the student is told this and an explanation provided.

Interfaces often use menus to help the user. As choices are made menus arranged in hierarchical order can become increasingly specific (Mestre and Touger, 1989). The interface should also help the user to search for information (e.g. a useful principle), or how to determine whether a principle is applicable (Larkin, 1981). Interfaces for tutors should minimise the working memory and processing load of students when they are problem solving (Anderson et al , 1990).

Apart from these components, the system should have a component for data collection to provide feedback to the designer. Clancey (1987b) recommends collection of four types of data: (i) student options, including requests for data and assistance; (ii) modelling decisions made by the program; (iii) tutoring decisions made by the program; and (iv) discourse situations during the dialogue. Buchanan et al (1983) suggests inclusion of a 'gripe' facility to record complaints.

ICAI systems need to handle a great deal of information. The expertise module, the student module, the tutoring module and the interface all require a knowledge base so that the user can access knowledge, make inferences and be provided with explanations. Knowledge is accessed by performing search and control techniques on the knowledge represented. The knowledge representation models and the access methods are discussed next.

3.4 KNOWLEDGE REPRESENTATION AND ACCESS

The knowledge base needs to be accessed for different purposes. A combination of representational models is therefore required. The main representational models used are production systems, frames and scripts, blackboard configuration, semantic networks and formal logic. Information can be displayed in pictorial, graphical or numeric form. The program developed in this thesis uses production systems, and frames and scripts. A brief description of each is given below.

3.4:1 Production Systems

Production systems use sequences of IF-THEN and IF-THEN-OR-ELSE decisions called production rules. Production systems were developed to represent problem-solving knowledge by Anderson (1983) for his PUPs (PenUltimate Production System) theory for geometry and algebra; and Newell and Simon (1972) for their model of human cognition. Production rules have two major categories, inductive inference e.g. IF <premise> THEN <conclusion>, and deductive inference e.g. IF <condition> THEN <action>. When the IF part of the rule is satisfied, the action specified by the THEN part of the rule is performed. The IF part of production rule can be a single assertion (Figure 3.1) or a list of assertions (Figure 3.2) which need to be true for the conclusion to be valid. Similarly, more than one conclusion can be inferred or deduced from a set of assertions. Individual rules are sometimes assumed to correspond to the units of human problem-solving knowledge, and the sequence of rules to correspond to the process of human thought.

IF	There is only one force acting on the body.
THEN	The body is not in equilibrium.

Figure 3.1: An example of a production rule with a single assertion and conclusion

IF	There is more than one force acting on the body. The forces are all in parallel or anti parallel directions. The forces sum to zero. It is possible for the body to undergo a turning effect. The net torque about an axis of rotation is zero.
THEN	The body is in dynamic equilibrium.

Figure 3.2: A production rule with five assertions and a single conclusion

IF	The body is in static equilibrium.
THEN	The forces acting on the body sum to zero. The net torque acting on the body about an axis of rotation is zero. The body is stationary

Figure 3.3: A production rule with a single assertion leading to three conclusions

3.4:2 Frames, Scripts and Schemata

Research shows that when encountering new situations in everyday life people use a well organised and connected body of knowledge from past experiences (Bartlett, 1932). This chunk of connected knowledge organisation is referred to as frames, scripts or schemata. Minsky (1975) defined a frame to represent stereotypical situations such as being in a certain kind of living room or going to a child’s birthday party. Schank and Abelson (1977) used a script to represent the information related to going to a restaurant. Bobrow and Norman (1975) suggest that a memory unit or schema may be a short list of the properties associated with an object or word; or knowledge organised round an event description. The terms frames and scripts are commonly used in expert and ICAI systems, while schema is commonly used in cognitive psychology. Since features of frames, scripts and schemata are common, the term frame will be loosely used to encompass all three ideas.

Hayes-Roth et al (1983) capture the essence of frames when they liken it to a prefabricated house, where the basic design decisions have been made but many other decisions are made to suit individual needs. Frames are organised as top and lower level knowledge. The top level knowledge which is always true of a certain situation, class of objects, elements or events is fixed and placed in slots as default knowledge. Items 1, 2 and 3 in Figure 3.4 constitute the top level knowledge. The slots are filled by default knowledge dependent on prior choices the user makes. The lower level knowledge is represented by empty “slots” which are filled in by the user depending on the specific case, falling generally within the realms of the top level knowledge. Items 4 to 14 in Figure 3.4 represent the lower level knowledge. The empty slots are filled as the student supplies the information or requests the system for the information. The completed frame of Figure 3.4 is shown in Figure 3.5. The information supplied for each item in the lower level is dependent on the default knowledge stored at the top level. Items lower in the structure within the lower level inherit properties from items higher in the structure as well. For example, in the completed frame of Figure 3.5 the translational kinetic energy (item 11) is dependent

1	Position	At the bottom of the loop
2	Object	Solid ball
3	Motion	Rolling
4	Mass	
5	Radius	
6	Height from reference level	
7	Linear velocity	
8	Angular velocity	
9	Moment of Inertia	
10	Gravitational acceleration	
11	Translational kinetic energy	
12	Rotational kinetic energy	
13	Gravitational potential energy	
14	Total energy	

For information on the question refer to Table 4.7 in the next chapter.

Figure 3.4: Frame for a rolling ball at the bottom of the loop

1	Position	At the bottom of the loop
2	Object	Solid ball
3	Motion	Rolling
4	Mass	M
5	Radius	r
6	Height from reference level	h
7	Linear velocity	v
8	Angular velocity	ω
9	Moment of Inertia (I)	$\frac{2}{5} mr^2$
10	Gravitational acceleration (g)	9.81ms^{-2}
11	Translational kinetic energy	$\frac{1}{2} mv^2$
12	Rotational kinetic energy	$\frac{1}{2} I\omega^2$
13	Gravitational potential energy	Mgh
14	Total energy	items 11 + 12 + 13

For information on the question refer to Table 4.7 in the next chapter.

Figure 3.5: A completed frame for a rolling ball at the bottom of the loop

on the default knowledge in the top level as well as the mass (item 4) and the linear velocity (item 7). To each slot are attached procedures in the form of rules which are performed as information is obtained. An example of a procedure may be to obtain the information for slot 14 (if left empty) by adding the information in slots 11, 12 and 13. The procedures also have certain conditions attached for performance.

Waterman (1986) mentions three useful types of procedures often attached to slots, namely addition of information into a slot, removal of information from a slot and information needed from an empty slot. Each of these start a whole chain of events and perform the functions to retrieve, obtain or collate information within the frame. Also attached to each frame are differing types of information. Minsky (1975) pinpoints three categories of information attached to frames:

- procedural: on how the information is to be used;
- predictive: what can be expected to happen next; and
- alternatives: what to do when expectations are not met or information is missing.

The amount of information that may be attached to frames can be infinite depending on the usage to which the frame is to be put.

3.4:3 Accessing knowledge

Most problem-solving systems access knowledge through search and control techniques. In search techniques the solution is referred to as a goal, the initial conditions are called the start state or the zero node and all the possible paths leading from the initial condition to the goal state constitute the search space. The solution path is the chosen path that leads from the initial state to the goal state. The search space may be likened to a complex tree diagram of interconnected nodes. Search can be performed using a forward (top down) or a backward (bottom up) reasoning, or a combination of both, termed bi-directional search. Forward search is also termed data-directed or data-driven search. Backward search is also called goal-driven or goal-directed search.

Clancey (1987a) describes the process involved in a search as forward chaining or backward chaining. Forward chaining applies rules whose conditions are satisfied by the problem description and backward chaining as applying rules whose conditions are satisfied by the goal. In forward and backward chaining, each time a set of rules are tested the first (or topmost) rule that matches is executed. To start the search a fact is usually placed in the working memory. For forward chaining, the first (or topmost)

rule in which an assertion matches the fact in the database is used. If the other assertions in the rule are satisfied then the conclusion of that rule replaces the fact in the working memory. This conclusion then becomes the fact in the working memory and is in turn matched against the assertions of the next topmost rule until a match is obtained. This process carries on until the goal or solution is reached or until all the rules have been tested and the rule base exhausted.

In the case of backward chaining, the fact placed in the working memory is first tested against the assertions in each of the rules starting with the topmost rule until a match is found. If no match is found, then it tests the fact against the conclusion of each rule again starting with the topmost rule. Let's say it matches the fact against a conclusion and finds a rule whose conclusion matches. It then places the first assertion in that rule as the fact in working memory and matches it against all other assertions and conclusions. If it finds a match then it replaces the fact in the working memory with the matching assertion or conclusion and so on until the goal is reached, or until all the rules have been used or tested. In both forward and backward chaining the essential difference is the method in which data and rules are searched (Waterman, 1986).

Forward chaining is appropriate when there are many possible and equally acceptable goal states and a single initial state while backward is more applicable for tasks in which there exists a single goal state and a plethora of relevant initial information (Brownston et al, 1986). Most systems use a mixture of forward and backward chaining.

Intelligent computer-assisted instructional systems have been constructed and applied within a wide range of fields. Very few of these systems make explicit a central or set of core learning theories or learning environments upon which a system is based. However, many of these systems have implicitly employed learning theories and environments. 30 ICAI systems are next reviewed with a view towards uncovering the learning theories they employ and the learning environments within which they operate.

3.5 LEARNING THEORIES EMPLOYED IN ICAI SYSTEMS

Many of the ICAI systems are prototype research projects and have not been extensively applied to any classroom setting. Barr and Feigenbaum (1982), Sleeman and Brown (1982) and Kearsley (1987) give a good description of many existing ICAI systems. Carbonell's SCHOLAR system (Carbonell, 1970) is seen as the first

application of artificial intelligence techniques in computer-assisted instruction. Since then numerous projects have been researched in various academic fields. A brief description of 30 of the existing systems developed for instructional purposes is given below.

1. Scholar (Carbonell, 1970) used to assist students in teaching weather in South American Geography using the Socratic style of question-answer technique. It diagnoses students' misconceptions and presents materials to enable students to see their errors.

2 Excheck (Smith et al, 1975) is an instructional system which presents courses in logic, set theory and proof theory in mathematics. It presents lesson material followed by exercises consisting of theorems which need proving. The program can follow mathematical proofs presented by students and provide feedback.

3. Sophie (Brown et al, 1975) provides the student with an environment to explore ideas, create hypotheses about a situation and receive immediate feedback in the domain of electronic troubleshooting. Sophie (A Sophisticated Instructional Environment) provides the environment through simulation of an electronics laboratory.

4. Isaac (Novak, 1977) is able to read, understand, solve and draw pictures of Physics problems (rigid body statics) stated in English. Abstract mapping of the problem is created and specialist programs used to solve different aspects of the abstract representation.

5. Newton (de Kleer, 1977) is a program which applies the idea of envisionment to the domain of simple classical mechanics. Envisionment is the ability to predict what will happen without actually seeing it happen. NEWTON is able to qualitatively solve problems and uses mathematical equations only when qualitative reasoning fails in producing a solution.

6. Buggy (Brown and Burton, 1978) is a program which determines students' misconceptions in basic arithmetic skills. It identifies the mistake the student makes and provides a mechanism for explaining why the mistake is being made

7. Bacon.5 (Langley, 1979) uses production rules to discover empirical laws in Physics. The system uses strategies for finding regularities, defining terms, ignoring differences, collapsing hypotheses and determining irrelevant variables to arrive at general conclusions.

- 8. Mecho** (Bundy et al, 1979) solves a wide range of mechanics problems using both English and predicate calculus statements. A formal representation of a problem is obtained from an English statement. and transformed into a representation to solve the problem.
- 9. West** (Burton and Brown, 1982) a computer game environment teaches patterns for the manipulation of arithmetic expressions. A guided discovery learning environment assists students to construct an understanding of a situation or task.
- 10. Why** (Stevens et al, 1982) is an extension of the SCHOLAR program. Rules used by human tutors are implemented to guide the tutoring dialogue on the causes of rainfall.
- 11. Wumpus** (Goldstein, 1979) has a game playing environment to teach logic, probability, decision theory and geometry. A player is required to track down and slay the vicious Wumpus. The system uses the interaction of four specialist modules namely Expert, Psychologist, Student model and Tutor.
- 12. Able** (Larkin, 1981) learns by increasing its knowledge of mechanics problems as it solves problems. Production rules are stored with the necessary conditions each time it applies a mechanics principle and an action which summons the knowledge the next time the principle needs to be applied. Starting with means-end analysis, as it acquires knowledge its behaviour becomes dominated by knowledge it has learned.
- 13. ACE** (Sleeman and Hendley, 1982) analyses complex explanations. Problem-solving monitors (PSM) are used to help students interpret Nuclear Magnetic Resonance Spectra. The user is assumed to know the algorithm but is given help when requested with solving a particular problem.
- 14. Integrate** (Kimball, 1982) tutors calculus in the domain of symbolic integration. The program is able to transmit problem-solving heuristics, generate appropriate examples, successfully deal with student posed problems, handle a wide range of student backgrounds capability and learns problem-solving heuristics from students.
- 15. Quadratic** (O'Shea, 1982) a self improving quadratic tutor, conducts experiments on teaching strategy in mathematics. The production rules are altered in accordance with data collected to improve student performance.
- 16. Leeds Modelling System (LMS)** (Sleeman, 1982) Given the problems (in Algebra) to be worked and the students' answers, the system tries to infer model(s) of

the student behaviour. The aim being to obtain the same answer as the student, using the inferred model.

17. Macsyma (Genesereth, 1982) is a large interactive computer system designed to assist mathematicians, scientists and engineers in performing symbolic manipulation of mathematical expressions. The system can be used to analyse the student's steps to a solution thereby inferring the problem solving plan. This plan can then be used to generate remediation tailored to the student.

18. Spade (Miller, 1982) can be used to debug a program plan. It interacts by introducing to the student a vocabulary for describing plans, bugs and debugging strategies. The students' program development process is organised by supportive prompting using a menu of design alternatives for the student to select and edit the program being developed.

19. Spirit (Barzilay, 1985) an intelligent tutoring system for probability theory has two modes of operation. The program either observes the student, intervening only when things are going wrong or manages a question and answer type session or dialogue. The mode of operation is constantly changed in accordance with the level of the student .

20. Dynamic Modelling System (DMS) (Ogborn, 1987) has been employed in mathematics modelling in Physics for topics like dynamics, wave motion and circuitry. Equations are entered to describe a model and initial values of the variables are provided. The system then outputs the calculated values and provides graphical depiction.

21. Guidon (Clancey, 1987a) teaches medical diagnosis by engaging a student in a dialogue. The dialogue discusses a patient suspected of having an infection. Students are taught how to use clinical and laboratory data to arrive at diagnostic inferences. The teaching rules are structured into discourse procedures and the diagnostic rules are hierarchically grouped.

22. Proust (Johnson, 1990) finds a bug in novice programs, determines how the bug may be corrected and suggests why the bug arose. The intentions of the program are used to determine what the program is supposed to do in order to effectively perform debugging. The intention is to use it as an instructional system which assigns programming assignments to students, reads over their work and gives helpful suggestions.

- 23. Gaslab** (Twigger, 1988 cited in Hartley, 1988) studies gas particle movements and collisions through modelling. The size of particles, pressure in container, volume of container, speed of particles can all be varied to model the behaviour of particles under various conditions. The walls of the container can also move to act as pistons.
- 24. GEO** (Duchastel, 1988) helps students learn Canadian Geography. The student has control and is able to explore geography according to individual interests. The program determines and teaches what the student does not know.
- 25. Stella** (Richmond et al cited in Hartley, 1988) uses quantitative modelling and has been used for teaching Newtonian dynamics, predator-prey relationships and the rock cycle in geology. The system builds a diagrammatic structure of the model, calculates values of variables as data is input and presents the value changes in the variables graphically or in tabular form. The independent functioning of the components of a model and their interdependence with each other are correctly demonstrated, providing qualitative science in teaching a topic.
- 26. Mendel** (Stewart et al, 1989) an intelligent computer tutor in transmission genetics provides a computer environment in which students can conduct realistic genetic experiments and receive appropriate advice. The system enables students to solve genetic problems by reasoning from causes (genotype information) to effects (phenotype information) as well as from effects to causes by solving problems created by a problem generator.
- 27. ACT and PUPs** (Anderson et al, 1990) (Penultimate Production System) developed for application in tutoring LISP, geometry and algebra, constructs performance models of how students execute skills to be tutored and learning models of how these skills are acquired. The performance model has a set of correct and incorrect production rules for performing the skill at hand and the learning model has a set of assumptions on how students' knowledge state changes after each step in a problem solving procedure.
- 28. Kangasauras** (Kinnear and Simmons, 1990) simulates transmission genetics and constructs a genetic model using a covert or an overt model of operation. In the covert mode the user has access to parents and offspring at the level of phenotypes only. In the overt mode the user is able to set the variables using both phenotypic and genotypic. By setting the variables the user can observe the operational outcomes of a genetic model.
- 29. Quest** (White and Frederiksen, 1990) is a circuit kit to help students solve problems in circuit building and fault finding by providing causal qualitative

explanations. It knows the functional properties of electrical components and retains structural knowledge of the circuits. Students make the circuits and Quest helps in guiding them and checking.

30. Mathematica (Wolfram, 1991) is a general software system for mathematical and other applications e.g. science, technology and business. It can be used as a numerical and symbolic calculator, as a high level programming language to create programs, for modelling and data analysis, and for representing knowledge in scientific and technical fields. Calculations and, symbolic and graphical computation is done for the user.

These 30 ICAI systems are now reviewed against the 13 learning theories and the learning environments of transmission and constructivism discussed in chapter 1, and the extent to which interactive discussion and problem solving are used (Table 3.3). The aim was to determine the learning theories which are explicitly or implicitly employed by the ICAI systems and the type of operating environment used. The 13 learning theories have been classified into 7 groups in accordance with the learning procedures implied by them (Table 1.1). Table 3.2 gives an example of each group and a brief summary of the way in which the learning theory is used.

Group (***)	Instructional System	Summary explanation of the learning theories used within each instructional system
A	Scholar (Geography)	Upon a correct response being provided by a student the system performs reinforcement by responding with positive feedback such as <i>very good</i>
B	Spirit (Probability Theory)	During tutoring the student is provided with information which is structured using an AND/OR tree. The dialogue and problem solution are performed using this hierarchical structure. Material is provided in graduated stages before exercises are given for solution.
C	Proust (Debugging Programs)	The system uses successive refinement of the programming steps of novices to lead them towards refining their program and hence their thinking. It is assumed that students taking programming are motivated to learn as it is self related. Therefore it is presumed that pinpointing bugs in novice programs will make them strive.

D	Geo (Geography)	It advocates the discovery approach of learning whereby the student inquires through statements or questions. The learner is seen as an active agent whose cognitive structures are undergoing change through interaction with the system. Sense is made of the subject material by being an active participant.
E	Mendel (Genetics)	It allows students to assume responsibility for posing their own problems and for designing and interpreting their own experiments. The only snag is that the initial data for experimentation is randomly generated by the system. The student then takes over and can produce as many genetic combinations as desired.
F	Stella (Geology)	It models the whole functioning of the system and gives the interconnectedness of causal parts and workings to provide a study of totality of the system. It provides a diagrammatic model depiction or graphical simulation of the system to enable learners to obtain a total picture and respond to whole patterns. The depictions and simulations are used to help in causing the reorganisation of the learner's constructs by clearing ambiguity.
G	Spade (Computer Programming)	It interacts with students by prompting and giving retrieval cues through hierarchically organised pull down menus providing integration of the encoded knowledge. Problem-solving techniques such as identification with previously solved problems, decomposition into simpler problems and reformulation of problem description help in relieving cognitive processing load and require reconstruction of information rather than mere recall.

*** See Table 2.1 for the grouping of the learning theories.

Table 3.2: Explanations highlighting the usage of the learning theories within the computer-assisted instructional systems.

In classifying the 30 systems into these 7 groups of learning theories the following points were noted.

- None of the systems make explicit mention of the learning theories used.
- Those which use the interactive discussion environment use the Socratic style of question and answer type of dialogue sessions.

- In practice a combination of learning environments are employed. The difference between settings is the degree to which each comes into play.

	Systems	Learning Theories (Groups are given in Table 2.1)							Operating Environments			
		A	B	C	D	E	F	G	T	ID	PS	C
1	Scholar	M	P						M	P		
2	Excheck		P	M					P	S	M	
3	Sophie			S	M		S			P	P	S
4	Isaac								P			
5	Newton		M								P	
6	Buggy					S					M	S
7	Bacon.5		M		M						P	
8	Mecho		M								P	
9	West			S	M				S	S		
10	Why		M		M				S	P		
11	Wumpus										M	
12	Able										P	
13	ACE					S					P	
14	Integrate		M			S			M		S	
15	Quadratic				S					S		
16	LMS		M						M		S	
17	Macsyma		P					S	P			
18	Spade							M		S	M	
19	Spirit		P					S	P	M		
20	DMS						S	S	P			
21	Guidon		M						S	M	P	
22	Proust			M					P		S	
23	Gaslab				P						M	
24	GEO				P	S			S	M		M
25	Stella						S	S	P			
26	Mendel				M	S			S	S	M	M
27	ACT/PUPs		M						M	S		
28	Kangasauras						S		P			
29	Quest		M		M				S	S	M	S
30	Mathematica							S	P			

T: Transmission ID: Interactive Discussion PS: Problem Solving
C: Constructivist
S: Used slightly M: Used to a medium level P: Used predominantly

Table 3.3: Learning theories and operating environments on which the ICAI systems are based.

From Table 3.3 it can be noted that the transmission and problem-solving environments are the most extensively used in many systems. The number of systems which are extremely transmission and problem-solving oriented are 9 and 7 respectively. Only 3 systems use interactive discussion in the main and there are none which predominantly espouse the constructivist environment. Of the 30 systems reviewed, 21 espouse graded skill development and discovery in stages

using hierarchical based theories to some extent or other (Table 1.1). This propagates an image of teaching and learning as highly stage structured and leading stepwise from one phase to another. This is understandably due to the state of the art of computer technology whereby everything is structured using production rules, frames, semantic nets or formal logic. As such, computers are to date only capable of handling information which has been pre-programmed in structured form. Furthermore, most of the systems use the transmission environment and operate using a problem-solving setting. These limit learners' reflection on their ideas as learners are given information and led stepwise to the goal or problem solution. Even then, much of the instructional material omits information that the student needs to know in order to perform the tasks and the student is left to figure them out by trial and error experimentation. An interactive learning environment is required which clarifies the ideas of students so that connections may be made.

Intelligent computer-assisted instructional environments have been used as teaching tools as well as to supplement student learning. They provide individualised sessions for the students. Therefore ICAI environments should encourage and be based as much as possible on the constructivist model. The core of our discussion in the ensuing section is to highlight the general program features required for a constructivist learning environment to be realised. The extent to which such a system may be realised is briefly discussed.

3.6 PROGRAM FEATURES FOR A CONSTRUCTIVIST ICAI SYSTEM

A student-initiated learning environment takes into account the constructivist model of learning whereby the learner is seen as an active participant in the process. The knowledge which learners bring with them to the learning situation and the active construction of meaning which goes on constantly while individuals interact with the environment are seen as important factors which need to be considered. Learning is seen as the way the mind organises reality through a combination of experience and information. In this form of learning environment, conceptual change is brought about by the student. The student makes decisions on learning (e.g. the sequence in which to learn the subject matter, and when to learn any particular topic). With these in mind, the essential features an ICAI system needs to have to function as a constructivist learning environment are:

1. A facility enabling the student to be an active participant in the learning process, thereby controlling the direction and pace of learning, preferably at all times.

2. The system must be like a knowledgeable teacher, aware of the pitfalls for students in their learning experience. It must have a working knowledge of the prior conceptions of students and the assumptions they tend to make in the learning and problem-solving process. This awareness and knowledge will enable the system to help the students.
3. Through interaction the system should be able to compile and determine the knowledge level and problem-solving awareness of the student. Research has shown that this is an important factor influencing learning. Some systems already use an overlay model (Goldstein, 1982) where the student's performance is compared to the expert's, or a buggy model (Brown & Burton, 1975) where the points at which the student's performance differs from the expert's are noted to obtain a student knowledge profile.
4. The system must enable the student to make learning decisions and handle alternative modes of operation dependent on student needs. The student should be able to choose the preferred style of learning e.g. by being given hints, by discussing his/her proposed solution to determine when help is required, by using analogy, or by the type of explanation required. The student might wish to use the system for various learning purposes e.g. simply to clarify ideas, to have a discussion on a particular aspect of the problem, to obtain help in solving a problem, to test whether an analogy is applicable, or to be assisted to make a graphical or diagrammatic simulation. Feedback given must be designed such that it forces the student to think.
5. There should be provision for the student to apply the ideas developed for consolidation and reinforcement. The system needs to provide a range of models, experiences, demonstrations, worked examples and analogies to test out acquired ideas or concepts.
6. The system must be able to challenge and discuss ideas which will help the student to deal with contradictory previous knowledge. This will help students to discard, modify and/or integrate prior knowledge with new situations. Merely explaining away an inadequate conception is not sufficient. To do this the system has to be supportive, cooperative, committed and interested in the student's knowledge. For the system to be effective it must seem to be encouraging. It should try to ensure that students who genuinely make an effort meet with some form of success.

7. While interacting the system should be able to make explicit the links to other topics within the subject as well as links to knowledge within the particular topic. Students are rarely made aware of these connections within the subject domain. Knowing such connections will enable the students to develop connections between their schemata and in turn enhance the possibility of coalescing them into larger functional schema units.

To realise a system propagating a constructivist learning environment we need to look at the mechanics of such a system within an actual classroom situation. The students work on a task that they are self motivated with and whenever they encounter problem areas they would seek more information. To perform the task the students need a certain basic working knowledge of subject matter, and avenues for seeking further information. Furthermore, they have options of consulting someone better versed in the domain to discuss ideas, or to discuss amongst themselves. Students can choose how to learn. They can ask the teacher to give an example, provide an analogy, or simply provide diagrammatic or graphical depiction to clarify the problem.

One problem with this ideal situation is to recognise when student conceptions are not in line with the principles governing the structure of the domain. Once the conceptions are known, the system can be used to challenge the student with counter examples or alternative applications and explanations. Initially therefore, the system could be programmed with alternative conceptions known to commonly exist amongst students. Over time, it can be used to collect data on alternative conceptions. Not only will this make the system dynamically evolving but will also bring it closer in similarity to the performance of a live teacher working within a student-initiated environment. A live teacher takes years to accumulate a repertoire of skills and information to handle students' conceptions and errors. The teacher develops a variety of techniques to deal with the common conceptions and errors. There are situations in which the teacher is astounded or baffled. For these reasons, we should not expect the system to deal immediately with all errors and conceptions.

It is impractical to expect a system to be able to solve the difficulties outlined above. A good teacher provides information and suggests avenues for students to move towards solutions to tasks. An ICAI system should provide activities with the same intentions. For a system to function in the manner described above dialogue is essential. To overcome this, pull down menus may be used to access key and common words necessary for communication and interaction between the student and the system. Words may be individually chosen through the menu to form sentences. The menu can also provide choice of algebraic constants, combination of

numbers, functions and mathematical operators for representing quantities and for forming equations. Necessary feedback could come through the system rephrasing or reiterating for confirmation the students' input. This is one area with which the state of the art of computer technology is still grappling. However this difficulty may be less extreme when working with limited dialogue within a restricted subject domain.

3.7 CONCLUSION

The development of computer-assisted instruction has been traced from traditional computer-based systems to the usage of artificial intelligence techniques used to develop expert systems resulting in ICAI systems. The differences between CBI and ICAI systems have been highlighted. The main components constituting the expertise module, the student model, the tutoring module and the interface have been explained to provide an understanding of ICAI systems. The representation of knowledge and the search and control techniques used to access knowledge and make inferences have been described to provide insight into the general workings of an ICAI system. Thirty systems have been reviewed to determine the learning theories and the type of operating environment used by each. The program features necessary to realise a constructivist based problem-solving environment are mentioned. The chapter concludes with a discussion of the extent to which these features may be realised.

In the next chapter, the research design and methodology used in this thesis are detailed. The different research phases providing a chronological order of the research performed and data collection and analysis are detailed.

CHAPTER 4

RESEARCH DESIGN AND METHODOLOGY

4.1 INTRODUCTION

This chapter describes the research design of the study, and the methodology used for data collection and analysis. The aim of the data collection was to inquire into the problem-solving process, to gather information for program development and for program evaluation. Each phase of research is elaborated with a discussion of:

- the activities performed and the rationale.
- the data collection.
- the analysis.

The research design consists of four phases

Phase 1: Initial data collection and analysis.

Phase 2: Problem selection and associated data collection.

Phase 3: Program development.

Phase 4: Program use and evaluation.

These four phases are detailed in section 4.3.

A variety of methods were used to ensure reliability, validity and richness of data collected. The methods in the order in which they were initially used are:

Think aloud sessions to capture the train of thought and problem-solving style of participants.

Pen and paper sessions to explore the information accessed and used to solve a problem.

Observations to record knowledge which might not be verbalised and to document the interaction with the program.

Interviews to seek clarification of data collected.

Written feedback to obtain information on difficulties encountered and use of information for assistance.

Questionnaires to obtain feedback on use of the program.

Program compiled interaction records to determine the interaction with the program.

In the next section, these methods are critically reviewed in the order in which they were first used. The pen and paper, and written feedback methods are discussed as written accounts of data. Program compiled records of interaction are included in observations, as they serve a similar function to collecting data through observing participant use of the program. For each, a brief introductory explanation is given. Then the criticisms are mentioned followed by the context in which it has been used in this study. An overview of the methods is given in Figure 4.1.

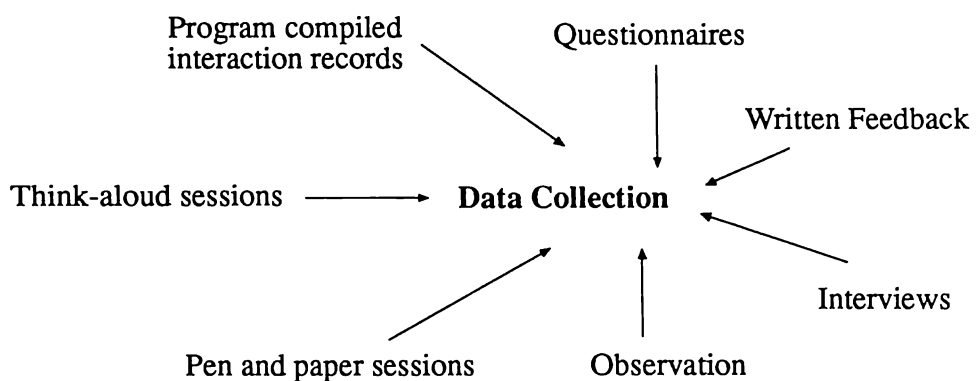


Figure 4.1: The methodology used for data collection.

4.2 REVIEW OF DATA COLLECTION METHODS

Think-aloud: Ericsson and Simon (1985) represent verbal reporting as bringing information into attention, then when necessary, converting it into verbalised code, and finally, vocalising it. The crucial issue is what information is heeded. They describe think-aloud as the process whereby successive states of heeded information is directly verbalised. The verbalisations are usually audio or video taped and transcribed to obtain the think-aloud transcripts.

Sokolov (1972), Krutetskii (1976), Larkin and Rainard (1984), and Glasson and Lalik (1993) are some of the many researchers who have successfully used think-aloud transcripts to uncover knowledge used to solve specific problems and to document the

thoughts of the solver. Think aloud transcripts do not tell why the solver performs the actions. It provides a regular indication of what the solver is thinking (Larkin and Rainard, 1984). The verbalisations are the content of working memory (Newell and Simon, 1972). The strongest criticism of verbal transcripts has been whether these can be used to represent and infer cognitive processes (Ericsson and Simon, 1985). Another criticism is that verbalising might affect the task performance. Thinking aloud is not an activity normally performed by the solvers. Solvers therefore find it difficult and at times awkward and have to be regularly reminded to continuously verbalise.

In this study, verbalisations have been used as a means of recording the problem-solving behaviour. The transcripts are not considered a complete record of the solver's thoughts, and no inferences to cognitive processes are made. The transcripts have been used to capture the train of thought of the solver, and the problem solving knowledge and style used. To minimise the effect on task performance, simple warm-up questions have been used to initiate the solver into the process.

Written accounts: Cohen and Manion (1989) describe accounts as a way of speaking and sharing our experiences from a behavioural point of view. The view taken here is that accounts may be written or verbal. Through both these modes experiences can be reported and shared. Written accounts here includes pen and paper sessions and written feedback. The pen and paper data collection method requires participants to produce a written account of their experience at performing a task e.g. reporting the solution to a problem. Written feedback here is the process whereby participants report information pertaining to their experience e.g. describing difficulties encountered in performing the task. Unlike verbal discourse, the person is under no pressure to produce immediate answers. It allows time to reason and weigh alternatives. Writing it is argued, has the advantage of stimulating thought (Lofland, 1971).

The criticism of written accounts has been the difficulty of conveying meaning. The researcher has to infer the meaning the person reporting is trying to convey. This is a particular problem for subjective data collection where opinions are sought. In the case of problem solving in a well structured domain such as physics, the solver is reporting specific data and inter-related information. This makes it more possible to trace the successive steps leading to the solution and to explain what the solver is doing. Chi et al (1981), Champagne et al (1985), Gunstone (1984), Heller and Reif (1984), Mestre and Touger (1989), Kinnear and Simmons (1990) and Grimellini-Tomasini et al (1993) are some of the many researchers who have successfully used written accounts to inquire into students understanding and to obtain insight into the problem-solving behaviour of experts and novices.

In this study, written accounts (pen and paper and written feedback) have been used to supplement the think-aloud data collection. Both methods were used to obtain the written problem solution, the information accessed and referred to during the problem solution, and information on the difficulties encountered. Written feedback was also used to obtain retrospective information on the way difficulties encountered were resolved.

Observation: The purpose of observation is to probe deeply and to analyse intensively (Cohen and Manion, 1989). Observation could be of an event or an episode involving one or more persons. It could be of the behaviour of the group or individual, of the task being performed or of both. This can be done by video taping or using pen and paper to document the information.

Observation studies are considered superior when data is being collected on non verbal behaviour. The observer is able to record behaviour and activities as they happen and is able to make appropriate notes (Bailey, 1978). The criticism of observations is that they are subjective, biased, impressionistic and lacking in preciseness (Stake, 1978). This is true when physical behaviour is observed or when the observer has to make inferences from the observations. Observation is a good approach when the behaviour being recorded is obvious. Then the salient features as well as related information can be documented. Adams and Biddle (1970), Brown (1992), and Fischer and Aufschnaiter (1993) are some researchers who have used observations to study classroom teaching and student problem solving behaviour.

Observations were used to document knowledge that might not have been reported by the verbalisations during the think aloud sessions or during program use e.g. checking the text and recording the features of the program used by the solver. This type of observation does not lead to inferencing behaviour and does not suffer from the drawbacks mentioned. The program was also used to compile the interaction of the user to determine how effective it is in documenting the features used by the students.

Interviews: Cannell and Kahn (1968) define a research interview as a conversation between two people initiated and focused by the interviewer. It is conducted to obtain a description, clarification, prediction or explanation. White and Gunstone (1992) describe an interview as being used to bring about an understanding of student thinking. Interviews can be of an instance, an event or a concept to probe the understanding of students.

Interviews provide opportunities for asking and probing to obtain clarification (Tuckman, 1972). Carr (1991) stresses that interviews are necessary to probe ideas

fully. The criticism of interviews is that respondents might attempt to anticipate and verbalise what the interviewer wishes to hear. Butts and Smith (1987) summarise the tension created by an interview, interruptions created by the interviewer, the context of the interview and the omission of some questions as factors affecting the data collected through interviews. Merton and Kendall (1946) use the term focused interview where the respondent's responses from a prior situation have been analysed and are used for further probing. Carr (1991) mentions several references of researchers who have successfully used interviews to probe the understanding of students.

In this study, focused interviews were used to seek clarification of written and verbal feedback of respondents. The tension factor was not present as the interviewees were discussing feedback provided by them and not answering further questions for information. The respondents did not have to anticipate the answers as only clarifications of their own work was discussed. The clarifications were sought to obtain insight into the data provided by the interviewee.

Questionnaires: A questionnaire can be described as a single or set of printed forms with a series of questions used for gathering information. The respondent is required to write responses to the set questions.

Questionnaires provide opportunity for probing and asking in a limited way (Cohen and Manion, 1989). They focus the thoughts of the respondent. The anonymous nature of questionnaires encourages honesty and increases reliability. The disadvantage is that questions and the answers provided by the respondents may convey different meanings to different people (Tuckman, 1972). For a good response, questionnaires must have simplicity of design, clarity of wording and content arrangement to encourage co-operation (Hoinville and Jowell, 1978). Cohen and Manion (1989) provide several references of studies which have used questionnaires to study classroom behaviour. Grimellini-Tomasini et al (1993) is a recent study which uses questionnaires to probe the understanding of students in the conservation laws of Mechanics.

In this study, questionnaires were used to obtain information for program evaluation. The problem of question clarity has been overcome by explaining and clarifying the questions to the respondents. They were able to seek clarification while they completed the questionnaire. This meant that the anonymity of response was lost. This was overcome by encouraging respondents to be truthful and open in their feedback and criticism. It was stressed that they should be critical when answering the questionnaire. The questionnaire included items requesting feedback on positive as well as negative aspects of the program.

In the remainder of this chapter the four research phases are discussed in the order they are detailed in the research schedule given below.

4.3 RESEARCH SCHEDULE

Phase 1: Initial data collection and analysis

Choice of topic, questions and target group.
Think aloud, and pen and paper problem-solving
data collection using experts and novices.
Observation.
Interviews.
Analysis of transcripts.

Phase 2: Problem selection and associated data collection

Problem selection for further data collection.
Pen and paper data collection and written feedback
by
(i) novices; (ii) an expert; and (iii) teachers.
Interviews.
The general and alternative problem solution.

Phase 3: Program development

Experimenting with the Hyper-X expert system
shell and using Hypertalk for program
development.
Learning C, the Think C Class Library and the
Macintosh programming structure.
Developing the program using the Think C Class
Library.
Program trialing and modification.
Written feedback.
Interviews.

Phase 4: Program use and evaluation

Problem solution using the program.
Solution of a related problem.
Program compiled interaction records.
Questionnaire administration.
Observation.
Interviews.

4.4 PHASE 1: INITIAL DATA COLLECTION AND ANALYSIS

4.4:1 Target group and topic

The decision to research and develop an ICAI system and the choice of rotational dynamics as the topic of research were discussed in chapter one. This section describes the choice of target group.

To take the prior knowledge of students into account an age group with prior exposure to Physics teaching and learning was needed. Year one undergraduates enrolling for a first year Physics course were chosen. They would generally have undergone one to two years of high school Physics teaching and learning. In addition to this they would have been taught Physics, as a component of Science, up to the fifth form level. Rotational dynamics is taught in detail only at the first year undergraduate level. These students will have been taught particle physics and their undergraduate study includes rotational dynamics. A further important prior experience relates to computer use. Whereas New Zealand high schools have insufficient computers to implement an ICAI system, university computer facilities are readily available and Physics students are encouraged to interact with them. A program developed for first year Physics students could be installed on computers in the Physics laboratory for general consultation.

The choice of problem is discussed in the following section.

4.4:2 Choice of questions

Problem selection was made by reviewing the rotational dynamics problems in several texts used by first year university Physics. The questions were chosen on the basis that a successful solution required:

- a number of theorems, principles, constraints and assumptions,
- an understanding and application of a combination of higher order principles (such as the conservation of energy, and conservation of angular momentum),

- a general understanding of forces and the writing and application of force equations,
- an understanding of both translational and rotational motion.

Six questions, two requiring a qualitative and four requiring a quantitative solution, were chosen for the initial problem-solving data collection. They were:

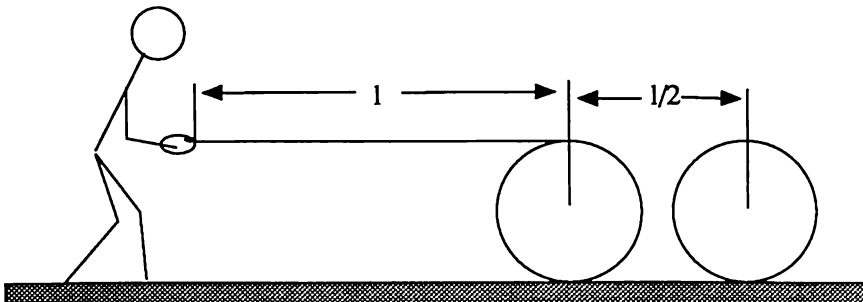
Qualitative questions

1. Older model helicopters have two propellers; some have both propellers on vertical axes but rotating in opposite directions, and others have one on a vertical axis and one on a horizontal axis perpendicular to the helicopter body at the tail. Why is a single propeller never used?

Beiser, A. (1962)

2. A cylindrical drum, pushed along by a board from an initial position shown in the figure rolls forward on the ground a distance $l/2$, equal to half the length of the board. There is no slipping at any contact. Where is the board then? How far has the man walked?

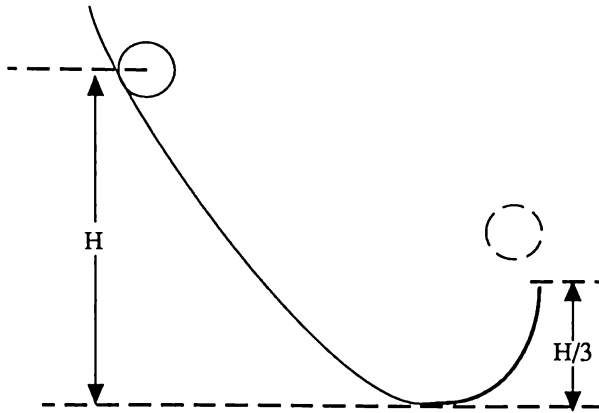
Resnick, R. and Halliday, D. (1988)



Quantitative questions

3. A ball is released from rest from a height H and rolls along the curved track without sliding as shown in the figure below. The ball leaves the track vertically at a height $H/3$. To what maximum height does the ball rise?

Weidner, R. T. and Sells, R. L. (1966)

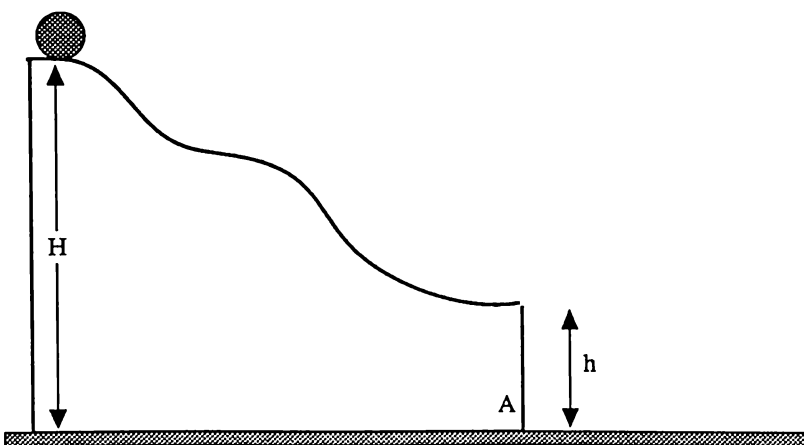


4. Two wheels are mounted side by side on the same axle. Wheel A whose moment of inertia is 5 kg m^2 , is set spinning at 600 rev/min . Wheel B, with a moment of inertia of 20 kg m^2 , is stationary. A clutch now acts to join A and B so that they must spin together. (a) At what speed will they rotate ? (b) Suppose the clutch acts gradually. Will the end result be the same as though they were joined suddenly ? (Assume the bearings are frictionless). (c) How does the rotational kinetic energy before joining compare with the kinetic energy afterward ? (d) What torque will the clutch have had to transmit if A makes 10 revolutions relative to B during the operation of the clutch ?

Gamow, G. and Cleveland, J. M. (1976)

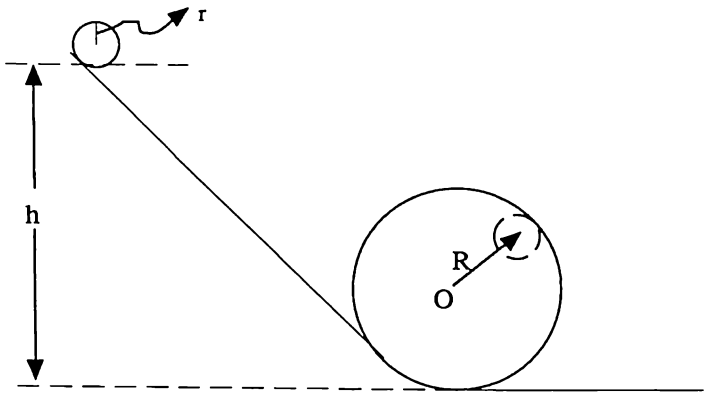
5. A homogeneous sphere starts from rest at the upper end of the track shown in the figure and rolls without slipping until it rolls off the right-hand end. If $H = 200\text{m}$ and $h = 6.0\text{m}$ and the track is horizontal at the right hand end, determine the distance to the right of point A at which the ball strikes the horizontal base line.

Resnick, R. and Halliday, D. (1988)



6. A uniform ball of radius r rolls without slipping along the loop-the-loop track shown in the figure. It starts at rest at height h above the bottom of the loop. If the ball is not to leave the track at the top of the loop, what is the least value h can have (in terms of the radius R of the loop) ? What would h have been if the ball were to slide along a frictionless track instead of rolling ?

Tipler, P. A. (1991)



Modifications to questions

Question 1 was modified replacing *All* with *Older model* since some newer model helicopters work on a single propeller. Units were converted from centimetre and gramme to metre and kilogram in question 4, and from foot to metres in question 5.

4.4:3 Think-aloud sessions

	University Lecturers	High School Teachers	D. Phil Students	Masters Students	Under- graduates
Think-aloud	1	0	2	4	6
Pen & paper	1	5	0	0	7

Table 4.1: Participants in the initial data collection.

All the participants in the study volunteered to help in the data collection. Table 4.1 gives the list of participants. Apart from 3 think-aloud participants (1 master’s student and 2 first year undergraduates) and 1 high school teacher from the pen and paper participants, all were males. Thirteen participants (Table 4.1) performed think-aloud problem-solving for all of the six problems. The lecturer, the two D.Phil students and a master’s student were considered experts. The lecturer had taught Mechanics at first year level for over twenty years, and the two D.Phil. students and the master’s student tutored the first year Physics course. The other three master’s students and the six first year Physics students were considered novices.

At each session the participants drew lots to determine the problem they would attempt. One problem was attempted each time. Sessions for each participant were separated by approximately a week. At each session, the text Physics (volume 1) by P. A. Tipler was available for reference. Prior to each session, the instructions outlined in Appendix A1 were read to the participants. The following were stressed:

- The aim was to collect data on how they went about solving the problem, to obtain information used by them to solve the problem and to capture their train of thought as they attempted the problem.
- They were required to verbalise every thought and action.
- The exercise was not a test of their ability to solve the problem.
- They were expected to attempt the question on their own.
- Help not pertaining to the solution of the problem could be obtained.
- They could take as long as they wished.
- Reference to the text provided could be made at any time during the session.
- They would be reminded to verbalise if they stopped or their voices faded.

1. Two plane mirrors make an angle of 90° . Show that there are three images for any position of the object. Draw appropriate bundles of rays from the object to the eye for viewing each image.
2. A thin converging lens of focal length 10cm is used to obtain an image which is twice as large as a small object. (a) Where should the object be placed ? (b) Where is the image ? (c) Is the image virtual or real ? Erect or inverted ?
3. A car travelling at 100 km/h crashes into an immovable concrete wall. How long does it take to bring the car to rest and what is its acceleration ?
4. A single force of 10N acts on a particle of mass m . The particle starts from rest and travels in a straight line a distance of 18m in 6s. Find its mass.

Figure 4.2: Warm-up questions used.

For the first four sessions, prior to solving the question, each candidate first attempted a different warm up question (Figure 4.2). This was done to get them used to thinking aloud and verbalising every thought and action during the problem-solving session.

During the think-aloud sessions, participants verbalised their train of thought as they solved a problem. The verbalisations were audio-taped. The researcher also observed and took notes to document actions which were performed but not verbalised. Participants were politely reminded to keep verbalising if they stopped or if their voice seem to fade off. The verbalisations were transcribed to obtain the problem-solving transcripts. Participants were individually interviewed for clarification of their transcripts. These were analysed and used to make a choice of one question for further research and program development.

The criteria for selection is outlined in the next section. The analysis of the transcripts is reported in the next chapter.

4.5 PHASE 2: PROBLEM SELECTION AND ASSOCIATED DATA COLLECTION

4.5:1 Problem selection

Analysis of the think-aloud transcripts showed that the qualitative questions were unsuitable choices either because the students could not start them; or because they could solve them easily. In the latter case, there were no intermediate stages or steps which participants found difficult and the solution did not require much processing of information by them. Due to the qualitative nature of the problem and answer, conducting a dialogue, even to a limited extent, would be problematic due to the lack of computer flexibility in parsing and recognising strings of words.

Of the four quantitative questions, numbers 3 and 5 were successfully solved by practically all the 13 participants. Since a question was needed with a level of difficulty requiring students to consult the program developed, these problems were not selected.

Questions 4 and 6 were successfully solved by 1 and 3 of the experts respectively, and by none of the novices. Both questions required the application of different principles, knowledge of constraints and assumptions for a successful solution. In question 4, the clutch and axle action requiring an understanding of torques proved too complex for novice as well as expert understanding. Solution to question 4 requires an understanding of the work energy principle which students generally have difficulty visualising in comparison to simple energy conversions involved in the motion of the ball in question 6. A further difficulty was that there was limited possibility of varying the quantities involved in this question. Question 4 was therefore not selected.

Question 6 was selected for further research and data collection, and program development. It involved the interplay of gravitational potential energy, translational

kinetic energy and rotational kinetic energy. The starting conditions of the ball, the type and size of the track and ball, the angle of slope of the incline and the acceleration due to gravity could be easily varied. Furthermore, calculations can be performed for various positions of the ball. The motion of the ball can be easily simulated to provide further insight for the students.

During the think-aloud data collection, the queries made of the question wordings and certain ambiguities which arose were used to modify the question. The changes were:

1. making the question clear and easy to comprehend; and
2. simplifying the mathematical considerations required for its solution.

A solid uniform ball of radius r slides down a ramp into a loop-the-loop track. It starts from rest at a height h above the lowest position of its centre of mass (as shown in the figure). (a) If the ball is not to leave the track at the top of the loop, what is the least value h can have in terms of R ? (b) If the ball were to roll instead of slipping what is the minimum value of h ?

Note: R is the radius of the path of the centre of mass of the ball.

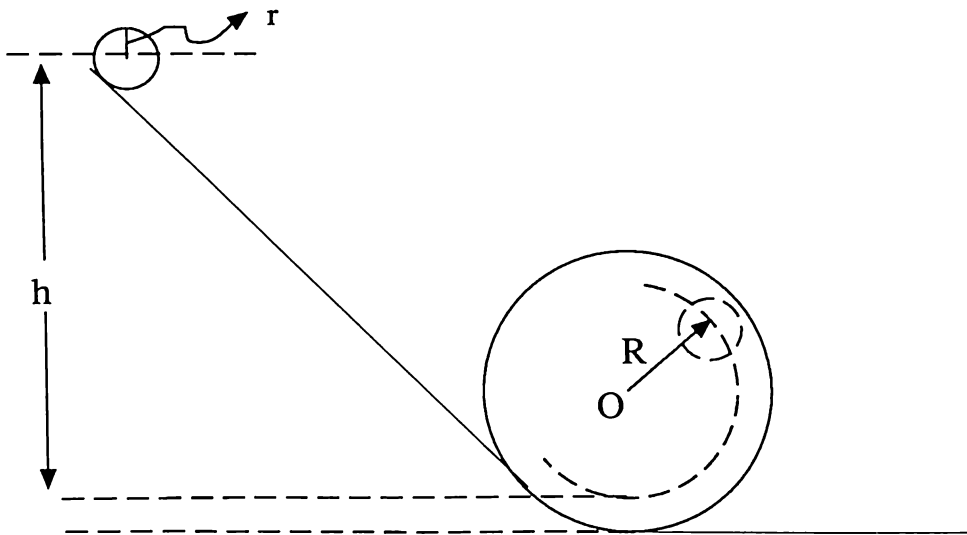


Figure 4.3: The modified version of question 6.

The information and supporting evidence leading to the modification of the question is mentioned in the next chapter where the analysis of the transcripts is detailed. The modified version of the question is given in Figure 4.3.

4.5:2 Pen and paper data collection and written feedback.

One university Physics lecturer, 5 high school Physics teachers and 7 first year Physics undergraduates (Table 4.1) were used to obtain further information about the modified question. The university lecturer had also participated in the think-aloud data collection but all the others had not. Four of the teachers had 5 to 10 years of experience teaching Physics to forms six and seven students. One was a beginning teacher. This teacher and the 7 undergraduates were considered novices for this phase of the data collection.

4.5:2a Novices

Each novice solved the modified form of the selected question in Figure 4.3. Each was given the question together with a set of instructions (Appendix A2). They were informed that the exercise was not a test of their ability to solve the problem. The aim was to collect data on:

- how the problem may be solved;
- the information needed to solve the problem;
- how they went about solving the problem;
- the examples they used or referred to solve the problem;
- the steps they took to obtain the information for solving the problem; and
- the steps leading to the solution.

They were requested to provide written feedback by detailing the difficulties encountered. If they were able to resolve the difficulty, they were requested to describe how they overcame it. If the difficulty hindered progress in the problem solution, they were requested to elaborate why they were unable to resolve it.

The participants took the question home to solve in their own time. Every participant handed in the question together with the requested information within a week of receiving the question. Upon analysing their transcripts, the eight participants were interviewed and their feedback clarified and discussed. The pen and paper and written feedback were performed to:

- determine any other ambiguities which might exist in the question wording.
- collect data on student presentation of problem solution.

- obtain information on how students generally went about solving an assigned problem.
- further document the difficulties encountered by the students while attempting to solve the problem.
- identify how students attempt to resolve and overcome difficulties encountered while problem solving.
- determine the kind of cues and hints which helped them to solve the problem.
- document the information they referenced while attempting to solve the problem.
- identify the problem areas encountered and the points during problem solving at which help and information needs to be supplied.

4.5:2b Expert

The lecturer was requested to solve the problem as would an expert well versed in the solution of first year rotational dynamics problems. The aim was to document the information, assumptions and steps used by experts in solving the problem.

4.5:2c Teachers

The four high school teachers were requested to solve the problem as they would teach it to students who had difficulties with the solution. The instructions given to them are given in Appendix A3. They were requested to distinguish the information they would present on the board while solving the problem from that which they would verbalise. The aim was to obtain the information and steps generally presented to explain the problem solution and address the difficulties encountered by students. While documenting the solution, the teachers were requested to assume that:

- their students had done the topic(s) required to solve the problem on rotational dynamics.
- the problem had been assigned for homework and a majority of their students were unable to solve it.
- the students have requested that the solution be performed on the board during one of their Physics lessons.

They were requested to:

- include all the information and steps which they would present to the students to enable them to comprehend the solution.
- include all the verbalisations they would use in explaining the solution steps while performing the solution.
- distinguish between the information which they would write on the board and that which they would present verbally.

4.5:3 The general and alternative problem solution

As a final stage of data collection on the problem solution, a comprehensive solution of the problem was compiled assuming only the fundamental concepts of displacement, time and the associated units. This made explicit all the knowledge required and every assumption made in solving the problem. This ensured that relevant information not uncovered by the think-aloud and pen and paper sessions was identified. The process highlighted the considerable related information required to solve the problem beginning with the fundamental building blocks of Physics, implied as known when the problem solving is performed or explained to students.

Through this, an alternative solution path not performed by any of the participants was uncovered. The expert solution, the teaching solution, the general solution and the alternative solution provide the necessary information and help required to solve the problem. All these solutions are detailed in chapter 6.

4.6 PHASE 3: THE PROGRAM DEVELOPMENT

4.6:1 The Hyper-X shell

At the University of Waikato, Macintosh computers are widely used in all departments. Therefore, it was decided to develop the program on a Macintosh. The initial decision was to use an existing expert system shell, its structure and supportive environment to develop a system. Most of the expert system shells proved to be costly, and could not be used on the Macintosh since they had been developed for IBM or IBM compatible type personal computers. This left the Hyper-X expert system shell which uses the Hypercard environment which is easily available on practically all Macintoshes. Furthermore, Hyper-X allows access into its background for modification and alteration of the supporting program documentation.

The Hypercard environment works on fields and buttons. Each button can be programmed to handle a command using Hypertalk, and information can be entered using fields. Hypertalk is the programming language supported by Hypercard. The initial step required learning Hypertalk so that modifications and additions could be made to the shell. The next step was to program the system for a pilot task related to determining the equilibrium state of a body. This required testing whether a body was in stable, unstable or neutral equilibrium.

As the pilot task proceeded it was realised that although the program documentation could be changed, the functioning of the shell could not be much altered. This inflexibility meant that features which would have been incorporated could not be included. Information could only be included as rules since the shell worked on a rule-based system. This proved to be rigid in structure since as more information was added the number of rules increased. This made the system slow as a result of the forward and backward chaining of the system going through all the related rules. A major change to the shell required writing and importing external functions and knowledge of a programming language. These difficulties led to the decision to learn a programming language and develop a basic interactive program which could be expanded for future use.

4.6:2 The programming

The Macintosh can be programmed using two languages, C or Pascal. C is a general purpose programming language originally designed for the UNIX operating system. This was chosen since the Physics department uses it for research work.

Program development required both learning the C language and becoming familiar with the Macintosh programming structure. The decision was taken to use the Think C Class Library, which provides the necessary background interfacing and support. This required learning and becoming familiar myself with another programming language with its own structure and commands, using object-oriented programming techniques. This complex structure then had to be applied to and placed within the Macintosh system.

4.6:3 Program

Program development was an extremely time consuming and complex process. The program was developed so that a stepwise record of the user's consultation was compiled. The program and its structure are described in chapter 7. Upon completion of the first draft the program was trialed by 2 D. Phil. students, a master's student, and 3 year one undergraduates in the Physics department. Participants were given the

instructions outlined in Appendix A4, asked to critically use the program, and to note aspects of the program they felt:

- needed correction;
- hindered their consultation;
- needed modification together with any recommended changes; and
- were helpful during program consultation.

After this trial participants were interviewed about their feedback. Modifications made are detailed in Chapter 7. The revised program was trialed by the same volunteers another time and further modifications are detailed in Chapter 7.

4.7 PHASE 4: PROGRAM USE AND EVALUATION

This was carried out in two stages using 18 undergraduates reading the first year Physics course. The volunteers had undergone the same instructional lectures on rotational dynamics about a month before the first trial. In stage one, each participant solved the problem individually using the program. The aim was to determine:

- the assistance students seek in interacting with a computer-assisted system.
- the level to which the system allows students to control the interactions.
- whether students find it encouraging and worthwhile to consult with a computer-assisted system.
- the features of the system deemed good by the students.
- the system limitations encountered by the students.

In stage two, only 14 of the 18 participants performed the data collection. The other 4 withdrew from the study. Each participant solved a related problem, with the option of consulting the program for help. The aim was to determine whether the computer assisted system aided the transfer of learning to the solution of a related problem.

The aims detailed above (for stages 1 and 2) are coincident with the research questions detailed in chapter 1. Details of the design and methodology of these two stages follow.

4.7:1 Stage 1

Each participant was read the instructions in Appendix A5, given the question and asked to first attempt a solution using pen and paper. This was intended to help them determine aspects of the problem requiring assistance. Participants could spend as much time as they wished attempting to solve the problem prior to using the program. When they wanted to consult the program, they were shown how to get started and the main program features were explained.

Participants could browse through the program as long as they wished before the problem-solving interaction. They were encouraged to go through each feature in turn and familiarise themselves with the program so that their interaction was not hindered when they commenced consultation.

During program use:

- each person was observed solving the problem; and
- a stepwise record of interactions was compiled by the computer and by the researcher.

Upon completion of the program:

- each person was administered the questionnaire in Appendix A6.
- individual interviews on written feedback and questionnaire answers were carried out.

Table 4.2 relates the feedback sought by the questionnaire to the research questions detailed in Chapter 1. Some of the feedback items listed below are related to more than one research question.

Feedback sought on:	Research question
hints given by the program. simulations provided. diagrams provided. qualitative envisioning provided. aspects for which the program would be consulted.	What assistance do students seek in interacting with an ICAI system?
amount of information available. flexibility of program to individual needs. control the students had during interaction.	To what extent does the system allow students to control the interactions?
help provided by the program. ease of program use. level to which an understanding of rotational dynamics was enhanced. accessibility to information. level to which students learnt how to solve the problem. features of the program which were liked.	Do students find it encouraging and worthwhile to consult with such a computer-assisted system?
guidance provided by the program. extent to which the program would be used.	Which features of the program are deemed good by the students?
program limitations encountered. difficulties encountered during interaction.	What system limitations do students encounter?
other related views and information.	

Table 4.2: The information sought by the questionnaire in stage 1 related to the research questions.

4.7:2 Stage 2

For stage 2, participants were given a related question (Figure 4.4) to solve using pen and paper, with the option of consulting the program. This stage commenced three weeks after the completion of stage 1 and ended four weeks later.

A marble rolls from rest without sliding from the top of a large sphere, which is held fixed. Find the angle from the top of the sphere of the point where the marble breaks contact with the sphere. The radius of the marble is 0.01m and that of the sphere is 0.8m. What would the angle be if the marble were to slide on a frictionless sphere instead of rolling?

Modified version of question from Tipler, P. A. (1991)

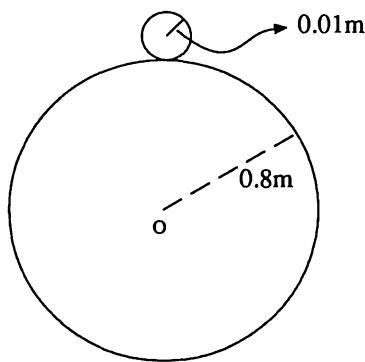


Figure 4.4: The similar question administered.

The participants had been informed prior to their participation in the study that they would be required to solve a second problem. The question was chosen for its similarity to the problem initially solved by the students (Figure 4.2). Each session was individually conducted. Before solving the problem the participants were read the set of instructions given in Appendix A7. They were informed that (i) the program did not solve this problem but could be used to obtain information and hints; (ii) they did not have to use the program to solve the problem; (iii) they could ask the researcher if they were unsure on how to access information in the program; and (iv) in attempting to solve the problem they could consult the program as many times as they wished.

Stage 2 aimed to determine:

- whether the program was able to help the students to solve a related problem;
- whether students would consult a program when they encounter difficulties;
- which features of the program students consult; and

- whether students can transfer their learning to the solution of a related problem.

Each student completed another questionnaire (Appendix A7) and was interviewed, after they had attempted the problem. The questionnaire covered the following aspects:

- The difficulties encountered while attempting the problem.
- The extent to which the program helped resolve the difficulties.
- The extent to which the program aided in the solution of the related problem.
- The similarities noticed by the participants between the two questions.

The information collected in stage 2 is used to answer the following research questions detailed in chapter 1:

- What assistance do students seek in interacting with a computer-assisted system ?
- Do students find it encouraging and worthwhile to consult with such a computer-assisted system ?
- Can students transfer their learning to the solution of related problems ?

4.8 SUMMARY

The research design and methodology used to collect data have been detailed and the research schedule outlined. Think-aloud transcripts, observations, interviews, pen and paper sessions and written feedback used to collect the initial problem-solving data and to help make a choice of question have been described. The program development has been traced from experimentation with the Hyper-X shell through to learning the C programming language and the Macintosh programming structure leading to the use of the Think C Class Library. The trial and use of the program to collect data for evaluation has been detailed. The methodology used to collect the data for evaluation has been specified.

In the next chapter the initial data collection transcripts obtained using the methodology described in this chapter are discussed and analysed.

CHAPTER 5

PROBLEM-SOLVING PROTOCOLS ANALYSIS

5.1 INTRODUCTION

This chapter analyses data from phases 1 and 2 of the research (section 4.3). A brief discussion of the think-aloud transcripts of the participants for the 6 questions (Chapter 4) is used to provide evidence for the choice of the selected problem. Then follows the analysis of the problem-solving of the participants highlighting:

- points at which the problem solving commenced.
- question clarifications and assistance sought.
- examples referred to for help.
- general mistakes.
- conceptions portrayed.
- major difficulties encountered.
- knowledge used.

The information obtained is used to determine the knowledge and explanations required to form the knowledge base of the problem-solving program which is described in Chapter 7.

The analysis leads to a description of the variety of problem-solving processes used by the participants, resulting in 14 categories of activities. Each is discussed using excerpts from the transcripts. The fourteen activities are used to form flow charts to represent the problem solving processes of the participants. Expert and novice strategies are compared. These are related to the findings on expert and novice problem-solving reported by other researchers.

Finally the analysis is related to the framework on learning theories, learning environments and problem solving detailed in Chapter 2. The problem solving processes are used to provide insight into:

- general and specific problem-solving strategies used.

- problem-solving perceptions and expectations.
- learning theories and environments implied by the problem solutions.

5.2 DATA ANALYSIS LEADING TO THE PROBLEM SELECTION

In this section the data analysis is used to justify the selection of question 6.

As stated in Chapter 4, questions 3 and 5 were considered too easy for program development as 9 out of 13 participants successfully solved them. Table 5.1 provides an overview of the success of each participant at solving the 6 questions.

Participant	Question					
	1	2	3	4	5	6
E1	**	**	**	**	**	*
E2	*	**	**	*	**	**
E3	**	*	**	*	**	**
E4	**	**	**	*	**	**
N1	**	**	**	•	**	•
N2	**	**	•	•	•	•
N3	*	**	**	•	**	**
N4	•	•	**	*	**	•
N5	•	**	**	•	*	*
N6	*	•	*	*	**	•
N7	•	•	•	•	•	•
N8	*	•	•	•	•	•
N9	**	**	**	*	**	•

Experts E1-E4

Novices N1-N9

- ** Successfully solved the problem
- * Partially solved the problem
- Unsuccessful attempt.

Table 5.1: Success of participants at solving the 6 problems.

Specific comments on the remaining four questions now follow.

Question 1: Participants answered the question verbally, and only 3 performed any written working. An example of the simplified and unlabelled diagrams drawn is shown in Figure 5.1. As mentioned in Chapter 4, the transcripts showed a lack of intermediate stages in the solution path: either the participants knew that torques and

conservation of angular momentum were involved or they did not know how to proceed.

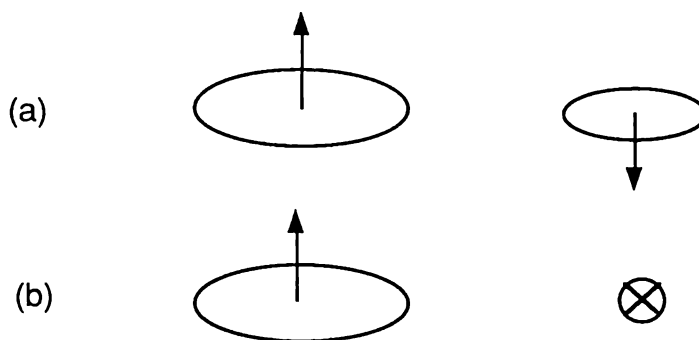


Figure 5.1: Examples of diagrams drawn by N3 to depict torque or angular momentum. (a) For both propellers on vertical axes. (b) For one propeller on a horizontal axis.

Question 2: The solution requires knowledge of rolling, of the relationship between translational and rotational motion, and understanding that energy is not dissipated as heat since the frictional force does not perform any work due to the rolling motion. Only 1 expert (E1) explicitly mentioned the relationship between rotational and translational motion. One novice (N6) from the 13 participants performed calculations or quantitative representation while problem solving. The other 12 participants attempted the question verbally and intuitively. Three participants reproduced a simplified form of the diagram given in the question.

The qualitative nature of questions 1 and 2 seemed to encourage a verbal solution with no consideration of intermediate steps. The two questions provide good examples for qualitatively discussing and relating ideas about object movements to physical principles. The transcripts for these questions provide rich data for studying conceptions and qualitative or verbal problem solving. The problems were not considered suitable for program development, since as mentioned in Chapter 4, program interaction requires written input of information in the form of words or symbols.

Question 4: Only the lecturer successfully solved question 4 (Table 5.1). This problem was considered too difficult for the pilot program. It would be a good problem to include if problems of gradational difficulty were used to make up the program.

Question 6: All the experts and 1 novice (N3) successfully solved this problem (Table 5.1). One expert (E1) made a slight algebraic error but his approach was correct. The transcripts of a successful expert solution (E2), and a successful (N3), a partially successful (N7) and two unsuccessful novice solutions (N1 and N8) are given

in Appendix B. The five transcripts show the variety of solution paths and the richness of data collected. This problem was considered to be reasonably difficult but not beyond the reach of year one undergraduates. For this and reasons given in Chapter 4, it was chosen for further data collection and eventual program development.

The next section discusses the 13 think-aloud, and the 8 pen and paper transcripts of the participants as they attempted question 6. The discussion first focuses on the problem-solving process used by the participants starting with the activities they performed as they commenced problem solution. Excerpts from transcripts are provided in two columns. The first column consists of the verbalisations and the second any corresponding written solution performed. Comments or explanations provided by the researcher are placed within parenthesis in the second column.

5.3 THE PROBLEM SOLVING PROCESS OF THE PARTICIPANTS

5.3:1 Activities performed at commencement of problem solving

The first verbalisations and activities performed are detailed since they give insight into the problem-solving approach used. Participants indicated first thoughts as they read the question or immediately on completion of the reading. Table 5.2 gives a summary of the different activities performed by the 21 participants.

The four experts verbalised inferences as they read the question indicating the processing of information . Three experts started with a description of energy changes while they were reading the question or immediately upon completion of question reading. The excerpt given below provides an example of the description performed.

	Think-aloud													Pen and paper							
Activity performed	E1	E2	E3	E4	N1	N2	N3	N4	N5	N6	N7	N8	N9	P1	P2	P3	P4	P5	P6	P7	P8
Described energy changes		•	•	•	•	•	•		•		•								•		
Drew and labelled a diagram			•	•				•					•			•	•			•	•
Sought question clarification					•																
Wrote a force equation									•		•		•	•			•				
Envisioned the motion	•						•	•													
Wrote an energy equation	•	•								•		•			•						•
Began solving for sliding motion						•								•				•			
Began solving for rolling motion										•		•			•						
Wrote displacement expression																•		•		•	
Wrote acceleration expression																			•		

Think aloud: Experts E1 - E4
 Novices: N1 - N9
 Pen and paper: Novices P1 - P8

Table 5.2: The first activities of the participants when problem solving.

Now rolls without slipping. So its also rolling so we going to have energy connected with its rotational motion. So let's see I guess, the particular point on that slope as it comes down is going to have a certain amount of potential energy until it reaches the bottom. It will have kinetic energy too due to its linear motion and it will have kinetic energy tied up in its rotational motion. So, the energy at a point is going to be the sum of those three quantities.	(Traces the motion of the ball, from the start to a point in the loop. Recognises gravitational potential, translational kinetic and rotational kinetic energies are involved). E2
---	--

The other expert performed a complete envisioning of the motion of the ball. This included a discussion of the energies involved at the various positions of the ball and the constraints needed for the ball to successfully perform the loop (see the excerpt given below).

If the ball falls, it starts with a certain amount of gravitational potential energy. As it falls this becomes kinetic energy of both rotation and translation. Its gravitational potential energy will be minimum at the bottom of the loop. Its kinetic energy of rotation and translation will be a minimum at the top of the loop. The point to look at is the top of the loop. This is the critical point. If the velocity at this point is large enough to provide an acceleration which can support the downward forces then the ball will remain in contact with the track. The constraint for this is that the force of the track on the ball must be greater than zero. So, the minimum value of h will be when this force is zero or just about. If the ball does not leave the track before it reaches the top of the loop then it will perform the loop.	(Describes the energy changes and mentions all the energies involved). (Considers the critical point). (Mentions the constraint which is important for the solution of the problem). E1
--	---

The experts processed the information and obtained an understanding of the motion before they began any quantitative representations. Two drew diagrams and first represented the given information, and then represented further information on the diagram as they described the energy changes, or performed the envisioning.

The 6 novices who described the energies involved did so very briefly and then immediately wrote an energy or force equation(see the excerpt given below).

A uniform ball of radius r rolls without slipping along the loop the loop track shown in the figure. It starts from rest at height h above the bottom of the loop. If the ball is not to leave the track at the top of the loop, what is the least value h can have in terms of the radius R of the loop. What would h have been if the ball were to slide along a frictionless track instead of rolling ? Okay, at any point its total energy is half $I \omega^2$ squared plus half $m v$ squared which has components plus $m g h$ dash wherever. h is still h which is always equal to its initial energy $m g h$.	(Read the question word for word and did not make any inferences while reading) $\frac{1}{2} I \omega^2 + \frac{1}{2} m v^2 + m g h' = m g h$ (Immediately performed a mathematical representation to solve the question.) N8
---	--

Two novices envisioned the motion of the ball. They did so very briefly and at a very superficial level using surface features such as circle, track, loop and height (see the excerpt given below). They immediately went on to perform quantitative representations by writing equations.

We've got, going around the loop starting at rest at a height from the bottom of the loop. If the ball is not to leave the track at the top of the loop means that we have to have er quite a bit of speed on. So we can start this by saying that we got something travelling in a circle which means that we might need to apply some centripetal force. We'll try that at the moment. We've got force equals $m v$ squared over R	(Except for speed no mention of any other physical quantity involved is mentioned until the equation $F = \frac{m v^2}{R}$ was written) N9
---	---

Except for one participant who wrote a few lines describing the energy changes (P1), the pen and paper participants did not perform any description of the motion. The written form of the exercise seemed to encourage the immediate writing of equations without first obtaining an understanding of the situation. This may be attributed to the expectancy of obtaining the solution by applying an algorithm.

Two experts and 8 novices completely read the question before inferring any solution steps. The other 2 experts made inferences while reading the question. One novice read the first part of the question and commenced the solution, and read the next part of the question when attempting the solution to that part.

The main purpose was to discover the purposes for which students would use the program, and features which need to be represented in the program.

The next significant feature of the interaction with the problem was question clarification and the seeking of assistance.

5.3:2 Question clarifications and assistance sought

Information sought for question clarification, and assistance to obtain or clarify relations between physical quantities was mostly sought from the text supplied. Some assistance was supplied by the researcher to locate required information when the participant was unsuccessful. The information sought is itemised below starting with the question clarifications requested (in *italics*) followed by an explanation. When information sought was not simply for a relation between quantities then an excerpt from the transcripts is provided to place the request in context.

Question clarifications

1. *Is that raised to the height h above the bottom of the loop?*

The participant wished to confirm that the starting height of the loop was from the ground to the bottom of the ball.

2. *Does uniform mean uniform density or is it mean density ? Is there any difference between mean and uniform ?*

The participant was seeking clarification of the word *uniform*.

3. *Is mass given ? Do we know it ?*

Realising that mass appeared in the expressions for energy and force, the participant wished to know whether it could be considered as a known quantity. It was not realised that mass cancels when the energy and force expressions are simplified.

4. *Am I to assume that r is negligible to R ?*

The participant inquired whether r should be included in the calculations for the height of the ball. It had to be included to obtain the starting height of the ball from the ground.

5. *Is it a solid ball or is it hollow? It could have nothing inside, like a shell.*

The participant inquired about the nature of the ball to determine which rolling inertia expression to use. The moment of inertia (I) for a solid sphere about an axis through its centre is $\frac{2}{5} Mr^2$ while that of a hollow sphere for the same axis is $\frac{2}{3} Mr^2$.

6. *How is the motion different when it slides or when it rolls?*

The participant was not aware of the condition required for rolling whereby the velocity of the centre of mass of the rolling object is equal to its tangential velocity of rotation. The expression $v = \omega r$ is only applicable for a rolling object or for rotation about a circle. For the case of rolling r is the radius of the rolling object, whereas for rotation it is the radius of the circle which the object is performing.

Queries 1, 4 and 5 arose due to ambiguities in the question. The measurement of height (h) from the bottom of the track to the base of the ball created problems (see the diagram for question 6 in Chapter 2). This meant r had to be included to obtain the height of the centre of mass of the ball above the bottom of the track. Except for the lecturer, none of the other participants included r in their calculations for the heights of the ball at different positions. Furthermore, the final expression for the starting height (h) could be obtained even if participants calculated the heights of the ball without including r . Therefore, the question was modified such that h was measured from the bottom of the track to the centre of mass of the ball. This meant, r did not have to be included for the calculation of the heights. To eliminate query 5 on the ambiguity of the inner nature of the ball, the word *solid* (Figure 4.3 for the modified question) was included in the question.

Queries 2, 3 and 6 were not attributed to the wording of the question. *Uniform* is a well used term and only one participant was confused by it. By using the word mean, the participant was able to resolve the meaning. However, an explanation of *uniform* is included in the program. Query 3 was addressed as a symbol for mass is included for writing energy and force expressions in the program. A value for it did not have to be given as it cancels from the final expression. Query 6 was addressed by including in the program an explanation of sliding, slipping and rolling.

Assistance sought

The other category of assistance sought involved mathematical expressions that the participants considered relevant to the solution. The following nine expressions were requested.

mgh

The expression for the gravitational potential energy in terms of the mass (m), the gravitational intensity (g) and the displacement from the zero potential energy reference level .

Now let's, so starts at rest, the initial, it has got initial potential energy which was one half h. What's the formula for potential energy in terms of height ? Is it one half m g h ? No, its just m g h isn't it ?	(Thought the potential energy was given by $\frac{1}{2} mgh$) (Refers to text to resolve the matter) $E_{pot} = mgh$ N1
--	---

$L = mvr$

The expression for the angular momentum (L) of a particle in terms of the mass (m), velocity (v) and displacement (r) from the axis of rotation.

Okay, now for looking back at the thing, orbital angular momentum, I can write, what's the formula that's, I'm sorry there's a formula that's, got theta dependence on it. Its a half m r or something like that No, will be m v R. Can we consider this a particle, but still got an r dependence in it that I probably don't want and that's to be honest not too dissimilar from what I had but let's just have look at that. Its got to equal, m v big R and its got to equal the where was it I had it on this diagram m g R, m g times two R. So the m's cancel out. The R's cancel out so my velocity must equal two g.	(Referred to the text to confirm the formula he wanted was mvR) (Equated angular momentum to the gravitational potential energy) $mvR = mg2R$ $v = 2g$ N2
--	--

$$L = I\omega$$

An expression for the angular momentum (L) of a body of moment of inertia (I) rotating about an axis. The angular velocity of rotation is ω .

So the ball is in motion, we can say that the orbital force that m the orbital force of gravity which just call m g big H. Okay orbital angular momentum is m v squared r I think. Okay, I omega that was the, I omega that's the constraint there, so that's m g times the height of the loop.	(Searching for an expression to substitute for the velocity of the ball and refers to text to obtain the expression) (Equates the gravitational potential energy to the angular momentum) $mgH = I\omega$ N2
---	---

$$I = \frac{2}{5}mr^2$$

The expression for the moment of inertia (I) of a solid sphere about an axis through its centre. This expression is needed to replace I in the expression for the rotational kinetic energy given by $\frac{1}{2}I\omega^2$.

$$a = \frac{v^2}{r}$$

The acceleration towards the centre, for an object performing circular motion of radius r. v is the tangential velocity of the object. This formula was requested by many of the participants as it is a required substitution for the acceleration towards the centre of a circle.

$$F = \frac{mv^2}{R}$$

The force towards the centre of rotation for an object performing circular motion of radius R is equal to the product of the mass and the radial acceleration. $\frac{v^2}{R}$ is the radial acceleration and v is the tangential velocity of the object.

There is a formula for force centripetal, without having a string. That one there the force on a string. Has got to be, there we are. Force gives us m g times R, the force exerted by the pail is therefore, here it is. So it is m v squared at the top over R. So that is m g equal to the force.	(Referring to the example of a pail being swung in a circle at the end of a string). $mg = \frac{mv^2}{R}$ N4
---	---

$$v = \omega r$$

The expression for the velocity of a rolling sphere of radius r or the tangential velocity of rotation for an object performing a circle of radius r . ω is the angular velocity of rotation.

$$\text{Translational kinetic energy} = \frac{1}{2} mv^2$$

m is the mass of the moving object and v is the translational velocity of the centre of mass.

$$\text{Rotational kinetic energy} = \frac{1}{2} I\omega^2$$

I is the moment of inertia of the object about an axis round which the object is spinning or rotating and ω is the angular velocity about that axis. For rolling, the axis is taken through the centre of mass of the rolling object and ω is the angular velocity due to the rolling motion of the ball.

Except for $L = mvr$ and $L = I\omega$, the information sought by the participants to aid their problem solving has been included in the program. The two relations of L were not included as they are not needed for the problem solution. When it was deemed necessary an explanation of the quantity or the relation between quantities is given in the program.

Reference by participants to examples of solved problems indicated links to other information relevant to program development. This is now described.

5.3:3 Examples referred to for help

Participants referred to two standard examples in the text provided for reference, both presented as solved examples. The two very similar examples related to a pail of water and a block, tied to the end of a string and whirled in a vertical circle. Six novices referred to the example of the pail whirled around in a circle (Figure 5.2) and 2 referred to the similar example of a block.

A pail of water is whirled in a vertical circle of radius r . If its speed is v_t at the top of the circle, find the force exerted on the water by the pail. Find also the minimum value of v_t for the water to remain in the pail.

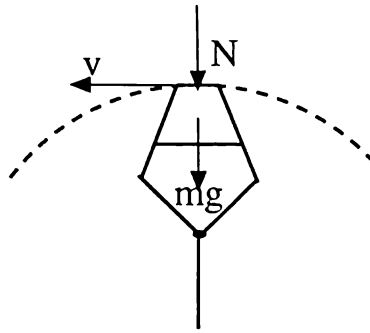


Figure 5.2: Example referred by the participants.

N and mg are the reaction force of the pail on the water and the weight of the water respectively. This is similar to question 6 where N would be the force of the track on the ball and mg would represent the weight of the ball. In the example, the following information was also given:

$$N + mg = m \frac{v_t^2}{r}$$

The minimum speed the water can have at the top of the circle occurs when $N = 0$

From this information the participants were able to obtain the equation

$$mg = m \frac{v_t^2}{r}$$

and deduce the expression for the minimum velocity of the ball at the top of the loop:

$$v_t = \sqrt{rg}$$

where v_t is the minimum velocity for the water not to fall out, or in the case of question 6, the minimum velocity at the top of the loop for the ball to successfully traverse the loop.

Only 1 of the 6 novices who referred to this example solved the problem. An excerpt from one of the novice protocols is given below to show the reference to the example mentioned.

There is a formula for force centripetal without having a string. That one there the force on a string. Has got to be, there we are. Force gives us m g times R the force exerted by the pail is therefore here it is. So it is m v squared at the top over R. So that is m g equal to the force. Now that will be zero at the top of the loop because that means it, m g must equal m v squared over R. m's can cancel, so its going to be just g and v R squared.	$\frac{mv^2}{R} = mg$ (Deducing that N = 0 at the top of the loop) N4
--	---

The participants who obtained the simplified force expression $\frac{mv^2}{R} = mg$ and the expression for the minimum velocity, were unable to solve the problem confirming that the problem cannot be solved with a simple algorithm but requires the application of other principles. The participants needed to relate the energies of the ball at two different places and to substitute the expression for the minimum velocity obtained from the example into the resulting expression . Chi et al (1989) found that novices who refer to examples to solve a physics problem are normally searching for the solution rather than trying to understand and adapt the example to the new problem. The novices used the force equation but did not understand it. Their next step was to look for a way of relating height, h, to the velocity that had been found. Novices were looking for a one step solution, an attribute of novices reported in the research on novice and expert problem solving. This is discussed in greater detail later in this chapter.

This example provides the information needed to write the force equation and the critical condition, N = 0, required for the simplification of the equation. However, the example does not explain why the condition N = 0 gives the minimum value of the velocity. The novices could mathematically deduce that if N equals zero, then the velocity would be a minimum. Only 1 of the 6 novices understood that N was a force acting on the water due to its contact with the pail. They used terms such as centripetal and centrifugal to refer to N. This implies that an explanation of the condition N = 0, needs to be provided in the program.

Two novices considered the example misleading. They thought that since there was no motion in the radial direction i.e. vertically up or down then mg should equal N and the forces should cancel. This was a popular misconception amongst the novices. The reason being they just considered the forces and did not consider the acceleration. To them acceleration is associated with movement of an object in a certain direction. This is one of the popular misconceptions mentioned in the literature (McCloskey, 1983). It is

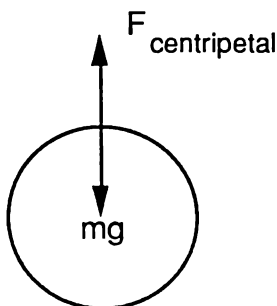
most likely to occur due to the lack of distinction or discrimination between velocity and acceleration.

The use students made of examples provides insight into the type of information that helps them to solve the problem. Mere provision of information is insufficient, it needs to be accompanied by an associated explanation which can be accessed if the student wishes.

5.3:4 General mistakes made while problem solving

The mistakes made by the participants while attempting the solution may be classified as mathematical, representational and procedural. The mathematical mistakes were generally made when substituting or manipulating quantities numerically or algebraically, and when simplifying an expression. They occur when adding, subtracting, multiplying and dividing quantities and are not elaborated as they commonly occur among students. The representational mistakes included diagrammatic errors, using incorrect relations to represent physical quantities, and using similar symbols for different quantities. Procedural errors occurred due to a lack of knowledge leading to partial consideration of information. Examples of representational and procedural errors are discussed below.

Example 1: A diagrammatic representational error



In the above diagram the normal reaction force (N) of the track on the ball has been left out. The centripetal force and gravitational force have been drawn opposite to each other indicating a system in equilibrium. The solver did not realise that the forces did not balance since the ball is not in equilibrium at the top of the loop. The acceleration towards the centre of the circle was not drawn as there was no motion in the direction of the forces. Another reason for drawing the centripetal force in the opposite direction was confusion between the terms centrifugal and centripetal (the centrifugal force is taken to act towards the centre of the circle). Eleven of the novices left N out of their force diagram for the top of the loop. When questioned 7 were not aware of the reaction force and the other 4 had forgotten to include it.

Example 2: Using incorrect relations

Below are 4 examples of the common errors made in representing physical quantities.

1. $Kinetic\ energy = I\omega^2$ (should be $\frac{1}{2} I\omega^2$)

2. $Force = mv^2R$ (should be $\frac{mv^2}{R}$)

3. $I = \frac{1}{2} mr^2$ (should be $\frac{2}{5} mr^2$)

4. $I = \frac{2}{5} mR^2$ (should be $\frac{2}{5} mr^2$)

These highlight the common errors made in representing physical quantities.

Another common representational error is using algebraic symbols to represent different quantities. Examples of this are:

Using the radius of the ball (r) in place of the radius of the circle performed by the loop (R) as shown in case 4 above.

Using the same velocity symbol to represent the velocity at the start, top and bottom of the loop (Example 3).

Example 3: A representational error due to using the same symbol to represent different quantities.

The force expression for the top of the loop was used to obtain an expression for the velocity at the top. This was represented as $v^2 = Rg$. Then an energy expression equating the energy at the start and the bottom of the loop was written. The same velocity symbol, v was used to represent the velocity of the ball at the bottom of the loop. From this $v^2 = 2gh$ was obtained. The two velocities were equated to obtain $2h = R$, an incorrect expression for the value of h. The solver did not realise that different values of the velocities were being equated.

Example 4: A procedural error due to partial consideration of information.

The following expression was written in equating the energy of the ball at the start and the top of the loop.

$$mgh = \frac{1}{2} I\omega^2 + \frac{1}{2} mv^2$$

The participant neglected to include the gravitational potential energy of the ball at the top of the loop. The expression should have read

$$mgh = \frac{1}{2} I\omega^2 + \frac{1}{2} mv^2 + 2mgR$$

The general mistakes made by the participants highlight the need for the inclusion of the following features in the program:

- A reference section which includes all the relevant relations between physical quantities.
- A check mechanism to detect non-diagrammatic representational errors.
- A component for drawing force diagrams to aid in diagrammatic representation.

5.3:5 Conceptions portrayed

The conceptions portrayed about the motion of the ball and its associated forces and energies are now discussed. For each conception different from the general theories of physics an excerpt from the transcripts is provided with an explanation of the alternative conception. The related physics information required to address the conception is also given.

There are two rotational energies

In the transcript below $\frac{1}{2} I\omega^2$ and $\frac{1}{2} mR^2\omega^2$ were used as the energies due to rolling and rotation respectively. The participant had the conception that straight line motion had translational kinetic energy associated with it, and rotational and rolling motion had rotational kinetic energies associated with them. The participant did not understand that as the centre of mass of the object is changing position there is translational energy involved.

Its actually moving in a circle there. I don't know whether that, must, have to take that into account. Seems to be not having any translational, I suppose there isn't any translational energy really because, no that's right once it goes on that loop, once its on that loop there, no there isn't any. Okay, I was just worried that there was no. Seems that would leave me with no $E_k T$ but I think that's right, there wouldn't be anyway because it isn't really moving in a straight line at all anyway. I think I'll say there's no translational kinetic energy at all I think I was right when I said that a few minutes ago but this energy of rotation of the ball and I feel there's also energy of rotation of the fact that the ball is going round the circular track, just thinking that it is a small particle or something like that, I think that yes, that is, that is the second form, that is the second bit of energy there is I think so there's some sort of E. I'll call it $E_k R$, well $E_k R$ two or something	(Conception that since the object is performing rotational motion in a loop there is no translational kinetic energy). (Has the conception that the translational kinetic energy is for objects moving in straight lines) (Implies there are two rotational energies. One due to rotation in the loop and another due to the rolling motion of the ball). N9
--	--

Minimum height is obtained if the velocity is zero at the top of the loop

The limiting thing that, it will fall off the track. If the speed at the top of the track is, zero is the limiting, limiting sort of value there. So, in other words if rolls without slipping. If it is not moving when it gets to the top of the loop it'll fall off. So if its not, if its stationary when it gets to the top the loop means the only energy its got will be the energy gravitational potential energy so if, right, so that would mean the least value h could have would be two R.	(Conception that velocity must be at least zero to successfully perform the loop) (Has the conception that if the velocity is even slightly larger than 0 then the loop will be performed. N9
--	--

One expert and 12 novices thought that the minimum starting height of the ball could be obtained by setting the velocity at the top of the loop to zero. They were unable to envision that the ball would leave the track before the velocity became zero. They were not aware that the velocity had to be large enough so that the central acceleration would be sufficient to enable the ball to perform the loop. The expert and 5 of the 12 novices later realised that the velocity could not be zero because it would fall vertically downwards and would not move forward. Substituting a zero value for velocity in the energy equation gave a value of $2R$ for the starting height. Participants used the knowledge that if an object started at a height h , and assuming it stayed in contact with the track then it would only reach the same height at the other end of the track and would oscillate between the starting and end points. They failed to note that if the starting height of the ball was greater than R (the radius of the circle performed by the ball) but less than the minimum height for the loop to be completed then it would leave the track. They were thinking of a track given in Figure 5.3 below.

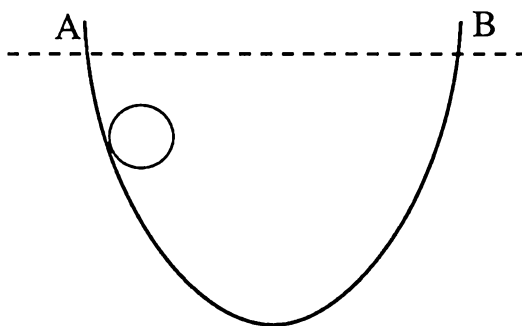


Figure 5.3: Ball oscillating between two points.

In the above case the ball would oscillate between A and B, the two points at the same height (Figure 5.3). They used this analogy of the oscillating ball to resolve the fact that the velocity could not be zero at the top of the loop. It is a good analogy to use if the ball were to start at a height less than R . When it starts at a height larger than R then it will leave the loop and not perform any oscillations.

They failed to understand that the success of the ball at performing the loop was dependent on the central acceleration towards the centre of the loop. This in turn was dependent on the tangential velocity and the radius of the circle (R) performed by the ball. This provides a good example of faulty analogous reasoning to resolve the matter correctly.

The force upwards must equal the force downwards at the top of the loop.

Yes, the least value h can have, so that's the critical condition. $m v^2 \text{ over } R$ must equal mg The force upwards must equal the force downwards point the gravity downwards, acceleration upwards	(No diagram was drawn but the transcript indicated that the acceleration was opposite to the force mg) N3
--	--

The conception is that since there is no motion in the radial direction the resultant force must be zero, therefore there has to be an opposite force to balance the gravitational force. This is required to support the ball and keep it from falling from the track. Seven novices drew the reaction force of the track on the ball pointing vertically up, in the opposite direction to the force of gravity. This is described in an earlier section on diagrammatic representational errors. An expert and 4 novices drew the centripetal force as balancing the gravitational force. The participants failed to consider the tangential velocity, v of the ball (Figure 5.4) which carries it through the loop.

The forces present at the top of the loop are the gravitational force mg , and the reaction force, N of the track on the ball. Both these forces act towards the centre of the loop as shown in Figure 5.4. The forces do not balance since there is an acceleration a (which can be inferred from the change in direction of the velocity). The acceleration is produced by the resultant force which acts towards the centre of the loop as well. This gives the force equation:

$$\frac{mv^2}{R} = mg + N$$

For the minimum value of the height the value of N is taken to be zero.

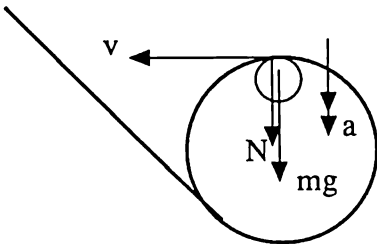


Figure 5.4: The forces and acceleration for the top of the loop.

Energy has direction and is like a force

The energy pulling it down mg is the force that's acting down there. Half mv^2 is that way half ω^2 is what's rolling there	(Gave energy direction and treated it like force. Drew Figure 5.5 to indicate the direction of the energies) N8
--	--

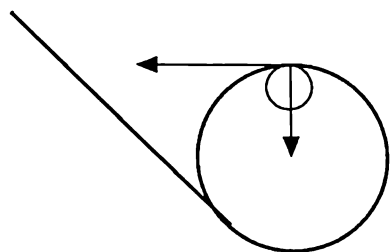


Figure 5.5: Energy treated as forces and given direction.

In the example above, the participant treated gravitational potential energy as the gravitational force and the translational kinetic energy as a horizontal force keeping the ball moving to the left as shown in Figure 5.5. The conception shown here is that for an object to move in a certain direction it must have a force acting on it in that direction. Four of the novices portrayed this conception. This is in line with the misconceptions of children and year one physics students in New Zealand, as reported by Osborne and Freyberg (1985) and Singh (1989) respectively. Velocity, energy and forces are treated as similar in the sense that they are associated with motion.

Force and energy were also treated as being similar when writing force equations. An example of this is given below.

$$mg2R + \frac{1}{2} I\omega^2 + \frac{mv^2}{R} = mgh$$

In this case the force expression $\frac{mv^2}{R}$ has been used instead of the expression for the translation kinetic energy of the ball.

The tangential and centre of mass velocities are not equal for rolling.

What will happen is that because it is rolling its centre of mass will move. No it is, the whole different speed. The ball's rolling therefore the speed at which the edge is touching. Edge speed is different.	(Asserting that the velocity of the centre of mass is different from the tangential velocity for a rolling ball.) N4
--	---

The conception is that the velocity of the centre of mass is not equal to the tangential velocity of the ball for rolling motion. The participant has the conception that the point of contact (the edge) is moving. In rolling motion (when the velocity of the centre of mass is equal to the tangential velocity) the point of contact of the sphere with the track has zero velocity. For sliding and slipping the tangential velocity is not equal to the velocity of the centre of mass. Only one novice explicitly portrayed this conception but several others were not aware of the reason for the applicability of the equation $v = \omega r$. All of them used or attempted to use this equation. Two novices were aware that the formula could be used because the velocity of the centre of mass was equal to the tangential velocity. One expert was not able to relate this. The novices applied the formula to rolling because rotational motion was present.

The conceptions reported in this section have been taken into account in the program by including

- separate force and energy equation checking facilities;
- simulations of the ball for various heights and radii of the loop;
- relevant explanations and diagrams to help the students come to terms with their conceptions;
- an interactive force drawing facility; and
- an envisioning feature describing the motion of the ball at various stages.

5.3:6 Major difficulties encountered while problem solving

In this section the difficulties encountered by participants due to a lack of understanding of the physics principles and concepts required to solve the problem are summarised. The major stumbling blocks were lack of understanding of the:

- motion of the ball.
- energy principle and its application to relate the energies at different points.

- forces involved and the forming of force equations.
- constraint required to enable the ball to successfully traverse the loop.
- relationship between force and energy equations.

Each of these are briefly discussed in the above order.

Motion of the ball: The participants knew the path that the ball would take if the height was sufficiently large for it to successfully traverse the loop. However, they had problems about how the height affected the success of the ball in performing the loop. They were not aware that the critical path of the ball was on its upward motion in the top half of the loop. If the ball reached the top of the loop without leaving the track, then it would not leave the track at any other point in its path. The 4 experts and 5 novices were able to envisage the motion of the ball for different starting heights. As stated in the discussion on conceptions, they thought that the ball would fall vertically when the starting height was insufficient. Fifteen participants (4 experts and 11 novices) knew that the ball had to start higher to roll without sliding. From the 11 novices, 6 related it to the fact that at the same height the rolling ball would have a smaller velocity.

Energy principle and its application: The participants had difficulty recognising the energies involved for the different motions (rolling and sliding) of the ball and at the different positions of the ball. They were unsure of the positions of the ball for which the energy equation should be written. A further problem was simplifying and combining the kinetic energy expressions for translation and rotation. The presence of friction for rolling motion made 6 novices unsure whether to include the work done by the frictional force. Three of the other 13 novices did not consider the frictional force because it was along the tangential direction. The other 10 did not even realise that there was a frictional force present or that it figured in rolling motion.

Forces involved: The 4 experts and 12 novices knew that a gravitational force and a force due to the reaction of the track on the ball were involved. Their difficulty was relating these forces to the motion of the ball so as to write a force equation. The novices showed a poor understanding of applying Newton's second law of motion (for constant mass) to form a force equation. They had difficulty resolving the forces to obtain the resultant force in the direction of importance for the problem and in relating this to the acceleration along the chosen direction. However they knew that the force equation should be written for the top of the loop, though as stated earlier, some were not aware of the reaction force of the track on the ball.

Applying the constraint: Only 3 novices and an expert explicitly stated that the value of the reaction force of the track on the ball ($N = 0$) could be used to determine whether the ball was in contact with the track. The other 3 experts knew of the condition although they did not verbalise it. However, the other novices did not realise that N was a contact force and did not exist if there was no contact between the ball and the track. Nineteen of the participants knew that the top of the loop was the point to consider. What they failed to consider was that the ball would only leave the track at a certain part of its path as mentioned in the discussion on the motion of the ball.

Relating the force and energy equations: Having successfully written the force and energy equations participants had to simplify them, to relate the quantities involved and to make appropriate substitutions. Some had difficulty relating the information from the two equations and making the substitution required to obtain the minimum starting height of the ball. As can be seen from Tables 5.3 and 5.4 only a novice and the 4 experts successfully solved the problem.

These major areas of difficulties have been included as separate sections in the program development, except for the constraint application. The program has been developed so that the user can practice writing energy and force equations for various positions of the ball in the loop and on the incline. The constraint information which is crucial to problem solution is included in the final analysis together with information on relating the force and energy equations.

5.3:7 Knowledge used to solve the problem.

The knowledge used by the participants when solving the problem can be compared to that needed to meaningfully solve the problem. (Tables 5.3 and 5.4). The first (Table 5.3) derives from the think-aloud investigation and the second (Table 5.4) from the pen and paper solution of the modified problem. Participants were considered to have the knowledge indicated if they verbalised it or wrote it as part of their solution. The knowledge needed to solve the problem has been grouped into energy expressions at the start, at the top of the loop, and that needed to write the force equation. The items, energy equation and force equation simply indicate whether the participants wrote those equations. The last two rows labelled h for sliding and h for rolling indicate whether the participants obtained the correct final expression for sliding and rolling.

Knowledge needed	E1	E2	E3	E4	N1	N2	N3	N4	N5	N6	N7	N8	N9
Energy considerations at the start													
height = $h + r$	•												
P.E. = $mg(h + r)$	•	•	•	•	•	•	•	•	•	•	•	•	•
K.E. _T = 0	•	•	•	•	•	•	•	•	•	•	•	•	•
K.E. _R = 0	•	•	•	•	•	•	•	•	•	•	•	•	•
Energy considerations at the top of the loop													
height = $2R + r$	•												
P.E. = $mg(2R + r)$	•	•	•	•	•	•	•	•		•	•	•	•
K.E. _R = $\frac{1}{2} I \omega^2$	•	•	•	•	•		•	•	•	•	•	•	•
K.E. _T = $\frac{1}{2} mv^2$	•	•	•	•	•	•	•	•	•	•	•	•	•
Considering forces at the top of the loop													
mg	•	•	•	•	•	•	•	•	•	•	•	•	•
Direction of mg	•	•		•	•								
N	•				•	•	•						
Direction of N	•				•								
$\frac{mv^2}{R}$	•	•	•	•	•	•	•		•	•	•		
Direction of $\frac{mv^2}{R}$	•	•			•								
$N \geq 0$	•	•			•	•	•						
$I = \frac{2}{5} mr^2$	•	•	•	•	•		•	•			•	•	•
$v = wr$	•	•	•	•	•		•	•	•	•	•	•	•
Force equation	•	•	•	•	•	•	•						
Energy equation	•	•	•	•	•	•	•	•	•	•	•	•	•
h for rolling		•	•	•	•								
h for sliding		•	•	•	•					•			

Experts: E1 - E4 Novices: N1 - N9

Table 5.3: Summary of information used by participants of the think-aloud session.

Knowledge needed	N1	N2	N3	N4	N5	N6	N7	N8
Energy considerations at the start								
height = h								
P.E. = mgh	•	•	•	•		•	•	•
K.E. _R = 0	•	•		•		•	•	•
K.E. _T = 0	•	•		•		•	•	•
Energy considerations at the top of the loop								
height = 2R								
P.E. = 2mgR	•	•	•			•	•	•
K.E. _R = $\frac{1}{2} I\omega^2$	•	•		•		•		•
K.E. _T = $\frac{1}{2} mv^2$	•	•		•		•		•
Considering forces at the top of the loop								
mg	•	•		•				•
Direction of mg								
N	•	•						
Direction of N		•						
$\frac{mv^2}{R}$	•	•		•				
Direction of $\frac{mv^2}{R}$								
$N \geq 0$	•	•						
$I = \frac{2}{5} mr^2$	•	•		•				
$v = \omega r$	•	•		•		•		•
Equate forces	•	•	•	•				•
Equate energies	•	•	•	•		•	•	•
h for rolling	•							
h for sliding	•	•				•		

Novices: N1 - N8

Table 5.4: Summary of information used by participants of the pen and paper session.

Comments:

1. It can be noted from the table that many participants had the necessary knowledge to write the force and energy equations but were unable to obtain the answer. Many participants failed to mention that N should be greater than zero. This does not mean that they did not know or use the condition. It simply means that they wrote the force equation $\frac{mv^2}{R} = mg$. Those who did not indicate certain information but seemed to use it for the solution of the problem were asked for clarification during the interviews. Most of the novices were not aware of the constraint needed. Some had obtained the above equation from the book.

2. The only difference in the information is that the heights used to calculate the gravitational potential energy at the start and at the top of the loop are different. In Table 5.3 the heights at the start and at the top of the loop are $h + r$ and $2R + r$ respectively. In Table 5.4, r is not included in the heights as a result of problem modification.

3. Participants used forces without indicating the direction in which they act. This however does not mean that they did not know the direction. Through interviews this was clarified. That information however is not included in the table. Most of the participants knew the direction of the gravitational force mg and the central force $\frac{mv^2}{R}$. The direction of the reaction force N , of the track on the ball was not known by many of the novices.

3. In Table 5.3, only expert E1 indicated the heights at the start and the top of the loop as $h + r$ and $r + 2R$ respectively. He made a slight algebraic error and did not manage to obtain the final expressions for the starting height of the ball. He had all the knowledge required for the solution but obtained $h = \frac{27}{10}(R - r)$ for rolling and $h = \frac{5}{2}(R - r)$ for sliding. All the others used h and $2R$ as the respective heights in the calculation of the potential energy at the start and the top of the loop .

Next the transcripts are used to discuss the activities performed by the participants while problem-solving. These are used to represent the problem solving flow charts and concept maps of the participants.

5.4 ACTIVITIES PERFORMED WHILE PROBLEM SOLVING

In this section the problem-solving activities performed by the participants are discussed. These have been classified into fourteen categories and labelled question

Activity	Think-aloud participants												
	E1	E2	E3	E4	N1	N2	N3	N4	N5	N6	N7	N8	N9
Analogous reasoning	0	0	0	0	3	1	1	1	0	1	0	0	2
Checking	0	3	9	4	1	2	4	1	4	0	2	1	3
Clarifying	4	5	2	2	7	6	4	15	2	0	3	5	6
Declaring quantities	13	11	18	17	8	19	23	16	9	11	6	9	9
Pictorial representation	1	1	2	2	5	0	4	0	0	0	0	2	0
Resolving difficulties	2	3	13	13	5	11	14	18	7	10	8	10	11
Qualitative analysis	2	3	3	0	6	4	0	6	2	0	5	2	8
Qualifying	7	12	8	8	11	12	21	12	5	7	3	4	2
Quantitative representation	13	11	16	16	16	24	20	12	10	13	9	8	18
Question reading	1	4	4	2	4	2	1	1	4	3	4	2	4
Recapitulating	1	4	8	5	7	8	10	13	5	3	3	3	12
Reference	0	0	2	1	7	6	3	1	0	3	2	1	8
Relating quantities	8	12	14	10	4	15	22	5	6	7	6	4	13
Symbol usage	2	2	1	1	3	5	3	4	3	1	2	1	5
Total	54	71	100	81	87	97	130	105	57	59	55	52	101

Experts E1 - E4 Novices N1 - N9

Table 5.5: Frequency of activities performed by the think-aloud participants.

reading, pictorial representation, declaring quantities, clarifying, qualitative analysis, reference, analogous reasoning, qualifying, resolving difficulties, recapitulating, relating quantities, symbol usage and checking. Excerpts from the transcripts are used to provide an example of each activity's occurrence within the problem-solving process. This is followed by a brief discussion of the different aspects of the activity. It should be noted that the activities performed are mainly inferred from the think-aloud transcripts. The frequency at which the think-aloud participants performed these activities is given in Table 5.5. The pen and paper transcripts also provided some examples for categories such as pictorial representation, reference, relating quantities, resolving difficulties, quantitative representations and symbol usage. Each of the activities are first explained.

Analogous reasoning

Analogous reasoning included the use of examples and information of a similar nature to that under consideration in attempting to make sense of the situation.

A similar system will be say a string with a mass on the end twirling. N1	Uses the simple pendulum as an analogy
---	---

Analogous reasoning was used by the participants to make sense of the forces involved and to try and understand the motion of the ball. Two examples in the text were used. One was a particle (see excerpt given above) and the other a pail, both performing vertical circular motion at the end of a string. None of the experts used any analogous reasoning. Six think- aloud novices and 5 pen and paper novices referred to the examples in the text.

Checking

Checking involved ascertaining the logic of the steps undertaken and the correctness of the mathematics used.

If R equal one metre that would mean h would be two metres long here. If R is one metre, we can have one. The g and the ten will cancel out, seven times twenty five hundred and fifty odd. So, that's a hundred and fifty over R squared. If r is a small ball bearing, then you are getting an enormous number. You could be getting h to be two hundred. In excess of two hundred easily and its not what I would expect, I would expect maybe twenty er four or twenty five R. So, there's some flaw in the approach, I have made a mistake again

E3

Substituting numbers and using intuition as a check

The experts performed checking for qualifying and clarifying the steps as an inherent part of their problem-solving. They used their practical knowledge and experience in their checks. Three experts used dimensions to check the correctness of expressions. The novices generally performed checking when they encountered difficulties or when the expression obtained failed to make sense. They used their intuitive knowledge as a check. An example is given above of a check performed by substituting numerical values and using practical experience and knowledge to determine the possibility of the answer being correct.

Clarifying

Clarifying involved attempting to make sense of the information under current consideration.

I know that's going to be my velocity at the bottom of the loop assuming its rolling but there's got to be some constraint to keep the ball on the loop and that's what I am missing at the moment.

N2

Clarifying problems encountered

Ok so that relates omega to the linear velocity of the ball v, divided by r so the total kinetic energy or the total energy can now be written entirely in terms of its height and its linear velocity at any particular point.

E2

Clarifying work already done

Clarification was done by referring to the text, by verbalising the difficulties encountered and by evaluating the working performed. Examples of the latter two clarifications are given above. The text was used to obtain clarification of formulae and information required. Verbalisation of difficulties was carried out to determine the next step for the solution of the problem. Clarification by evaluating working done up to that stage was used to relate previous working to the present.

Declaring quantities

Declaring a quantity is the process whereby a participant mentioned or used a quantity.

The kinetic energy here does not, at the top of the loop is not zero. It must pass around it and, and the ball is not to leave the track at the top of the loop

E4

Declaring a value

The resultant force on it which is producing that acceleration at the uppermost point is $m g$ plus N

E1

Declaring associated quantities

Declaring involved mentioning a principle or quantity, an equation, an expression using one or more quantities, or merely stating a value for a quantity. Examples of declaring a value and an associated quantity are given in the excerpts quoted above. As can be seen from Table 5.5, this is an activity which is generally performed by all the think-aloud participants.

Pictorial representation

Pictorial representation refers to drawing a diagram and marking information on it.

So if I draw a rough diagram. At a height h there I have a ball with energy $m g h$. It comes down to the bottom of the loop here which is a point I will label A. Goes around the loop. The top of the loop will be a point I will label B and then we will come down again.

N2

Drawing a diagram and marking information on it

This activity was performed by all the experts, 3 think-aloud novices and 6 pen and paper novices. The diagrams drawn by the experts contained precise information on the forces involved and the associated acceleration. Those of the novices had incomplete information of quantities involved and often included markings to indicate

the ball positions. Novices failed to successfully perform the problem solution because they had incorrect or incomplete representations of the information pertaining to problem solution. This is in agreement with the research findings on novices as reported by Larkin (1981).

Resolving difficulties

Resolving difficulties involved trying to correct a mistake and to finding means to perform the next step in the solution to the problem.

I'm not, I am just trying to establish for my own sake whether it is that simple relationship between the angular velocity ω is just, is the velocity just r times ω where r is the radius. When you are looking at circular motion.

Just trying to establish if it is the same case for when the ball is rolling along. I am sure I have done this before so should be able to just.....

E2

Trying to relate quantities and establish relations

I know what I've done with this, stated the obvious, it doesn't give you the least value for h . It gives you the value of h which depends on v

N8

Due to unknown quantities

Resolving difficulties often occurred when participants reached an expression or answer which did not make sense or which they had difficulty in attempting to simplify. The difficulties usually arose due to an unknown quantity appearing in the final expression, a misconception or perception mistake, a mathematical error or wrongly relating quantities. Two examples of attempting to resolve difficulties faced are given above.

Qualitative analysis

Qualitative analysis involved describing the motion of the ball to make sense of the situation.

If the ball were to slide along the track then it wouldn't fluctuate along the sides of the loop. It would come to rest at the bottom of the loop. If it rolls it will depend on the speed of the ball and the radius of the track whether it would go round.

P5

Envisioning using surface features

Right , centre of rest, where it starts from rest got a certain amount of potential energy. At the bottom of the loop that will be converted all to kinetic energy. As it goes back up potential energy again then kinetic energy as it rolls off.

N1

Envisioning using physical entities

Envisioning (the process of qualitatively describing a situation) involved using surface features mentioned in the question, or physical entities, to describe the motion of the ball. Examples of both types of envisioning are given above. One novice and the 3 experts used physical entities to envision the motion. The other 6 novices used surface features or superficial references to physical entities. One expert and 2 novices did not qualitatively describe the motion of the ball. This is similar to the research findings of Chi et al (1981) who reported that novices used surface features while experts used physical entities to categorise physics problems

Qualifying

Qualifying involved making deductions and using supportive information to determine that the solution steps are correct.

There's no friction and its a closed system then you would have to start off at twice R, in order to, but if you start at exactly twice R, the ball, reach the bottom of the loop, start going up the loop, at the top of the loop will recover all the lost potential energy, collect potential energy and it will stop exactly at the top of the loop in which case it will fall down.

Has to be slightly greater than twice R. The second part I will put down for now h minimum got to be greater than two R.

N1

Making deductions

I have assumed conservation of energy. I have assumed that there is no energy lost for rolling without slipping which is never a totally valid assumption but an assumption which you always make in such a situation.

E1

Supportive facts or principles

An example of each type of qualifying action is given in the excerpts quoted above. All the 21 participants performed this to some extent. One think-aloud novice performed this activity excessively taking the wrong solution path several times and experiencing difficulties in making sense of the steps undertaken (N3 in Table 5.5)

Quantitative representations

Quantitative representations included using numbers or algebraic symbols to represent quantities, choosing and writing equations and performing mathematical manipulations.

Okay, if ball were to slide instead of rolling h would have to be equal to two R ,

N2

Representing quantities using numbers or algebraic symbols

It will be equal to mgh equals half mv^2 . m 's cancel so that means v^2 is equal to two gh . Okay now put that into that equation there and we get that gR equals two gh . g 's cancel

N1

Performing manipulations

Two examples of quantitative representations used by the participants are given above. This was an activity which all the 21 participants performed to a large extent (Table 5.5).

Question reading

Question reading involved referring to the question.

A uniform ball of radius r rolls without slipping along the loop the loop track shown in the figure. It starts at rest at height h above the bottom of the loop. Ok, first of all, the height, the first impression I get is, you use the conservation of energy. You are given the initial height so the height above the bottom of the loop. So energy argument is obviously going to be in here somewhere. If the ball is not to leave the track at the top of the loop , what is the least value h can have in terms of the radius R of the loop ? What would h have been if the ball were to slide along a frictionless track instead of rolling ? Rolls without slipping. E2	Reading and processing information
--	--

Three styles of reading the question were used by the participants. Two experts and 8 of the 9 think-aloud novice participants read the question verbatim before commencing the next problem solving step. The other novice read the question in parts and noted information directly provided by the question wording. The other two experts processed the information provided as they read the question. The excerpt quoted above provides an example of this.

Recapitulating

Recapitulating involved pondering over the solution steps which had been performed, trying to link the solution steps, recalling prior knowledge and going back to previous steps to make corrections if necessary.

Okay, what I am going to do now is I know what my, I can write down an equation for the, I'm going to go back to my old handy kinetic energy equation. So I, something that I know is true N2	Recalling prior knowledge
---	---

The experts recapitulated when relating solution steps and when considering the plausibility of the answer. The novices mainly recapitulated when they found an error in their calculations or logic.

Reference

Reference refers to the process of accessing information from the text or from lecture notes, for solving the problem.

Okay we need something like inertia of the ball
Involves its mass (checking book). Inertia, inertia, inertial
mass. Look at that see what happens, three fifty three.
That's inertial mass that's not what I want, rotational inertia,
rotational dynamics, rotation rolling condition, rotation of a
rigid body, two sixty three.

N4

**Using keywords to
obtain information
from the text**

Information in the text was accessed by using keywords as shown in the example above. Two experts and a novice did not make any reference.

Relating quantities

Relating quantities involved using previous experience and knowledge to recall information needed to proceed in the solution of the problem.

Now instead of $m v^2$ we have from the force
relationship that $m v^2$ is $m g (R - r)$
 $m g (R - r) = m v^2$ and $m g R$ and $m g r$

E1

**Using equations
and algebraic
relations**

The knowledge recalled was in the form of equations and algebraic relation.

Symbol usage

Symbols were used to algebraically represent physical quantities used in the problem solution.

Let's do it in a different way and redefine some variables.
Let's call this distance from the top of the loop to where
we're going to release the ball from h .

N3

Assigning symbols

Comment: It should be noted that these activities are inter-related in the problem-solving process. For example when resolving a difficulty, participants might

recapitulate, perform quantitative representations and qualify working being done. Furthermore, there is overlap between the activities. For example, in the process of recapitulating the participant would also relate quantities.

Only the lecturer discussed the real versus the ideal situation for the motion of the ball when checking and clarifying the final expression obtained (see excerpt given below). All the other participants restricted their thinking to the ideal situation. This is not surprising as idealised situations are commonly used in Physics problem solving to simplify the situation.

Nothing occurs to me. Apart from mathematics error I don't see there's anything wrong. I have assumed conservation of energy. I have assumed that there is no energy lost for rolling without slipping, which is never a totally valid assumption but an assumption which you always make in such a situation. There's always some energy loss associated with rolling. If this was a real ball on a real track then it is invalid to assume conservation of energy. There's always going to be some internal frictional energy lost in a rolling ball. It's never absolutely true that energy is conserved in any microscopic situation. It is unlikely that there would be total rolling without slipping. You could arrange it by having smoother surfaces which were not steel on steel in which case you are more likely to have pure rolling without slipping but you are also less likely to have pure conservation of energy. So this is an idealised situation.

The activities highlight the actions which students would wish to perform while problem-solving. The program should enable their performance. The extent to which the features of the program are structured to encompass these activities is discussed in Chapter 7.

Now follows a discussion of the frequencies at which the different activities are performed (Table 5.5).

5.5 GENERAL DISCUSSION OF INFORMATION ON ACTIVITIES PERFORMED

From Table 5.5 it can be gathered that the most frequent activities were: declaring quantities, resolving difficulties, qualifying, quantitative representation, and relating quantities. All except qualifying, involve mathematical representation. Qualifying was

used to make sense and to justify the mathematical representation and the problem-solving steps being used. The excessive use of mathematical representation is expected since it is emphasised in Physics instruction at the university level. A large part of Physics training involves becoming familiar with mathematical representations. This explains the low frequency use of qualitative analysis, analogous reasoning, pictorial representation and reading the question.

The least used activity was analogous reasoning. The two similar examples in the text which the participants could use provided sufficient information and mathematical representation so that participants did not require further reference. Reference which involved obtaining mathematical relations from the text was infrequent, since once mathematical information had been accessed it was applied to perform mathematical manipulations.

The least frequent activities were pictorial representation and symbols. Participants generally drew a single diagram and summarised information on it. Some participants drew more than one diagram when they encountered a difficulty and wished to add more symbolic information. Related symbols were generally represented on the diagram at the same time. Since symbols needed to be assigned beyond those given in the question, the participants moved quickly to perform mathematical manipulations.

The next least frequent activity was reading the question. This could be because once the problem has been read and initially understood, the need to reference it again does not arise. Reinterpreting information in the question was also not engaged frequently. This can again be attributed to the speed with which the students moved to mathematical representation of symbols. Also, participants tended to use diagrams and symbols to represent the information provided by the question which reduced the need to consult the question. Participants tended to refer to the question a second time when they completed the first part of the question and wished to confirm what else the question entailed.

Qualitative analysis was infrequently performed perhaps because it does not entail mathematical representation. Participants tended to qualitatively envision or analyse the motion of the ball when difficulties were encountered with the mathematical representation, or when exploring alternative mathematical representations. Participants preferred quantitative to qualitative reasoning since the solution requires calculations.

Some differences in the frequency of activities performed by experts and novices are now discussed.

5.6 DIFFERENCES BETWEEN EXPERT AND NOVICE USE OF THE ACTIVITIES

Generally, the total number of activities used to solve the problem were much less for the experts than the novices. The expert with the largest frequency of 100 (Table 5.5) took a couple of wrong paths before obtaining a solution. The novice (N3) who was successful had a frequency of 130. Novices with a low total frequency of activities were unsuccessful, and did not persist for long as they were at a loss for ideas. N1, N2, N4 and N9 had higher totals as they explored different paths to the solution.

Quantitative representation, declaring quantities, relating quantities, question reading and symbol usage are necessary activities to attain the solution. There was little difference between experts and novices for these activities, except that the experts related quantities more. The novices had difficulty relating quantities which did not have an obvious relationship, and used symbols to infer similarities and connections between quantities. The experts, on the other hand, used the conceptual meaning of the quantities to relate them.

Apart from the general differences described above the frequency of using an activity (Table 5.5) provides little insight into the differences between experts and novices. In some cases novices performed an activity with a low frequency due to lack of knowledge and in other cases they performed the activity excessively for a similar reason. Some activities differently performed by novices and experts and not obvious from Table 5.5 are briefly discussed below.

Novices referred to examples and analogously deduced information when they encountered difficulties with the problem solution. Examples were used to make sense of the problem under consideration. None of the experts performed analogous reasoning. They knew the information required to solve the problem, and were either lecturing or tutoring the subject. Two experts used the text to confirm a formula. One expert was not sure about the constraint, $N = 0$, but was able to obtain the constraint logically.

As mentioned in an earlier section, the experts constantly checked their working and logic as they progressed. One expert performed more checks because he twice took the wrong path and had difficulty determining the condition $N = 0$. The novices only superficially checked when they obtained a simplified form of an expression or when they were appraising their answer.

Clarification was performed by experts and graduate students N1, N2 and N3 in connecting and relating quantities or when ascertaining the plausibility of their answer.

Other novices used clarifications to determine whether the correct equation was being used by comparing known variables against unknown. Novice N4 (Table 5.5) performed a large number of clarifications because of difficulties in choosing equations. This novice wrote about 7 equations, and performed clarifications of the known and unknown quantities.

All the experts used pictorial representations when they drew detailed force diagrams. Only 3 of the think-aloud novices drew diagrams to show the heights of the ball at different positions. Six of the 8 pen and paper novices drew diagrams, 3 of which were force diagrams.

Next the 14 activities are represent as problem-solving flow charts for the 13 think-aloud participants. These show the sequence in which the problems-solving activities were performed by the participants.

5.7 PROBLEM-SOLVING FLOW CHARTS

Flow charts depict the complexity of the problem-solving process and the idiosyncratic sequence in which activities are performed. There is no linear pattern in which information is accessed and used. Flow charts can be used to:

show the sequential movement of the participant from one activity to the next (Figure 5.6); or

depict the frequency of movement of the participant between any two activities (Figure 5.7).

Both the charts show the number of times an activity was accessed. The total number for each activity may be ascertained by summing the number of times the activity was accessed.

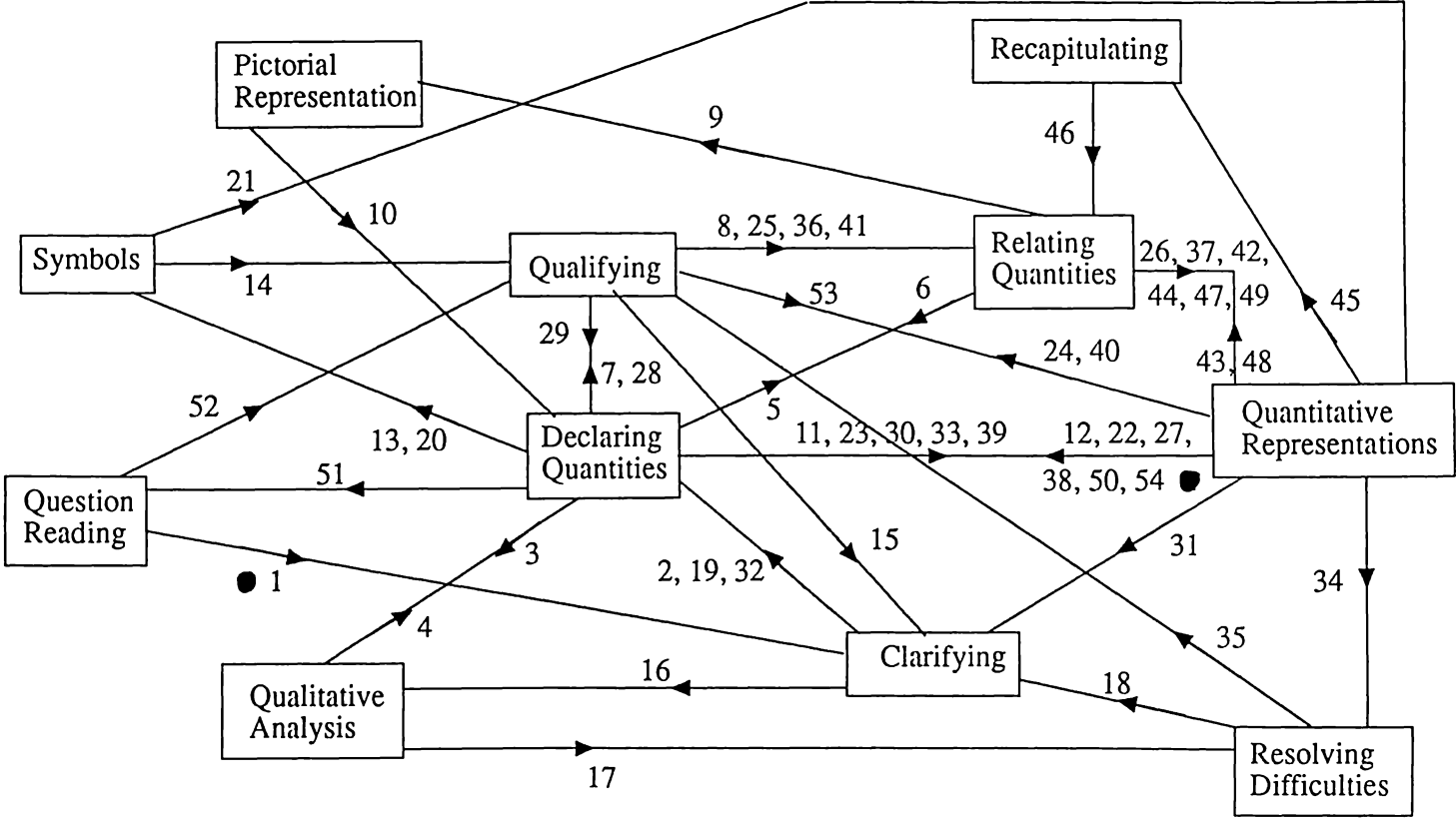


Figure 5.6: The sequential problem-solving flow chart of expert E2.

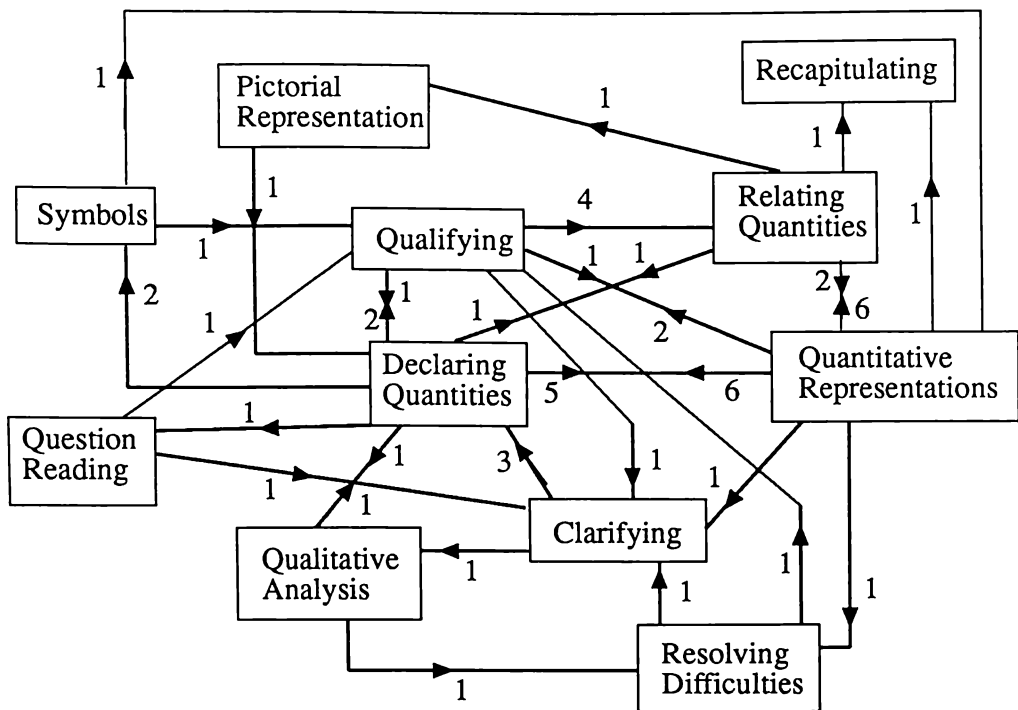


Figure 5.7: The frequency flow chart of expert E1.

The sequential and frequency flow charts for a successful expert (Table 5.6 and 5.7) and two unsuccessful novices (Appendix B2) are provided. The 2 unsuccessful novice examples show greater variety of activity, and are representative of the novices since only one of them was able to successfully solve the problem. The first unsuccessful novice flow chart (N1) represents novices who explored a number of possibilities before declaring that they were at a loss on how to continue. The other novice flow chart (N8) represents participants who tried a few possibilities and were then at a loss on how to continue.

These flow charts highlight that there is no fixed sequence for these activities. Expert E1 (Figure 5.7) declared quantities, performed quantitative representation and related quantities at a higher frequency than the other activities. This may be because he knew how to proceed with the solution of the problem. Novice N1 (Appendix B2) did quantitative representation, qualifying and referred to the text with a higher frequency. This arose due to difficulties encountered and different possibilities being explored. This entailed referring to examples in the text, clarifying the examples to make sense, and performing quantitative representation of the information given in the example. Novice N8 (Appendix B2) on the other hand, spent a lot of time declaring quantities and trying to resolve difficulties encountered. The quantitative representation arose out of attempting to relate and represent the declared quantities.

The flow charts given in Figures 5.6 and 5.7, and Appendix B2 provide insight into the appropriate structure for the program which is the focus of this research. The unfixed nature of accessing information and performing the activities means that a hierarchical order of connecting and representing information will not suffice. The information needs to be accessible at various stages in the problem-solving path so that the user can use it when required and not at a pre-determined point in the program. The random pattern of the problem-solving flow charts also support a constructivist environment, where information is accessed when needed or requested to help make sense of a situation.

Frames and rules as described in Chapter 3 provide suitable means of representing information for these interactions (Chapter 7). In the rest of this chapter, problem-solving behaviour is related to the framework on learning theories, learning environments and problem solving.

5.8 GENERAL AND SPECIFIC PROBLEM-SOLVING STRATEGIES USED

When problem solving, the participants used a variety of general and specific problem-solving strategies (Chapter 2). The participants were not always aware of the strategies they were using, which were inferred from the think-aloud, and pen and paper data collected, supported by excerpts from the transcripts. The strategies used are now discussed in relation to the fourteen activities identified in section 5.4 to show their relevance to program development.

5.8:1 General strategies

Strategies	Think-aloud		Pen and Paper
	Experts	Novices	Novices
Assessing information	4	5	3
Transformation of information	4	9	7
Attempting one-step solution	0	4	3
Generalising information application	1	7	6
Solution assessment	4	4	2

Table 5.6: Number of participants using the different strategies.

Five general strategies: assessing information, transforming information, generalising application of information, attempting one-step solution, and assessing the solution

were used by the participants. Each of these general strategies can be linked to a group of the 14 activities previously discussed. Table 5.6 gives the number of participants whose transcripts were inferred to contain the use of these general strategies.

So, the situation is that the ball is to keep in contact with the track at the top of the loop. Which means that you want the minimum speed that will take it over the top of the loop keeping it in contact with the loop, and one can use the conservation of energy. If the ball falls it starts up with a certain amount of gravitational potential energy. As it falls this becomes kinetic energy of both rotation and translation and since you were told that it rolls without slipping then there is a simple relation between the linear speed and the angular speed at any point. So I would think that the first thing was to do is to look at the situation of the ball as it passes over the top of the loop where the forces that act on it are.....	Assessing information given (determining the conclusions which can be inferred) (assessing to infer the relation between linear and angular quantities) E1
I must convert all this energy into kinetic energy. So, I could write down that my kinetic energy is, I could write down as E kinetic is going to be a half $m v^2$ plus a fifth $m v^2$ squared which I don't like because that is bringing me up with the answer that I am going to end up with the velocity is going to be $m g h$. So, I have done something wrong I know that's going to be my velocity at the bottom of the loop assuming its rolling but there's got to be some constraint to keep the ball on the loop and that's what I am missing at the moment. I thought it would have just been an energy restrain that I have to have enough energy at the top of the loop to do it. Maybe what else could it be?	Assessing information to determine the next step (Realises a mistake has been made) (Assesses the information to know which step to perform next) N2

Assessing information: Assessing information involved use of some or all of the 6 activities: clarifying, pictorial representation, resolving difficulties, qualifying, question

reading and recapitulating. Four experts and 8 novices (5 think-aloud and 3 pen and paper) explicitly demonstrated this general strategy to assess the information given or implied in the question and later to determine the next step in the problem solution. An excerpt is given above for each type of assessment.

Transformation of information: Transformation of information involved the use of some or all 3 activities: pictorial representation, quantitative representation and symbol usage. The information was transferred to diagrams or mathematical symbols. Apart from a pen and paper novice, all the participants transformed information to aid them in solving the problem. The novice who did not transform the information solved the problem qualitatively and intuitively. Examples of excerpts showing the quantitative and pictorial representation are given in section 5.4.

Attempting one-step solution: This strategy was used only by novices (Table 5.6), who expected to obtain an answer by applying an algorithm and substituting values to obtain an answer. Activities performed were quantitative representation (to represent the algorithm) and relating quantities (to make the substitutions). An excerpt giving an example of a one-step solution expectation follows.

We get m times g times h equals a half and we want for ball, m r squared times omega squared. That will be a half I think for the inertia and we can cancel out some mass so we have g h is a half, quarter r squared omega squared Seem to be getting in some mess here, because of the rotational inertia seems to be getting in the way, and we haven't got the radius of the small r either so we can't use that in our formula.	One-step solution expectation (Expects the unknown quantities to cancel giving the solution) N5
--	---

Generalising information application: Four activities: reference, relating quantities, symbol usage and resolving difficulties were performed when generalising information for application. This strategy was applied when participants encountered difficulties or when they recalled solving a similar problem. They referred to the text for similar examples or for familiar relations between physical quantities. These were then applied to the solution of the problem with little discrimination. In some cases, similarity of symbols was used to relate quantities. One expert (E3, who took a few wrong paths before obtaining the solution) and 13 novices (7 think-aloud and 6 pen and paper) used this strategy. Two excerpts showing the general application of information follow.

The speed of its moves, centre of mass is moving is made to, where is that one? Centre of mass is moving. We know how fast centre of mass is moving. Wonder how fast its turning. So, there's a formula for that somewhere. Know how fast the velocity is moving at the edge must know how fast it is turning. What's that one? Where's that one? Oh, of course, it must be equal to omega is two pi r over t over moves two pi in Where they have omega (still searching book). No, where is it? Somewhere there's, I still remember something about an omega being related to velocity somehow. Got to find that formula before I go any further. Its not here. Angular velocity angular, acceleration okay here we are. Angular velocity is given by, v is equal r omega linear velocity, omega. Okay what we want to do here. That looks like it might be it velocity is equal to radius times omega okay omega is equal to v over r, er that's it okay omega equals v over r.	Generally applying information without discrimination (recalls there is relation connecting the linear and angular velocity) (Searching for relations between quantities using symbols). (Obtains the relation but does not check its applicability. Uses it since it has the symbols for velocity and angular velocity in it). N4
--	--

So v squared over R, does that help? That could actually be it actually so that my force will equal m v R which could be m v squared over R and that's the planetary formula. That's the one that this one up here wasn't. That's, the circular motion I had that requirement. So that says that v squared must equal g R So putting that into h is seven tenths v squared plus two g R. h is, no g in that's v squared over g. Okay, putting in that v squared equals g R	Generally applying information by using a relation with similar quantities. (Used $mg = \frac{mv^2}{R}$ for planetary motion even though the equation did not include the reaction N, of the track on the ball.) N2
---	--

Solution assessment: Solution assessment involved use of some or all of 4 activities: checking, clarifying, qualifying and recapitulating during problem-solving or at the completion of the task. As mentioned in section 5.4, the experts checked and clarified the information and solution steps throughout the process while the novices

left this until the end. All the experts performed periodic or final assessment of the solution whereas only 6 novices (4 think-aloud and 2 pen and paper) did so. Two excerpts show this strategy for final and intermediate assessment.

So I think that's all there is to it. If it is sliding then its just a particle. Then can think just in terms of its linear velocity at all points and should just be a simple case of dropping out the energy term associated with its rotational motion and all the previous work we should just carry through. Just omitting that term and that should be the case. Just thinking whether that, no doesn't make a great difference to the final result twenty sevenths tenth to twenty five tenths times R. Its not a big difference. Just wondering if that is, makes sense. I guess, most of its energy is contained in its linear motion and so and so its potential energy, so rotational motion doesn't have too big an effect on the final result. Well, that's it.	Assessing the solution at the end. E2
I know that there is something wrong here because this is leading to an answer where my velocity is zero which is the requirement that I want then it seems to me that h will equal two R which isn't right. Its not taking into account the fact that it is rolling. Which, okay if the if this tells me that, for the part of the question what would h have been if the ball were to slide along a frictionless track instead of rolling. I know then that h would have to be two R. So for part b, h will equal two R.	Solution assessment while performing the solution E3

Next the specific problem-solving strategies employed by the participants are discussed.

5.8:2 Specific problem-solving strategies

The specific problem-solving strategies detailed in Chapter 2 were: generate and test, means-end analysis, problem decomposition, forward strategy, backward strategy, brainstorming, analogy, abstraction, and envisioning. Brainstorming (a group activity) and backward strategy requiring knowledge of the goal state were not used since the problem solving was individually performed and the answer was not known. Examples

of the use of each of the others by the participants are now provided through excerpts from the transcripts or general problem-solving concept maps.

Generate and test: The problem solver uses this strategy by generating possible solutions one at a time and testing their applicability. One expert (E3) and 7 novices (5 think-aloud and 2 pen and paper) used this strategy. The expert used this when a wrong path was taken. The novices wrote different equations or relations in their attempts to connect the given information to unknown quantities. When an equation did not provide the needed connection they went on to use another and so on until they made a connection or ceased performing the solution. The excerpt below shows the equations generated by novice N2 until the step leading to the solution was accidentally stumbled upon.

At point A I can write down that the, I will have lost all my potential energy assuming that it is zero at that point so that I can write that the kinetic energy at that point is going to be a half m v squared plus its kinetic angular part which is going to be a half I omega squared. which is seven tenths m v squared..... So, at the bottom of the loop I know that m g h at the bottom of the loop m g h must equal seven tenths m v squared. Cancelling the m's out. So I can write v squared equals ten sevenths g h. At top of the loop, the total energy at the top of loop say point B, the total energy so that m g h minus my potential energy there so the difference in that which is just going to be m g two R must be equal to its kinetic energy which I have previously calculated to be seven tenths m v squared at all points. at the top of that loop let's write down something like, at the bottom of A I know that v at A is going to be the square root of ten over seven g h. Okay, I omega that was the, I omega that's the constraint that, so that's m g times the height of the loop. Its got to equal m v big R and its got to equal the where was it I had it on this diagram m g R, m g times two R. So the m's cancel out. The R's cancel out so my velocity must equal two g. Now that doesn't seem right either.	(Writes the expression for the total energy at the bottom of the loop) $E = \frac{1}{2}mv^2 + \frac{1}{2}I\omega^2$ (simplifies it) (Equates energy at the bottom and start) $mgh = \frac{7}{10}mv^2$ (Trying to relate R to the velocity and makes a mistake) $mgh - mg2R = \frac{7}{10}mv^2$ (searching for a substitution for v) $mgH = I\omega$ (writes another expression to obtain v) $mvR = mg2R$ $v = 2g$
---	---

Okay, how do I want angular momentum..... planets generally mgh is a half mv^2R where R is the big R in our case. I'm looking, what this stationary planet's formula to get a orbit in for a planet. I must have that was the one. So I might be able to use the force in this case. So, the force is gravity. In this case force will equal, no that's energy, force is mass times gravity Let's, I can relate that to an angular momentum system can't I, $m v R$. What's it for an angular system will be er because what I am probably looking for is the torque, angular momentum type rule. The torque could be something like, $m g$ will equal m times v times R. So v, v will be g over R. That could actually be it actually so that my force will equal $m v R$ which could be mv^2 over R and that's the planetary formula. That's the one that this one up here wasn't. That's, that's the circular motion I had that requirement. So that says that v^2 must equal gR	(writes another expression to obtain v) $mgh = \frac{1}{2}mv^2R$ (Considering another expression) $mg = \frac{mv^2}{R}$ (now considers angular momentum and weight) $mg = mvR$ $v = \frac{g}{R}$ (stumbles on this expression) $mg = \frac{mv^2}{R}$ (Hence, obtains the velocity expression) $v^2 = gR$ N2
--	--

Means-end analysis: In this strategy the difference between the current solution state and the goal state is first noted, then attempts made to reduce the difference. This strategy was used when participants were unsure of the steps leading to the solution. One expert (N2) and 10 novices (6 think-aloud and 4 pen and paper) used this strategy when attempting to simplify expressions, and to eliminate unknown quantities through substitutions. The excerpt given below shows the application of this strategy by a novice.

Now rotational kinetic energy is a half times I which is that times omega squared. Omega is, now half times two over five times little m times little r squared times omega squared. Omega is angular velocity. Angular velocity is two pi times r. No its not. Omega is equal to what's it, two pi r is the diameter. Two pi times r divided by t, how does that look. Okay I might just leave it omega at the moment, it might cancel itself out..... Got to find that formula before I go any further. Its not here, angular velocity angular, acceleration okay here we are. Angular velocity is given by, v is equal r omega linear velocity, omega. Okay what we want to do here. That looks like it might be it velocity is equal to radius times omega okay omega is equal to v over r, er that's it okay omega equals v over r.	(Wrote the rotational energy equation and substituted for I, the moment of inertia) (Tried to substitute for ω so that it could be combined with the translational kinetic energy expression and simplified) (Searching for substitutions to eliminate ω so that the expression could be simplified) N4
---	---

Problem decomposition: This strategy involves breaking a problem into smaller sub-problems. Each sub-problem is then solved and combined to form the solution to the original problem.

So I would think that the first thing is to look at the situation of the ball as it passes over the top of the loop where the forces that act on it are That's the relationship that exists for that force diagram at the top of the loop. Now in terms of energy its initial potential energy is that. Once at the top of the hoop it is going to have Now instead of m v squared we have from the force relationship that m v squared is m g	(Creates a sub problem of considering the force equation) (Considers the energy equation) (Combines the two) E1
--	---

All the experts and 5 novices (4 think-aloud and 1 pen and paper) decomposed the problem into sub-parts. They solved for the velocity of the ball by applying the energy principle to the motion of the ball at two different positions. They then solved for the

velocity by writing the force equations at the top of the loop. These sub-problems were then combined to solve for the starting height of the ball. The excerpt above from the transcript of E1 shows the solution being solved through decomposition.

Forward strategy: This strategy involves using the information given in the question to infer the steps required to solve the problem. It involves working from the known quantities to the goal state. It involves a lot of general information but it is specific in its directional movement towards the goal. Three experts and 2 novices (1 think-aloud and 1 pen and paper) portrayed this strategy. The think-aloud novice (N3) commenced the solution attempt with this strategy but later reverted to some other strategy when difficulties were encountered. A concept map of the problem-solving steps used by E2 (Figure 5.8) is used to show the use of this strategy.

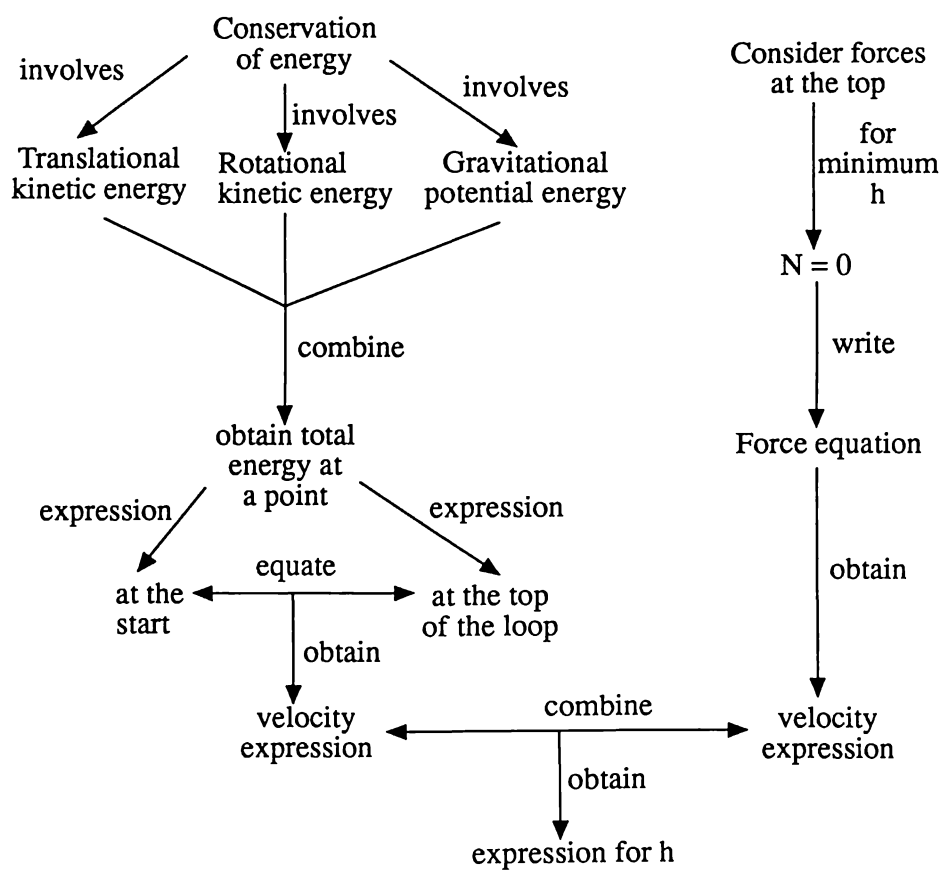


Figure 5.8: Problem solution concept map of E2.

Analogy: This strategy involves comparing the problem to another example. To be successful it requires recognition of the differences and similarities between the example and the problem. The participants who used this strategy concentrated on the

similarities and ignored the differences. As stated in section 5.4, 6 think-aloud and 5 pen and paper novices used this strategy. For an example of an excerpt illustrating an analogy strategy refer to section 5.4.

Envisioning: This strategy involves visualising what will happen when the ball moves to predict how the motion will take place or change at different positions. An understanding can only be fully obtained when envisioning uses physical entities. As shown in section 5.4, one novice and 3 experts envisioned using physical entities. The 3 experts who envisioned in detail, then went on to solve the problem using the forward strategy (Figure 5.6).

Comments: The general and specific problem-solving strategies highlight how the participants explore and access information for the solution of the problem. Most of the participants used a combination of strategies.

This analysis of the problem-solving strategies was used to inform the representation of knowledge in the program. The knowledge needs to be represented

- in descriptive form, to cater for envisioning.
- as basic relations, to enable means-end analysis to be employed.
- in diagrammatic form, to enable transformation of information and help obtain a total picture.
- with the applicability conditions of the knowledge, to enable participants to generate and test.
- separately, to enable problem decomposition.
- choice of variable values, to enable simulations for envisioning and solution assessment

Next the problem-solving perceptions and expectations of students are highlighted.

5.9 PROBLEM-SOLVING PERCEPTIONS AND EXPECTATIONS

When students are given problems to solve they have expectations about the process required, about why they should solve the problem, and about how they can benefit from it. These expectations will influence their interaction with a problem-solving

program. In the interviews students were asked why they did problem-solving in Physics. Their reasons were that they would like to:

- obtain a better understanding of the subject.
- progress from simple to more difficult problems.
- meet the challenge of being able to do the problems.
- make connections and sense.
- increase their confidence in handling the subject matter.
- know how to apply theory to problems.
- complete assignments.
- be able to solve and present a solution to a problem.
- find out what they don't know or need to know.
- learn how to problem solve.

These expectations show that students perceive problem-solving as a beneficial and worthwhile activity. This suggests that if a program is developed to help them solve problems they will interact with it. Next the learning theories implied by the transcripts and problem-solving styles of the participants are discussed.

5.10 THE LEARNING ENVIRONMENT

In this section the way participants used and accessed information is used to determine the learning environment appropriate to the problem-solving process. The discussion supports the premise that the program developed should be based on the generative learning theory using the constructivist learning environment with the use of transmission to supply information.

The random and idiosyncratic way in which the fourteen activities were performed, points to a generative model of learning. Knowledge accessed and the way it was used was governed by the prior knowledge of the problem solver.

The experts used a forward strategy which implies large well organised schemata of knowledge. The generate and test, and means-end analysis methods used by the novices suggest fragments of independent schemata which have fewer connections. The experts were using prior knowledge and experience to solve the problem, while novices on the other hand were using prior knowledge and building on it in trying to make sense while applying the knowledge.

The discussion shows that Ausubel's discovery learning, Kelly's schema theory and the generative learning model could be used to describe the problem solving of the

participants. The generative model encompasses discovery learning and supports organisation of knowledge using schema. When problem solvers sought information to make progress, checked out understanding of parts of their procedure, and referred to previous knowledge in an idiosyncratic manner they revealed the need for a flexible constructivist learning environment. Transmission of knowledge has to be incorporated in this environment so that students can access knowledge which they need. The information should be supplied in small doses using canned text rather than in large blocks. The requirement for information to be available when needed means that the program has to be structured such that information can be accessed at different stages of the problem-solving process. This provides justification for the use of a constructivist environment with transmission of information built into the program structure as proposed in Chapter 1. The structure of the program is explained in detail in Chapter 7.

5.11 CONCLUSION

In this chapter, the think-aloud, and pen and paper transcripts have been used to justify the selection and modification of the loop-the loop problem (Figure 4.3). The analysis has been used to determine the steps performed by the participants when commencing the problem solving. It has been found that experts processed information as they read the question and that they qualitatively analysed the motion of the ball to obtain a total picture. Novices on the other hand did not process information while reading the question and went on to directly write mathematical equations in attempting to solve the problem. The information and assistance sought by the participants to obtain clarification and relations between quantities has been outlined. Further information used by the participants has been determined. The general mistakes, conceptions and difficulties encountered have been used to compile the declarative and procedural knowledge required to help problem solving. The major difficulties encountered by the participants were in visualising the motion of the ball, using the energy principle, writing the force equation, applying the constraint, and in relating the information obtained from the energy and force equation. The knowledge needed to solve the problem was compared to that called on by the participants to determine the knowledge which needs to be supplied in a problem-solving program.

Fourteen activities have been identified to describe the problem-solving processes used by the participants. The problem-solving flow charts of the participants demonstrate the random and idiosyncratic way in which the activities are performed. The activities have been used to identify general problem solving strategies and some specific

strategies have been discussed to show how the information for the problem solution is used and accessed.

The problem-solving activities performed have been used to argue that constructivism with a certain amount of transmission would provide a suitable environment for a problem-solving program.

In the next chapter, the written feedback obtained on the solution of the problem in phase two of the research is presented. The solution written by an expert, by teachers and a general and alternative solution to the problem are presented.

CHAPTER 6

THE PROBLEM SOLUTION

6.1 INTRODUCTION

In this chapter, the solutions to the problem obtained through written feedback in the second phase of the research (see chapter 4) are presented. An expert and four teachers were requested to write the solutions. The expert's transcript provides the information assumptions and steps used by experts in solving the problem. The teachers' transcripts provide the information generally presented to explain the problem solution and address the difficulties encountered by students. The teachers solutions documents the verbal and written information used to present the solution to students. The expert and two teaching solutions are presented. Two teachers' solutions are presented to show the difference in approach in explaining the steps leading to the solution of the problem.

Next a comprehensive solution to the problem is presented. This was compiled to ensure that relevant knowledge not uncovered by the think-aloud and pen and paper sessions was identified. It is also used to highlight the vast amount of related knowledge which needs to be considered if the problem is solved using only the fundamental mechanics concepts of mass, length and time. Through this, an alternative solution path not performed by any of the participants was developed. Documenting this knowledge provides information on connections with other topics in Physics and possible extensions for teaching and learning related topics using the solution of the problem.

6.2 THE EXPERT'S SOLUTION

The solution given in this section is an example of the solution an expert (lecturer) would perform to present the main solution steps including the explanations provided.

This solution provides the minimum information required to understand the solution, assuming an understanding of the necessary related knowledge. The solution steps fit the forward problem-solving strategy generally used by experts.

Solution

The conservation of energy can be used as there is no energy lost due to frictional effects. The ball starts from rest with a certain gravitational potential energy and at a

height h above the lowest position of the ball as shown in the diagram given (Figure 4.3). As it slides along the smooth frictionless track, part of this becomes translational kinetic energy.

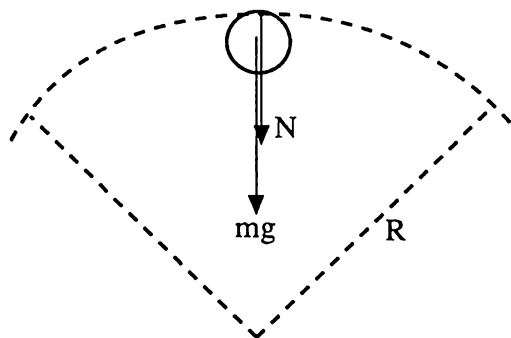


Figure 6.1: Force diagram for the top of the loop.

As the ball passes over the top of the loop the forces that act on it are the downward force of gravity. If it is moving with sufficient speed and stays in contact with the track there is a reaction force due to the track. The situation required is where this is equal to zero at the top of the loop. This will provide the minimum starting height of the ball such that it will perform the loop successfully.

The radial acceleration is given by

$$a = \frac{v^2}{R}$$

The resultant force on the ball which produces the radial acceleration at the top of the loop is

$$mg + N$$

but $N = 0$ for the minimum speed of the ball.

This gives

$$mg = \frac{mv^2}{R}$$

$$v^2 = gR$$

Take the zero potential energy reference level at the lowest position of the centre of mass of the ball. Considering the energy at the initial point and the top of the loop

$$mgh = mg(2R) + \frac{1}{2}mv^2$$

$$= 2mgR + \frac{1}{2} mgR$$

Therefore,

$$h = \frac{5}{2} R$$

If the ball rolls without sliding then as it starts moving the initial gravitational potential energy is converted to both kinetic energy of translation and rotation.

Considering the energy at the initial point and the top of the loop

$$mgh = mg(2R) + \frac{1}{2} mv^2 + \frac{1}{2} I\omega^2$$

For a rolling ball,

$$I = \frac{2}{5} mr^2$$

where r is the radius of the ball and

$$v = \omega r$$

Therefore,

$$\begin{aligned} mgh &= mg(2R) + \frac{1}{2} mv^2 + \frac{1}{2} \frac{2}{5} mr^2 \left(\frac{v}{r}\right)^2 \\ &= 2mgR + \frac{7}{10} mv^2 \\ &= 2mgR + \frac{7}{10} mgR \end{aligned}$$

Therefore,

$$h = \frac{27}{10} R$$

Next, 2 of the 4 teachers' solutions are presented.

6.3 THE TEACHERS SOLUTIONS

The solutions of the 4 teachers may be classified into 2 styles. Both styles are presented below. Three teachers stated that they would present the diagrams and the main solution steps on transparencies using an overhead projector (Solution B). One teacher preferred drawing the diagram and writing the solution on the board as it was performed (Solution A). All four teachers stated that they would verbalise the written

information as well while explaining the solution. The two solutions presented provide examples of the variation in information which may be presented to students.

Solution A is first presented with the information which would be written on the board in italics. Next an example of a solution presentation using transparencies is provided (see solution B below).

Solution A

Let us make sure we understand the question. The ball is stationary at a height h above the bench and then slides down the track. What else do we know about the ball when it is at height h ?

It has

$$\textit{potential energy at the start} = mgh$$

Since h is the height of the centre of mass at the start. What happens to that potential energy after it is released? It is changed gradually into kinetic energy as it slides down the ramp.

Is there any other form of energy such as that due to friction? No, because the loop is frictionless.

So

$$\textit{K. E. at the bottom of the loop} = \textit{P. E. lost in falling through height } h$$

What happens to this kinetic energy? Some is changed into potential energy as it rises round the loop but the ball also has some kinetic energy which stops it from leaving the track at the top. How much kinetic energy does it need to just not leave the track ? How can we calculate this ?

$$K.E. = \frac{1}{2}mv^2$$

So we need the speed. What do we know about motion in a circle which will help us find the speed ?

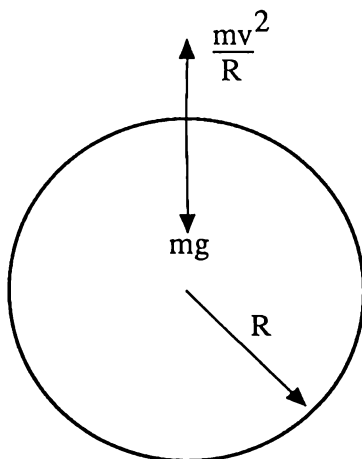


Figure 6.2: Diagram drawn to show the forces acting on the ball at the top of the loop.

$$\text{Centrifugal force} = \frac{mv^2}{R}$$

and this is just balanced by the weight.

$$\therefore \frac{mv^2}{R} = mg$$

$$v^2 = gR$$

$$\begin{aligned} \therefore \text{K.E. at the top of the loop} &= \frac{1}{2}mv^2 \\ &= \frac{1}{2}mgR \end{aligned}$$

and

$$\text{PE at the top of the loop} = mg2R$$

Since height above the bench is twice the radius of the loop.

$$\begin{aligned} \therefore \text{Total energy at the top} &= \frac{1}{2}mgR + 2mgR \\ &= 2.5mgR \end{aligned}$$

This energy = total PE lost by ball

$$mgh = 2.5mgR$$

$$h = 2.5R$$

If there is rolling but no slipping the ball has rotational energy at the top of the loop as well as kinetic energy and potential energy. So

$$\begin{aligned} \text{Total energy at top of loop} &= K.E. + P.E. + \text{rotational energy} \\ &= \frac{1}{2}mv^2 + 2mgR + \frac{1}{2}I\omega^2 \end{aligned}$$

$$\text{Moment of inertia of a sphere} = \frac{2}{5}mr^2$$

and

$$\begin{aligned} \omega^2 &= \frac{v^2}{r^2} \\ &= \frac{gR}{r^2} \quad (\text{Since } v^2 = gR) \end{aligned}$$

since the speed of the ball is the speed round the loop.

$$\begin{aligned} \therefore \text{rotational energy} &= \frac{12}{2 \cdot 5}mr^2 \frac{v^2}{r^2} \\ &= \frac{1}{5}mv^2 \\ &= \frac{1}{5}mgR \quad (\text{Since } v^2 = gR) \end{aligned}$$

$\therefore$ Initial PE = total energy at top of loop

$$mgh = \frac{1}{2}mgR + 2mgR + \frac{1}{5}mgR$$

Dividing through by mg

$$h = \frac{R}{2} + 2R + \frac{R}{5}$$

$$h = 2.7R$$

Solution B

(In this solution the information which would be presented using transparencies is given in Figures 6.3 and 6.4)

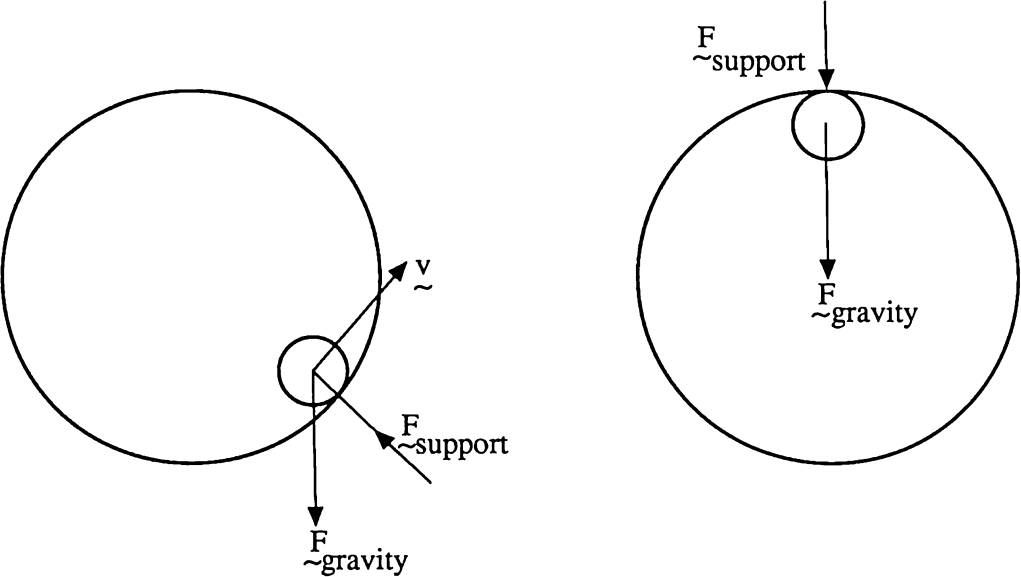


Figure 6.3: Forces at a general point and at the top of the loop.

$$mg = \frac{mv^2}{R}$$

$$v^2 = gR$$

$$\Delta E_p \rightarrow \Delta E_k$$

$$mg(h - 2R) = \frac{1}{2} mv^2$$

$$mg(h - 2R) = \frac{1}{2} mgR$$

$$2h - 4R = R$$

$$2h = 5R$$

$$h = 2.5R$$

$$mg(h - 2R) = \frac{1}{2} mv^2 + \frac{1}{2} I\omega^2$$

$$= \frac{1}{2} mv^2 + \frac{1}{2} \times \frac{2}{5} mr^2 \times \frac{v^2}{r^2} \quad (v = \omega r)$$

$$= \left(\frac{1}{2} + \frac{2}{10}\right) mv^2$$

$$mg(h - 2R) = \frac{7}{10} mgR$$

$$h = \frac{27}{10} R$$

$$= 2.7R$$

Figure 6.4: Solution steps on a transparency.

Read the question carefully. Consider the motion of the ball on a frictionless track for various heights (referring to the diagram given in the question). In moving from initial position to bottom

$$\Delta E_p \rightarrow \Delta E_k$$

As ball moves up the loop, it slows down.

$$\Delta E_p \rightarrow \Delta E_k$$

It undergoes circular motion but not at a uniform speed.

If the ball is moving too slowly it will lose contact with the track. At one particular speed the ball will just remain in contact with the track at the top of the loop. Above its speed the ball will *easily* remain in contact with the track (as long as the loop is strong enough to withstand the forces applied to it)

The speed at which the ball will just remain in contact with the track should supply us with the solution using further analysis. What are the forces acting on the ball in the loop?

F_{support} is the force supplied by the track which changes the direction of the ball keeping it in circular motion. F_{support} is continually changing as the ball's position and speed changes.

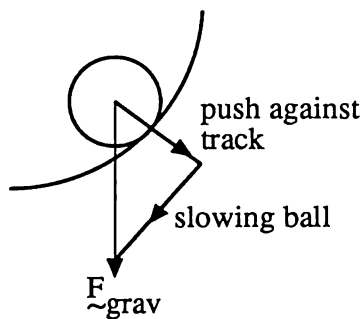


Figure 6.5: Diagram showing the 2 components of F_{grav} .

The effect of F_{grav} on the moving ball is a combination of slowing it down and pushing it against the track. Consider the situation when the ball is just in contact with the track at the top of the loop but not really pushing against it. Then $F_{\text{support}} = 0$. The central force (F_c) which maintains the ball in circular motion is supplied by gravity. (i.e. mg). Note as speed of the ball increases at this point F_{support} must also increase in magnitude.

$$F_c = F_{\text{grav}} \Rightarrow \frac{mv^2}{R} = mg$$

$$v^2 = gR$$

At top of the loop:

$$\Delta E_p \rightarrow \Delta E_k \quad \Rightarrow \quad mg(h - 2R) = \frac{1}{2}mv^2$$

Substituting

$$v^2 = gR$$

$$mg(h - 2R) = \frac{1}{2}mgR$$

$$2h - 4R = R$$

$$2h = 5R$$

$$\mathbf{h = 2.5R}$$

This is the minimum take-off height.

Any questions?

For the second part of the question when the ball rolls without slipping (due to friction of the track) some of the E_p is converted into rotational E_k . Thus, a greater height will be necessary to keep the ball in contact with top of the loop. Minimum speed squared at the top of the loop is still

$$v^2 = gR$$

But this time

$$\Delta E_p \rightarrow \Delta E_k + \Delta E_{\text{rotational}}$$

$$\Rightarrow mg(h - 2R) = \frac{1}{2}mv^2 + \frac{1}{2}I\omega^2$$

$$= \frac{1}{2}mv^2 + \frac{1}{2} \times \frac{2}{5}mr^2 \times \left(\frac{v}{r}\right)^2$$

(for a solid sphere, $I = \frac{2}{5}mr^2$ and $v = \omega r$)

$$= \left(\frac{1}{2} + \frac{1}{5}\right)mv^2$$

$$mg(h - 2R) = \frac{7}{10}mv^2$$

$$mg(h - 2R) = \frac{7}{10}mgR$$

$$h = \frac{27}{10}R$$

$$\mathbf{h = 2.7R}$$

This is the minimum take-off height for the rolling ball. Note this is greater than in the first situation as some of the potential energy is converted to rotational energy.

Comments:

1. In both solutions the teachers used a similar strategy. First they referred to the question and obtained information from it. In solution A, the teacher then followed with a qualitative analysis of the energies involved as the ball moved and wrote the energy equation. The forces were then considered and the force equation written. Finally, the two equations were related and the value for the starting height obtained. In solution B, the only difference was that the force equation was considered first.
2. $v = \omega r$ was mentioned by all the 4 teachers but no explanation for its applicability in the rolling case was forwarded. It was assumed that this would have been explained during other teaching sessions.
3. $I = \frac{2}{5}mr^2$ was substituted as the moment of inertia of the ball. Apart from mentioning that it was the expression for a solid sphere, no other qualification of its applicability was made. The fact that it was for an axis through the centre of the sphere was not mentioned. It was assumed that the students would know this or it would have been explained during other teaching sessions.
4. The two solutions used a different approach in considering the forces involved. In solution A, the reaction force of the track on the ball was totally left out. The teacher wished to simplify the solution to the problem so that students did not have to consider information not directly leading to the final answer. In doing so no reference was made to the reaction force. This might lead to the assumption by students that there is no reaction force present. In solution B, the teacher considered the forces involved at a general point prior to considering the top of the loop. This approach provides the students with insight into the different forces involved.
5. Different explanations were forwarded when writing the force equation

$$\frac{mv^2}{R} = mg$$

In solution A, the concept of centrifugal force was used. However no reference was made to the direction of the acceleration. This could mislead the students into thinking that the acceleration acts away from the centre of the circle rather than towards it. This explanation uses the idea of opposite forces balancing to equate. This implies equilibrium and can again mislead the students into assuming that there is no acceleration. In solution B, the concept of a central acceleration towards the

centre of the circle to support the central force was used. However, in forming the equation the teacher mentioned that the gravitational force supports the acceleration. This gives the impression that the gravitational force makes the ball perform the loop. In fact it is the gravitational force that causes the ball to leave the track, if its velocity is not large enough. The acceleration towards the centre supports the gravitational force and not vice-versa. These diverse approaches used in forming the force equation need to be addressed. They provide a constant source of confusion amongst students if not carefully explained.

6. In solution A, the students have been left to deduce that the force equation remains the same. The force equation used for sliding has been applied for rolling without any comments or qualifications. In solution B, the teacher made a passing mention that the expression $v^2 = gR$ is still applicable but no justification was provided. The teachers later reiterated that this would be addressed when questions were asked. They assumed that if the students did not understand, then queries on its use would be voiced.
7. In solution A, the frictional force for the rolling ball was not mentioned. In solution B, it was pointed out that the ball rolled because of friction. It was assumed that students knew that was the only effect of the frictional force and would ask if they did not understand.

Educators have to assume some prior knowledge, their difficulty is in determining the nature of this. Without making this assumption the solution of each problem would be lengthy and time consuming. For the purpose of this research it is valuable to present a solution encompassing all the related knowledge.

6.4 THE COMPLETE SOLUTION

A solution of the problem was compiled using only the fundamental mechanics concepts of time, mass and displacement, and deriving all other physical quantities and relationships. (Details of these derivations referenced in the solution are found in Appendix C.) The solution shows the complexity of such an approach to presenting a solution. However, the related information suggests the richness of such a process for teaching and learning, and for making connections to other knowledge. The complete solution follows.

Solution

Sliding: For sliding, the track is smooth which means there is no energy lost due to frictional effects. The ball starts from rest at a height h , with a gravitational potential energy of mgh . The zero reference level for the gravitational potential energy is chosen at the lowest position of the centre of mass of the ball during its motion. As it moves down the track, part of this energy becomes translational kinetic energy. Since it slides, the ball can be treated as a particle or point mass.

Force considerations: In the loop, the forces that act on the ball are the downward force of gravity (mg) and a normal reaction force (N) towards the centre of the loop due to the track. These are shown in figure 6.6 below.

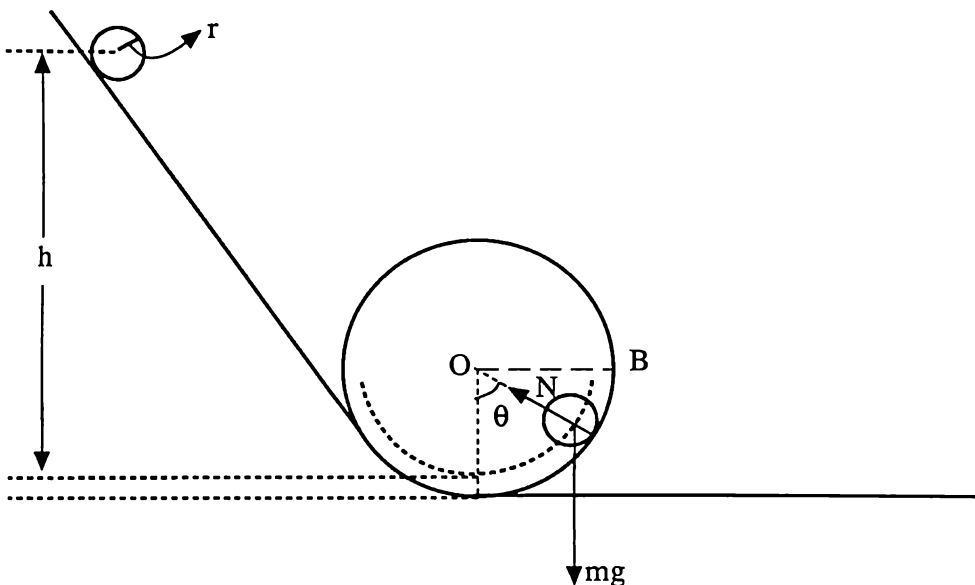


Figure 6.6: The forces acting on the ball at a general point in the loop.

The reaction force acts on the ball only if it is in contact with the track. This will definitely be the case until the point marked B . Above B , for the ball to remain in contact with the track it must start from a sufficient height, h . This will ensure that it has sufficient speed due to the initial potential energy being converted into kinetic energy. This speed will have a minimum value at the top of the loop. Therefore, if the ball remains in contact with the track at the top of the loop then it will not leave the track at any other point of its motion.

For an object in vertical circular motion, the direction as well as the magnitude of its velocity changes with time. The change in direction of the velocity is produced by a net force towards the centre of the circle. The acceleration (a_r) due to this acts radially towards the centre of the circle. The change in magnitude of the velocity is caused by a

net force acting in the tangential direction producing an acceleration (a_n) in that direction as well. The directions of the accelerations are shown in Figure 6.7. The forces acting on the loop can be resolved along the normal (or radial direction) and tangent to the circle. These directions are shown in Figure 6.7.

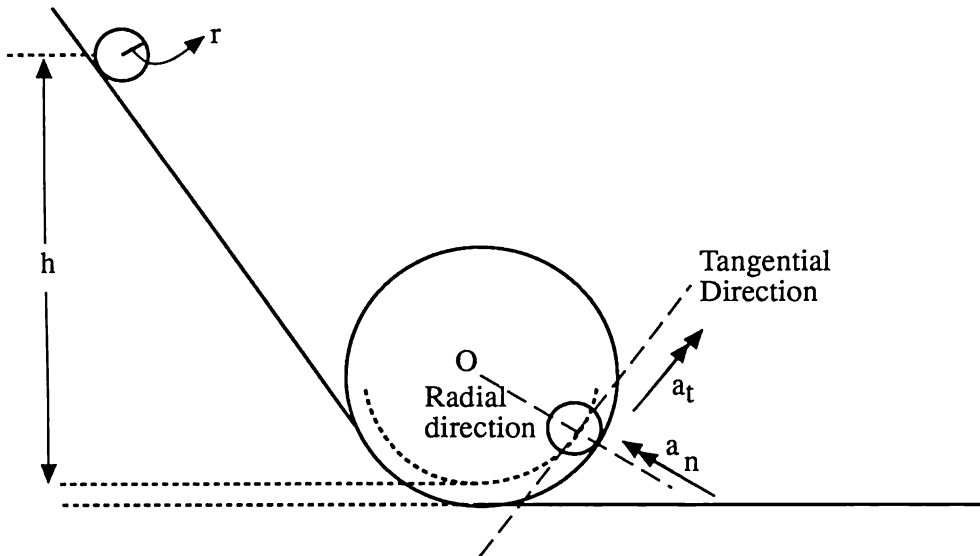


Figure 6.7: The radial (normal) and tangential directions.

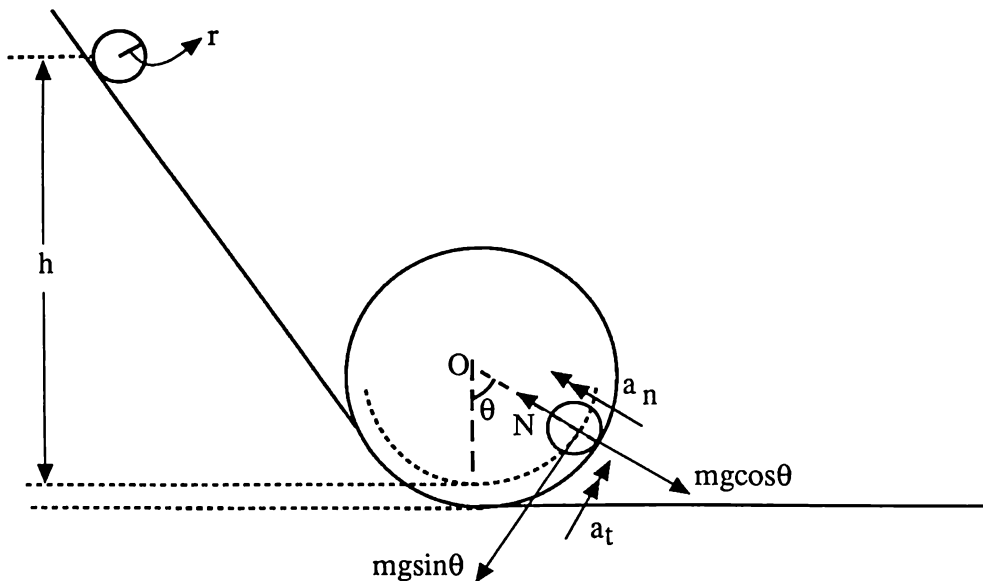


Figure 6.8: The resolved forces and the tangential and radial accelerations.

Along the radial direction:

Defining the positive direction as towards the centre of the circle.

$$N - mg\cos\theta = ma_n \quad (\text{Appendix C1.2 for resolution of vectors})$$

where a_n is the acceleration of the ball towards the centre of the circle.

Figure 6.8 shows the resolved forces and the direction of the tangential (a_t) and radial (a_n) accelerations. Newton's second law of motion for the case of constant mass

$$F = ma \quad (\text{Appendix C2})$$

can be used to write equations of motion for each of the two mutually perpendicular directions in which the forces have been resolved.

F is the resultant force acting on the object along a direction used for resolution;
 a is the acceleration of the object in that direction; and
 m is the constant mass of the object.

Along the normal or radial direction

$$F = N - mg\cos\theta$$

gives the resultant force. Therefore,

$$N - mg\cos\theta = ma_n$$

where a_n is the acceleration in the radial direction

For any circular motion, the radial acceleration towards the centre is given by

$$a_n = \frac{v^2}{R} \quad (\text{Appendix C4})$$

where v is the tangential speed.

This acceleration (a_n) acts towards the centre of the circle at all times and has a maximum value at the lowest point in the loop when the velocity is a maximum, and minimum at the top of the loop when the velocity will be a minimum.

Therefore,

$$N - mg\cos\theta = \frac{mv^2}{R}$$

Note that as the ball moves past B (Figure 6.9), θ increases beyond 90° or $\frac{\pi}{2}$ radians, so $\cos\theta$ changes sign (Appendix C1.3)

The component of gravity is now in the same direction as N , as shown in the diagram below.

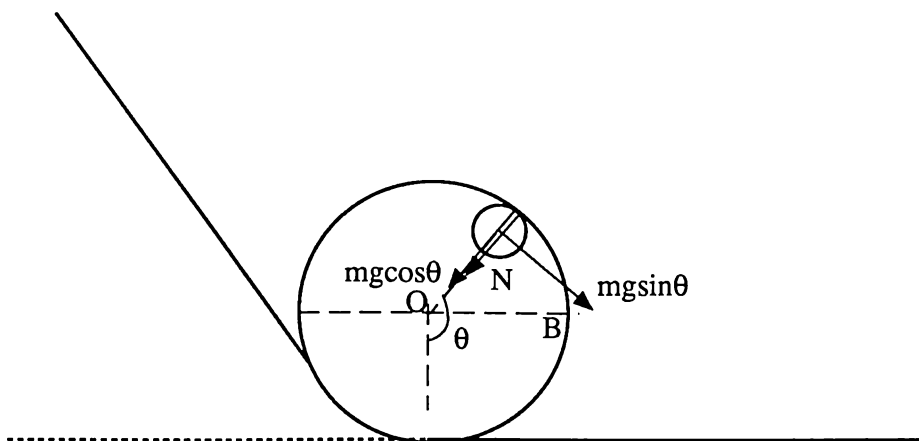


Figure 6.9: The direction of the forces in the top right quadrant of the loop.

At the top of the loop,

$$\theta = 180^\circ \text{ or } \pi \text{ radians.}$$

and

$$\cos \theta = \cos 180^\circ = -1 \quad (\text{Appendix C1.3})$$

Then

$$N - mg \cos 180^\circ = \frac{mv^2}{R}$$

$$N + mg = \frac{mv^2}{R}$$

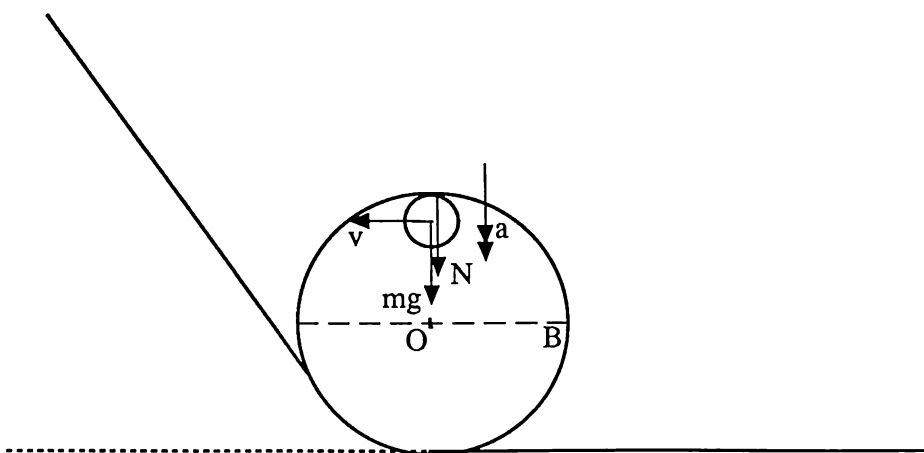


Figure 6.10: The forces at the top of the loop.

At the top of the loop the ball has still a tangential velocity (Figure 6.10). This velocity needs to be sufficient for the ball to go round the loop without losing contact with the track. There is no force in the tangential direction at this instant in time. Note that mg and N now act together to provide the required acceleration towards the centre. The limiting case would be when mg acting alone provides the acceleration i.e. N tends to

zero ($N \rightarrow 0$) indicating the ball momentarily loses contact at the top of the loop, and continues tangentially, immediately regaining contact with the loop. This is the constraint required to obtain the minimum value of h .

For the ball to remain in contact with the track

$$N \geq 0$$

At the top of the loop, for the limiting case $N \rightarrow 0$. So

$$\frac{mv^2}{R} = mg$$

$$v^2 = Rg$$

$$v = \sqrt{Rg}$$

This gives the minimum speed to complete the loop, and therefore the minimum kinetic energy at the top is

$$\begin{aligned} \frac{1}{2}mv_{\min}^2 & \quad (\text{Appendix C5}) \\ &= \frac{1}{2}mRg \end{aligned}$$

Energy considerations: As stated earlier, the energies involved are gravitational potential energy and kinetic energy. Since there is no energy conversion to other forms the total energy of the ball at any instant will be constant. It is assumed that the ball slides without friction. The lowest position of the ball is taken as the zero potential energy level.

The initial gravitational potential energy of the ball

$$= mgh \quad (\text{Appendix C6})$$

At the top of the loop:

The gravitational potential energy of the ball

$$= mg(2R)$$

$$= 2mgR$$

The translational kinetic energy of the ball

$$= \frac{1}{2}mv^2$$

The total energy of the ball

$$= 2mgR + \frac{1}{2}mv^2$$

Initial energy = Energy at the top of the loop

$$mgh = 2mgR + \frac{1}{2}mv^2$$

Since

$$v^2 = Rg$$

$$mgh = 2mgR + \frac{1}{2}mgR$$

Therefore,

$$h = \frac{5}{2} R$$

The ball needs to be released at a minimum height of 2.5h above the ground to successfully perform a complete loop without leaving the track, assuming that it slides along a smooth track and does not roll.

Rolling: If the ball is rolling without sliding (see Appendix C3.3 for an explanation on rolling), part of its initial gravitational potential energy is converted to translational as well as rotational kinetic energy. Specifically this is translation of the centre of mass and rotation about the centre of mass. For a given total kinetic energy the translational speed will be lower as part of the energy is transformed into rotational kinetic energy, leaving less available for the translational kinetic energy. Therefore, the ball needs extra initial potential energy so that it will have a sufficient translational speed at the top of the loop to enable it to roll round without leaving the track.

The ball can no longer be considered to be a particle because of its rotational inertia. The linear and angular speeds of the ball are related by

$$v = \omega r \quad \text{(Appendix C3.3)}$$

where v is the translational velocity of the centre of mass;

ω is the angular velocity with which the ball is rolling about an axis through its centre of mass; and

r is the radius of the ball.

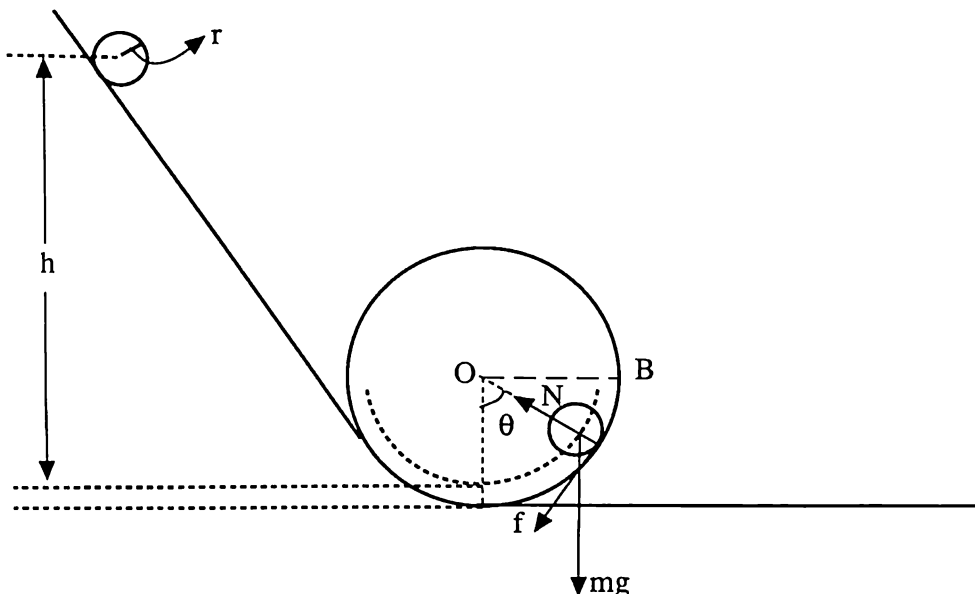


Figure 6.11: The forces acting on the rolling ball (f is the frictional force).

Force considerations: If the ball is rolling, there will be a frictional force (f) acting on it in the tangential direction (Figure 6.11) at the point of contact with the track. The frictional force does not do any work since there is no slipping or sliding. The frictional force exerts a torque (Appendix C9) on the ball. This supplies angular momentum to the ball and causes it to roll. The rate of change of the angular momentum of the ball is equal to the torque exerted (Appendix C10). It should be noted that the frictional force is present only if the track is rough and acts on the ball when there is a tendency for the ball to slide or slip due to another force acting on it. At the top and bottom of the loop, there is momentarily no force acting on the ball in the direction of motion. Therefore, at these two points there is no frictional force acting on the ball. Figure 6.10 shows the forces acting on the ball at the top of the loop. These are the same irrespective of whether it is sliding or rolling. The same requirement that $N \rightarrow 0$ gives the minimum velocity required at the top of the loop. Therefore, the expression for the velocity of the rolling ball at the top of the loop is

$$v = \sqrt{Rg}$$

Energy considerations: The rolling motion of the ball may be considered to be a pure rotation about its centre of mass combined with a translation of the centre of mass (Appendix C16).

At the top of the loop:

The gravitational potential energy of the ball

$$= mg(2R)$$

$$= 2mgR$$

The translational kinetic energy of the ball

$$= \frac{1}{2} mv^2$$

The rotational kinetic energy of the ball

$$= \frac{1}{2} I\omega^2 \quad (\text{Appendix C8})$$

The moment of inertia of a solid ball about an axis through the centre of mass is

$$I = \frac{2}{5} mr^2 \quad (\text{Appendix C15})$$

The total energy at the top of the loop

$$= 2mgR + \frac{1}{2} mv^2 + \frac{1}{2} I\omega^2 \quad (\text{Appendix C16})$$

Due to rolling

$$\omega = \frac{v}{r}$$

Initial energy = Energy at the top of the loop

$$mgh = 2mgR + \frac{1}{2} mv^2 + \frac{1}{2} I\omega^2$$

$$mgh = 2mgR + \frac{1}{2} mv^2 + \frac{1}{2} \frac{2}{5} mr^2 \left(\frac{v}{r}\right)^2$$

$$= 2mgR + \frac{1}{2} mv^2 + \frac{1}{5} mv^2$$

$$= 2mgR + \frac{7}{10} mv^2$$

Since,

$$v^2 = Rg$$

Then,

$$mgh = 2mgR + \frac{7}{10} mgR$$

Therefore,

$$h = \frac{27}{10} R$$

Notice that for rolling, h is $2.7R$ but for sliding it is only $2.5R$. This gives a difference in potential energy at the start to be $0.2mgh$. The extra $0.2R$ provides additional potential energy which is transformed to kinetic energy of rolling. At the top of the loop, the kinetic energy due to rolling is given by

$$K = \frac{1}{2} I \omega_{\min}^2$$

where

$$\omega_{\min} = \frac{v_{\min}}{r}$$

$$K = \frac{1}{2} \cdot \frac{2}{5} m r^2 \frac{v_{\min}^2}{r^2}$$

The minimum velocity at the top of the loop is given by

$$v_{\min}^2 = gR$$

Therefore,

$$\begin{aligned} K &= \frac{1}{5} mgR \\ &= 0.2mgR \end{aligned}$$

This is equal to the additional potential energy at the start.

Next an alternative solution for rolling is given.

6.5 ALTERNATIVE SOLUTION FOR ROLLING

The solution can also be performed by considering the motion of the ball about a tangential axis, as shown in Figure 6.12. An explanation of this is given below.

In Appendix C16, it is shown that the total kinetic energy of the ball may be considered to be the sum of the translational kinetic energy of its centre of mass and its rotational kinetic energy relative to the centre of mass. The motion may be considered as purely rotational by choosing a tangential axis of rotation passing through the point of contact between the ball and the track and perpendicular to the plane of the loop-the-loop track. Since the point of contact is instantaneously at rest, there will be no translational motion of the axis of rotation at any instant of time.

Therefore, the ball will be considered to have only rotational kinetic energy at any instant in time for its path along the track.

Taking the lowest point of the ball (at the bottom of the loop) as the zero gravitational potential energy level, the initial gravitational potential energy

$$= mgh$$

At the top of the loop the gravitational potential energy of the ball

$$= mg(2R)$$

The rotational kinetic energy of the ball

$$= \frac{1}{2} I \omega^2$$

where I is the moment of inertia of the ball about a tangential axis

As there is no conversion of energy to other forms the total energy at any instant will be the same.

Initial energy = Energy at the top of the loop

$$mgh = mg(2R) + \frac{1}{2} I \omega^2$$

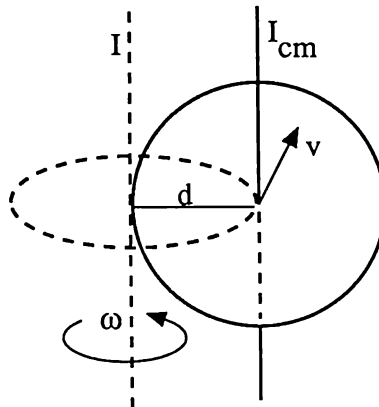


Figure 6.12: Applying the parallel axis theorem.

Using the parallel axis theorem

$$I = I_{cm} + md^2$$

where I_{cm} is the moment of inertia of the ball about an axis through the centre of mass; and d is the distance between the two axes (Figure 6.12).

For a solid ball,

$$I_{\text{cm}} = \frac{2}{5} mr^2$$

So,

$$\begin{aligned} I &= \frac{2}{5} mr^2 + mr^2 && (d = r, \text{ Figure 6.12}) \\ &= \frac{7}{5} mr^2 \end{aligned}$$

Since it is pure rotational motion

$$v = \omega r$$

where v is the tangential speed of the centre of mass of the ball while rotating round the axis through the point of contact (Figure 6.12);

ω is the angular speed of the centre of mass; and

r is the displacement of the centre of mass from the axis of rotation.

Therefore,

$$\begin{aligned} mgh &= mg(2R) + \frac{1}{2} \frac{7}{5} mr^2 \left(\frac{v}{r}\right)^2 \\ &= 2mgR + \frac{7}{10} mv^2 \end{aligned}$$

Since

$$v^2 = Rg$$

at the top of the loop (see force considerations for the general solution in section 6.3).

$$\begin{aligned} mgh &= 2mgR + \frac{7}{10} mRg \\ h &= \frac{27}{10} R \end{aligned}$$

The information given in Appendix C is applicable to this solution as well.

6.6 CONCLUSION

In this chapter the solutions to the problem obtained using pen and paper have been presented and discussed. The information experts externalise to solve such a problem is minimal, and the interconnections and conditions of applicability of their knowledge

are not explicitly stated. The teachers provided the knowledge which would normally be presented to students. The general solution outlines the considerable amount of information relevant to the solution. The alternative solution for rolling provides another aspect of the knowledge required to solve the problem.

The difference between the knowledge explicitly mentioned in the solutions by the expert and teachers and that derived in the general solution shows the extent of knowledge which is assumed and not normally referenced during problem solving.

While it is accepted that it would be tedious to reference all this information when solving the problem it provides the information needed to set up an interactive program. This would also clarify conditions of applicability and the validity of assumptions made.

Not all the information compiled has been included in the program since this would make the program too tedious to use. It would hamper the interaction as the program would need to keep branching off. However, the necessary qualifications for the use of the knowledge are included to highlight the applicability of the knowledge. The information supplied by the general solution can be used as a supplement for referencing information not included in the program.

In the next chapter the features of the program are highlighted showing how the knowledge outlined in this chapter and Chapter 5 of the research is incorporated.

CHAPTER 7

THE PROGRAM

7.1 INTRODUCTION

In this chapter, the program developed for the loop-the-loop rotational dynamics problem (Figure 4.3) is detailed and discussed. The Macintosh environment coupled with the supportive environment provided by the Think-C Class Library (an object-oriented version of C) and the C programming language have been used to develop the program. The information gathered in phases 1 and 2 of the research schedule using a variety of methods (Chapter 4) and analysed in Chapters 5 and 6 have been included. This has informed the interaction style and knowledge base of the program. A user manual describing how to get started and detailing the features and workings of the program is given in Appendix D1. The program is supplied on a floppy disk included in the manual. The complete program code is given in Appendix D2.

The aim was to provide the students with a program which they could consult for a multitude of purposes (section 1.1) and which would allow them to control their learning in the interactive process. Therefore the program is based on a constructivist environment with information being supplied through transmission when needed or requested. The program has also been used to determine the assistance students seek and the features they use in interacting with a computer-assisted problem-solving system. It compiles the interaction of the student during consultation providing data for insight into student knowledge, conceptions and problem-solving styles.

The discussion commences with the six main components of the program namely force equations, energy equations, envisioning, solution of the problem, simulation and answer checking. The purposes for which these can be used and their workings are discussed in detail. This highlights the features of the program which allow access to information. Then follows a discussion of the internal structure of the program to highlight the provision of the expertise module, the student module, the tutoring module, the interface and the knowledge representation and access. These terms are explained in section 3.3. Then follows a discussion on the incorporation of the 14 problem-solving activities detailed in section 5.4. Next the 6 main components of the program are discussed to highlight the provision of the problem-solving strategies used by participants. The structure of the program is described in terms of the extent to which it provides interaction consistent with a constructivist learning environment where provision of knowledge when needed or requested is made. The chapter

concludes with a discussion of the knowledge and program features which have been intentionally excluded.

The six main components of the program are first detailed to provide insight into their use and the structure of the program.

7.2 MAIN PROGRAM COMPONENTS

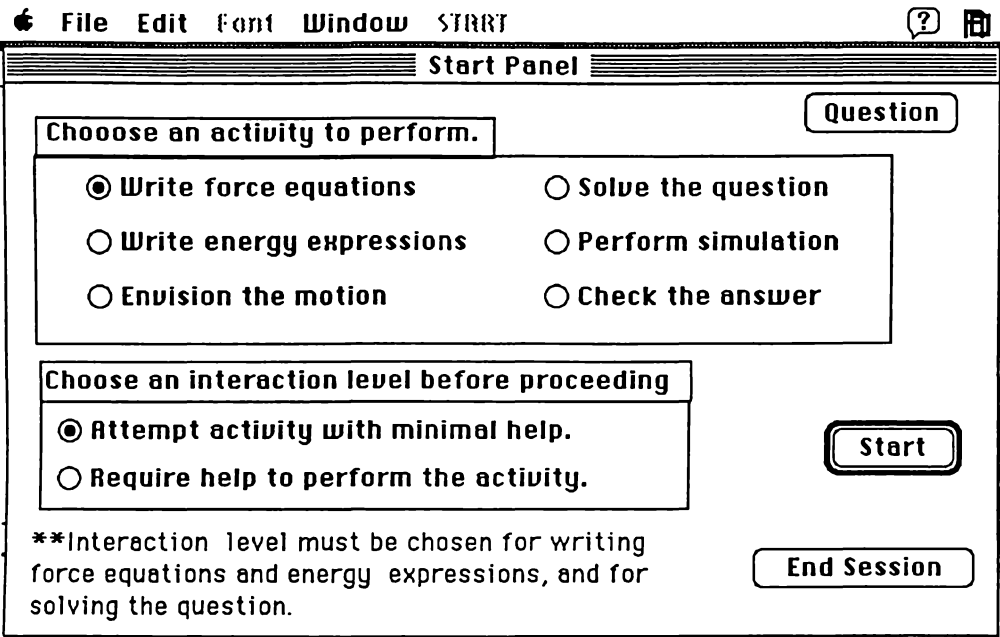


Figure 7.1: The start panel.

The six main components provide help and information for writing force equations, writing energy expressions, performing the solution of the problem, envisioning the motion of the ball, performing simulations and checking the answer to the problem (Figure 7.1). Two levels of interaction are available for the first three purposes, the first level providing final expressions as a check, the next level giving hints and cues. At both levels the following relevant information is accessible through buttons and menus:

- An explanation on commencing the particular interaction.
- An explanation of the features currently on the screen.
- Relations and expressions of physical quantities needed.
- A check facility to provide feedback.

The *Start panel* (Figure 7.1) is first discussed to highlight information provided at session commencement and access provided to the rest of the program. General

movement from one section of the program to another is also described. Then follows a detailed discussion of each of the six main components. For each component the initial information provided on the screen is first detailed, then the information which may be accessed is discussed. Finally, some general facilities not directly involved with the solution of the problem are mentioned and briefly discussed.

7.2:1 The start panel and general access

The interaction begins with an untitled worksheet (Figure 7.22) followed by the start panel shown in Figure 7.1. The activity *Write force equations* and the first interaction level labelled *Attempt activity with minimal help* are selected as the default choices during program commencement. (The default selections have been made to highlight the need to make selections prior to commencement of interaction, and are not intended to imply the activity which the student should first perform.)

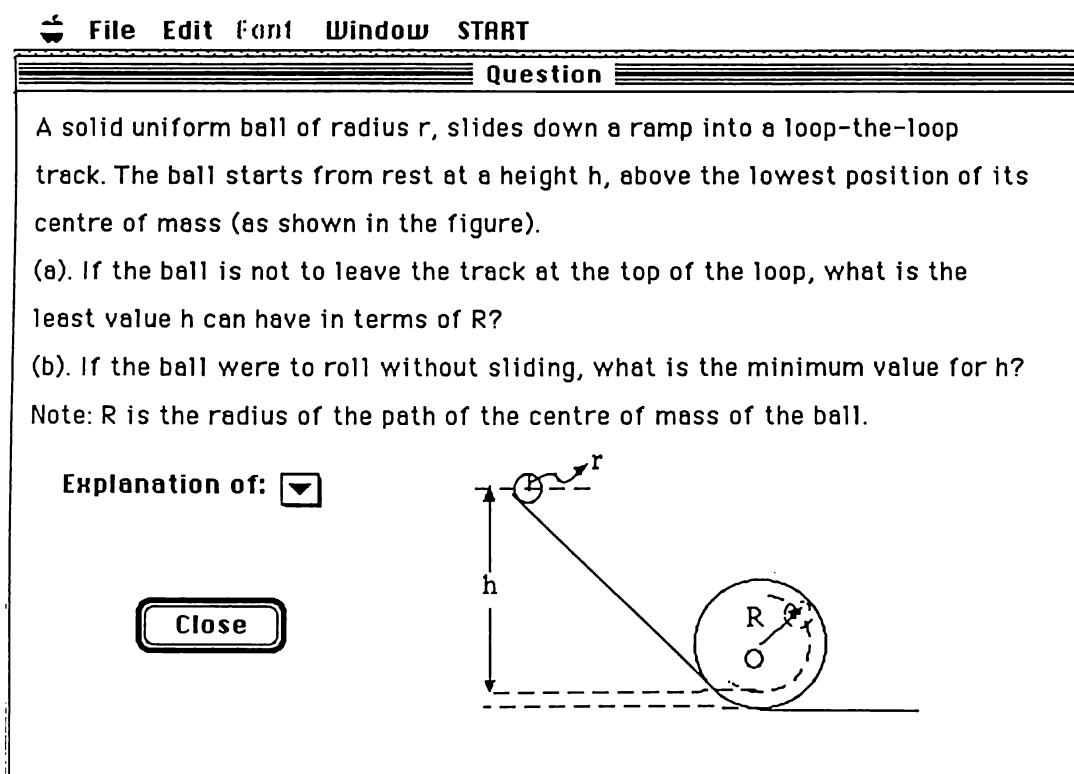


Figure 7.2: The question window providing the problem wording and access to question clarification.

Explanation of:

✓ centre of mass
rolling vs sliding
slipping
uniform ball

Pure translation Pure rotation Rolling

When the ball moves it can rotate as well as translate. Except for the centre, each point on the ball has a velocity due to rotation V_T and that due to translation V_{cm} .

For sliding: $V_T = 0$

For rolling $V_{cm} = V_T$

Friction must be present for the ball to roll.

OK

Figure 7.3: The question clarifications available and the information provided when rolling vs sliding is selected.

The session is commenced by clicking either the *Question* or the *Start* button. Clicking the *Question* button opens a window providing access to the problem (Figure 7.2) and question clarifications sought by students (Chapter 5). Figure 7.3 shows the possible clarifications, and an example of the information shown when the rolling vs sliding selection is made.

Clicking the *Start* button opens a window to commence one of the 9 interactions: Force Equations, Check Force Equations, Energy Expressions, Check Energy Expressions, Envision, Perform Solution, Perform Solution Check, Perform Simulation, and Check Answer.

File Edit Font Window START	Untitled Worksheet 1 Question Perform Simulation
--	---

Figure 7.4: The contents of the Window menu indicating the open windows during interaction.

When a window is opened its title is included in the *Window* menu in the menu bar. Figure 7.4 shows the contents of this menu indicating that the question, simulation and untitled worksheet are the open windows at that time. Open windows may be selected and brought to the front by:

- selecting the relevant menu item in the *Window* menu;
- clicking in any visible portion of the window; or
- making the relevant choices in the *Start panel* followed by a click in the *Start* button.

When the start panel is the front window, the *Start* menu is disabled and appears greyed. If the start panel is in the background then the *Start* menu is enabled. If the Start panel has been closed by a click in the *End Session* button then it can be opened by selecting *Start Session* from the *Start* menu. A click in the *End Session* button closes the Start Panel window and all the others except any worksheet (e.g. Untitled Worksheet 1) which might be open. It also replaces the menu item labelled *Start Panel* with the label *Start Session* in the *Start* menu (Figure 7.5).



Figure 7.5: The Start menu with the start panel closed, open and in the background, and open and currently active.

The 9 interaction windows are next discussed in detail.

7.2:2 Force Equations and Check Force Equations

This double component enables the student to write force equations at various positions of the ball during its motion. Two levels of interaction are available, the first opens the window titled *Force Equations* and the second opens the window titled *Check Force Equations*. Both windows are shown in Figure 7.6 below.

The common features between the two interactions are:

1. the option to choose sliding or rolling.

- File Edit Font Window START Window Information Force Diagrams

File Edit Window START Window Information

Check Force Equations

☒ Sliding
☐ Rolling

Position of ball ▼

Need help with: ▼

Direction

☒ Perpendicular to track
☐ Along or tangent to track

Check window

Worksheet

CHECK

a F g
 M N r
 R B J
 sin cos f

+ - *
 ÷ =
 Square
 Return

Velocity ▼
 Angular velocity ▼
 Number ▼

Force Diagram

DONE

157

- the keyboard choices using buttons (e.g a, F, g) and pop up menus for accessing velocity and angular velocity symbols, and numbers to type the force equations.
- the *Window Information* menu which provides information on the items presented in the window (Appendix D1).
- a worksheet for writing the force equations and performing the general working.
- a DONE button for closing the window and ending the interaction.

Force Equations: For this interaction clicking the *CHECK button* provides the final force equation for the chosen position of the ball for sliding or rolling. An example of an answer for sliding motion of the ball at a general point in the loop is given in Figure 7.7.

SLIDING: AT A GENERAL POINT IN THE LOOP

In the radial direction:

$$N - Mg\cos\beta = Mv^2 / R$$

In the tangential direction:

$$a_t = - g\sin\beta$$

a_t is the acceleration in the tangential direction.

NOTE:

Select the "Window Information" menu to identify the other quantities represented by the symbols.

OK

Figure 7.7: The final force expression obtained for a general point in the loop.

The Force Diagrams menu can be used to obtain force diagrams for 9 positions of the ball: at the start, on the incline, at the bottom of the loop, in the lower right quadrant of the loop, in the top right quadrant of the loop, at the top of the loop, in the top left quadrant of the loop, in the lower left quadrant of the loop, and along the horizontal.

Figure 7.8 gives an example of the force diagram for the lower left quadrant of the loop.

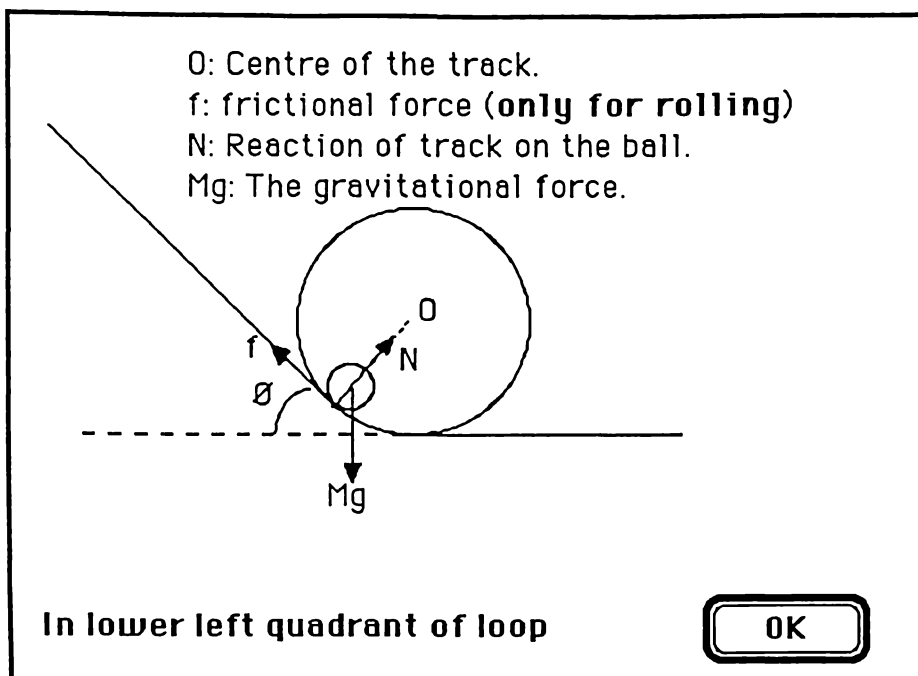


Figure 7.8: The force diagram for the lower left quadrant of the loop.

Check Force Equations: The major difference between this interaction window and the *Force Equations* window is the provision of interactive checking and force diagram facilities. The check facility helps to check the force equations placed in the *Check window*. The student can perform general working without checking by using the worksheet. If the student encounters difficulties in writing the force equation for a selected position and direction, the program suggests starting with $F = Ma$. The following steps are required to check the force equation:

- Choice of sliding or rolling.
- Selecting the direction for the force equation.
- Selecting a position of the ball.
- Placing the equation in the Check window.
- Clicking the Check button.

Each time the equation is modified the *CHECK* button is clicked to continue the interaction. Information is provided using dialogue boxes (Figure 7.9). The window provides practice in writing the force equations for various positions of the ball. To solve the problem the force equation needs to be written for the top of the loop. If this is successful, the student is prompted to the *Perform Solution Check* window if the solution to the problem was desired.



PERPENDICULAR TO THE TRACK GENERALLY IN THE LOOP

Your equation should read $Ma = N - Mg\cos\beta$.

a is the acceleration towards the centre of the track.
Substitute for " a " using "relations" given in "Need help with".

OK

Figure 7.9: Information provided through a dialogue box during interaction.

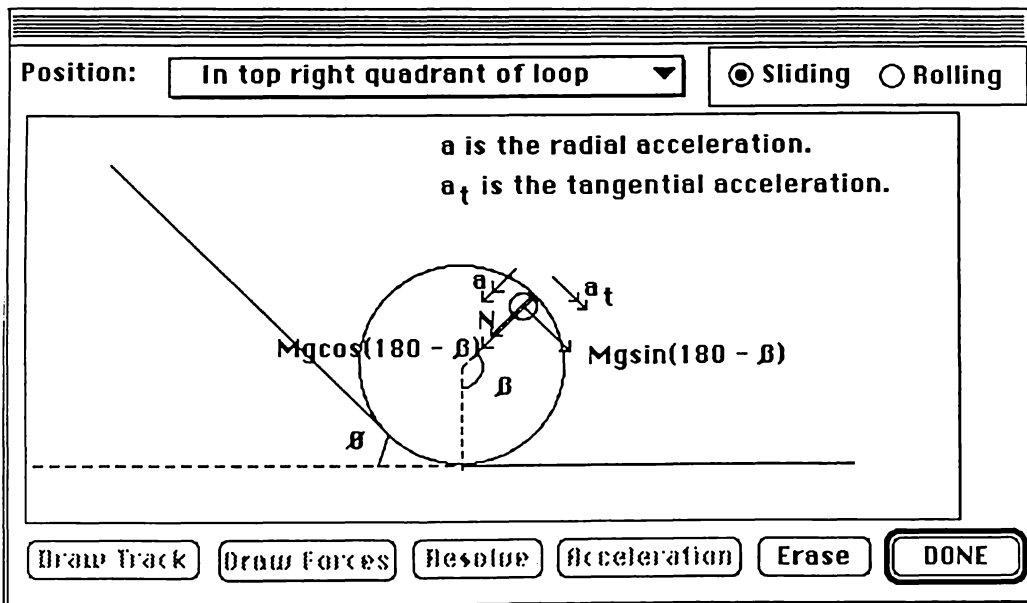


Figure 7.10: The Force Diagrams window showing the resolved forces and accelerations for a rolling ball in the top right quadrant of the loop.

The interactive force diagram facility can be accessed by opening the *Force Diagram* window where forces for rolling or sliding motion at a position of choice can be drawn for 9 choices of ball positions. Diagrams for resolved forces and associated acceleration

directions can also be depicted. Figure 7.10 shows the *Force Diagram* window with resolved forces and accelerations depicted for a rolling ball in the top right quadrant of the loop.

7.2:3 Energy Expressions and Check Energy Expressions

This double component of the program enables the student to write energy expressions at various positions of the ball during its motion. Two levels of interaction are available, the first opens the window titled *Energy Expressions* and the second opens the window titled *Check Energy Expressions*. Both these windows are shown in Figure 7.11.

The common features between the two interactions are:

1. the option to choose sliding or rolling.
2. the choice of checking the energy expressions for different positions of the ball.
3. the *Need help with* option to obtain help on commencing the interaction, and information on relations between physical quantities and on energy expressions.
4. The keyboard choices using buttons (e.g, h, l) and pop up menus for accessing velocity and angular velocity symbols, and numbers to type the energy expressions.
5. The *Window Information* menu which provides information on the items presented in the window (Appendix D1).
6. a worksheet for writing the energy expressions and performing the general working.
7. a DONE button for closing the window and ending the interaction.

Energy Expressions: Clicking the *CHECK* button provides the final energy expression for sliding or rolling which can be checked for the start, the bottom of the loop, the top of the loop and for a general point in the loop. The equation obtained by equating the energy expressions for any two of these four positions may also be checked. Figure 7.12 shows the dialogue boxes which present the energy expression at the top of the loop for sliding motion, and the equation obtained by equating the energies at the start and top of the loop.

File Edit Font Window START Window Information

Energy Expressions

☒ Sliding
 ☐ Rolling
 Need help with: ▼

Energy of the ball at the: start ▼

CHECK

Worksheet

g h l

M r R

y

= * ÷ +

- Square

Return

Velocity ▼

Angular velocity ▼

Number ▼

DONE

File Edit Font Window START Window Information

Check Energy Expressions

☒ Sliding
 ☐ Rolling
 Need help with: ▼

Position 1 ▼
Position 2 ▼

Translational K.E.		
Rotational K.E.		
Gravitational P.E.		
Total energy		

Equate Totals CHECK

g h l

M r R

y

= * ÷ + -

Square

Velocity ▼

Angular velocity ▼

Number ▼

DONE

Figure 7.11: Writing energy expressions interaction windows.

AT THE TOP OF THE LOOP

Energy of the sliding ball = $2MgR + \frac{1}{2} Mv_T^2$

Clarify

OK

EQUATING THE ENERGY AT THE TOP AND BOTTOM OF THE LOOP

For a sliding ball

$$v_B^2 - v_T^2 = 4gR$$

OK

Figure 7.12: The final energy expression for the top of the loop and the equation for the start and the top of the loop.

Check Energy Expressions: The major difference between this interaction window and the *Energy Expressions* window is the provision of interactive checking and equating facilities. The check facility helps to check the energy expressions placed in the respective energy slots. The interaction starts with a selection of sliding or rolling (the default selection is sliding). To write the expression for a single position a choice from Position 1 (at the start, the bottom of the loop, the top of the loop, or a point on the incline or the loop) is made. Position 2 offers the first three of these choices. To equate the energy expressions at two positions of the ball, both choices are made and the energy expression is placed in each of the relevant slots. Clicking the *CHECK* button starts the check. Each time information in a slot is modified the *CHECK* button is clicked to continue the interaction. Information for guidance is provided using dialogue boxes (Figure 7.13).

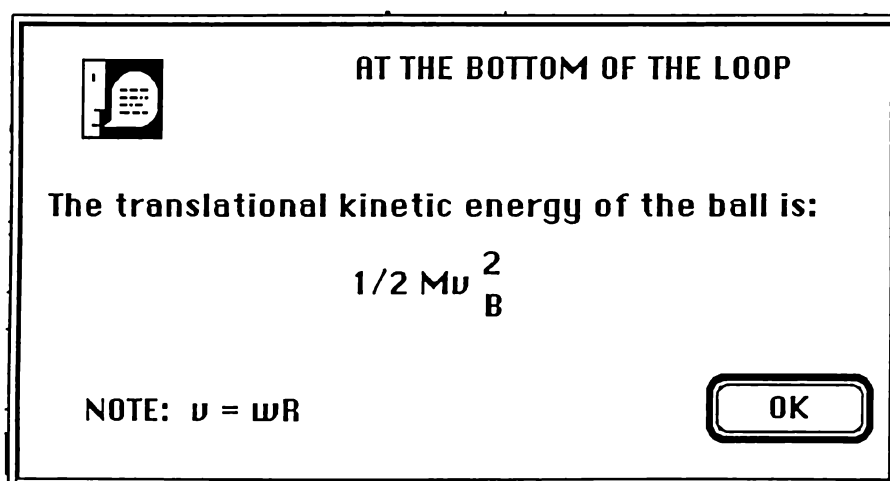
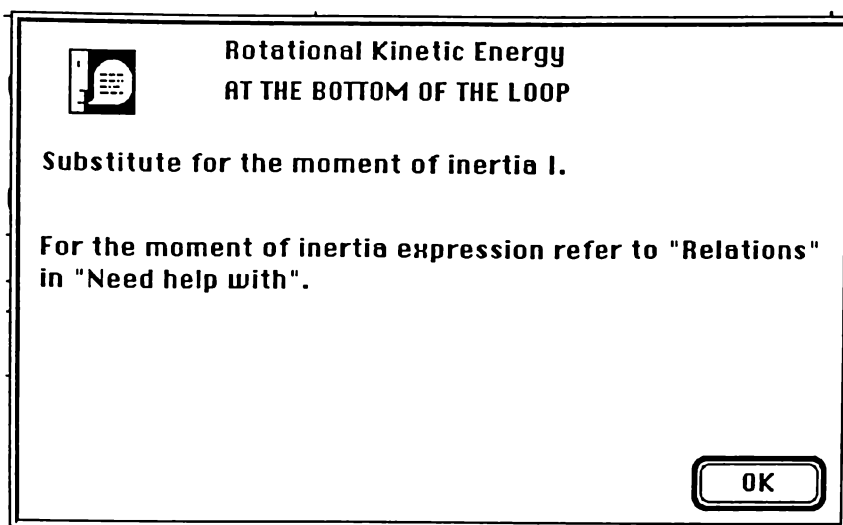


Figure 7.13: Two examples of information provided through a dialogue box during interaction for writing energy expressions.

If the check is successful the program sums the translational, rotational and gravitational energy expressions and places the final expression in the *Total energy* slot. If selections have been made for both positions then clicking the *Equate Totals* button equates the two total energy expressions and places the simplified expression in the slot beside the button. Figure 7.14 shows the completed check for the start and the top of the loop, the two positions at which the energy expressions have to be written to obtain the solution. In this case the student is prompted to write a force equation to help solve the problem.

File Edit Font Window START Window Information

Check Energy Expressions

☐ Sliding ☒ Rolling Need help with:

At the start Top of loop

Translational K.E.	0	$\frac{1}{2} Mv_T^2$
Rotational K.E.	0	$\frac{1}{5} Mv_T^2$
Gravitational P.E.	Mgh	Mg2R
Total energy	Mgh	$\frac{7}{10} Mv_T^2 + 2MgR$

$v_T^2 = 10/7 g(h - 2R)$

g h l = * ÷ Velocity
 M r R + - Square Angular velocity
 y Number

Figure 7.14: A completed check for the start and the top of the loop for a rolling ball.

File Edit Font Window START

Envision

Perform envisioning for: ☒ Sliding ☐ Rolling

Select one of the following and click the "Display" button to commence the envisioning.

☐ Overview ☐ Moving down in the loop
☐ Down the incline ☐ Along the horizontal
☐ Moving up in the loop ☒ Insufficient starting height

If the ball starts at an insufficient height then:
 if $h \leq R$ the ball will oscillate between the starting point and the point in the loop at a similar height;
 otherwise, it will leave the track on the way up, in the upper half of the loop and perform free fall projectile motion.
 You can simulate these by selecting the Perform Simulation option from the Start panel.

Figure 7.15: The envisioning window showing the description of the motion for *Insufficient starting height*.

7.2:4 Envision the motion

This component of the program provides a qualitative description of the motion of the ball at various positions during its motion. A selection can be made between sliding or rolling. The default selection is for the sliding motion of the ball. The description of the motion can be obtained by selecting one of the following 6 options and clicking the *Display* button: overview, down the incline, moving up in the loop, moving down in the loop, along the horizontal, and insufficient starting height.

Figure 7.15 shows the envisioning window which can be opened by selecting *Envisioning the motion* and clicking the *Start* button in the Start Panel (Figure 7.5). The description for *Insufficient starting height* is also shown in Figure 7.15.

7.2:5 Perform Solution and Perform Solution Check

This double component of the program provides information to solve the question. Two levels of interaction are available, the first opens the window titled *Perform Solution* and the second opens the window titled *Perform Solution Check*. Both windows are shown in Figure 7.16.

The common features between the two interactions are:

1. the choice of checking the answer.
2. the *Need help with* option to obtain help on commencing the interaction, and information on relations between physical quantities and on energy expressions.
3. the keyboard choices using buttons (i.e. F, h, I) and pop up menus for accessing velocity and angular velocity symbols, and numbers to type the energy expressions.
4. the *Window Information* menu which provides information on the items presented in the window (Appendix D1).
5. the *Force Diagrams* menu which provides force diagrams for 9 positions of the ball (section 7.3:2).
6. a DONE button for closing the window and ending the interaction.

Perform Solution: For this interaction clicking the *CHECK button* provides the final expression for the minimum starting height, h , for sliding or rolling motion of the ball. Figure 7.17 shows the dialogue box produced when the CHECK button is clicked for rolling motion.


 File Edit Font Window START Window Information Force Diagrams

Perform Solution

☒ Sliding ☐ Rolling
 Need help to: ▼

Worksheet

a	F	g
h	I	M
N	r	R
y	Ø	β
f		

cos	sin	=	*
÷	+	-	√
Square		Return	

Velocity ▼
 Angular velocity ▼
 Number ▼


 File Edit Font Window START Window Information Force Diagrams

Perform Solution Check

Position choice: Need help with: ▼

Position 1 ▼

Position 2 ▼

Force equation:

Force Check

Energy expression: Energy Clues

a	F	g
M	N	r
R	β	Ø
cos	sin	sin

+	-	*
÷	=	
Square		
Return		

Velocity ▼
 Angular velocity ▼
 Number ▼

Figure 7.16: The two windows which can be used to perform the solution to the problem.

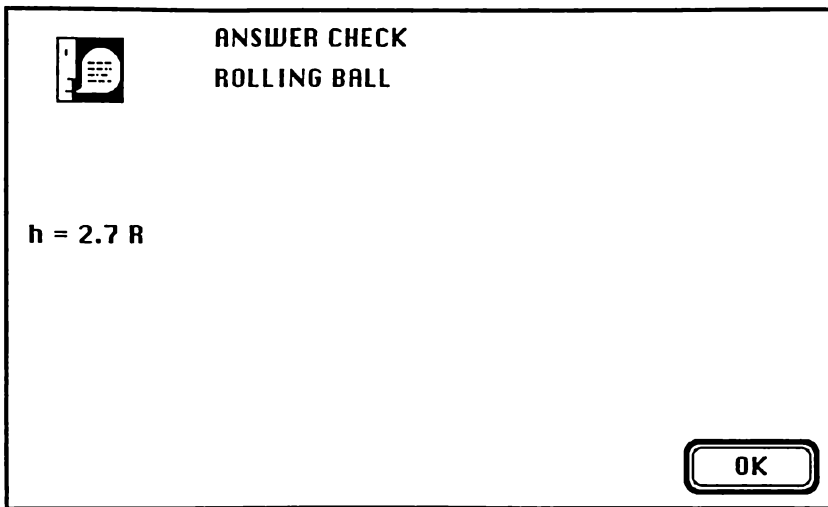


Figure 7.17: A dialogue box giving the final expression for the minimum starting height for rolling motion.

Perform Solution Check: The major difference between this interaction window and the *Perform Solution* window is the provision of the following information:

- a check of the force equation for the top of the loop.
- clues on the positions for which to write the force equation and energy expressions.
- further simplification of the force equation for the top of the loop in the *Force Equation Check* interaction window.
- clues on the equations required to solve the problem.

Interaction starts by clicking the *Force Check* button. Choices of the position of the ball for which the equation should be written are then required. If the wrong choice is made information to correct the choice is given. Figure 7.18 shows dialogue boxes available.

The force equation can be checked by placing it in the *Force equation* slot and clicking the *Force Check* button. The check facility only checks the force equation for the top of the loop. Checks for force equations at other positions of the ball can be made in the *Force Equation Check* window. The clues to help write, check and simplify the force equation include:

the position at which the force equation should be written.

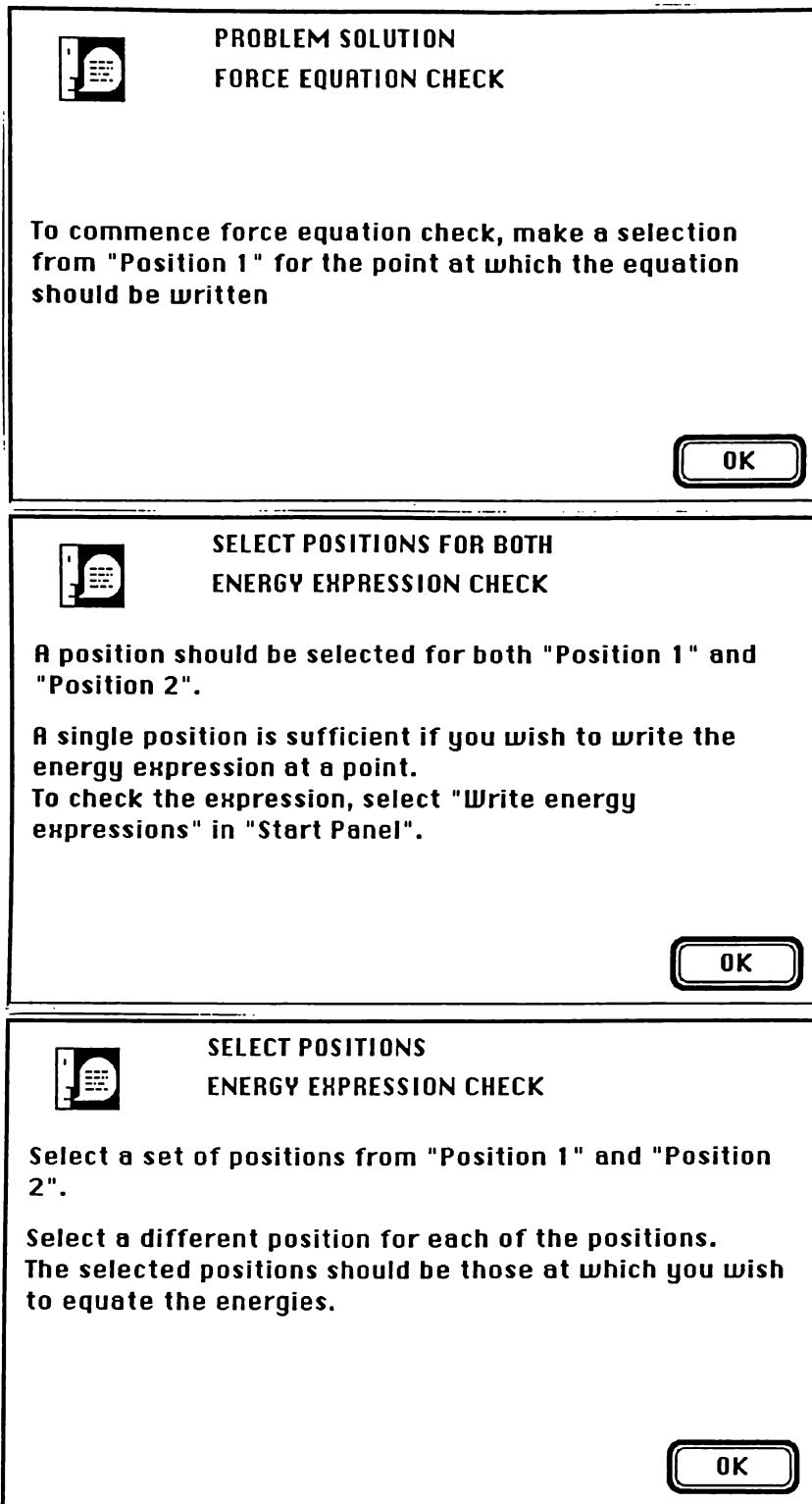


Figure 7.18: Dialogue boxes giving information on the positions at which to write the force equation and energy expressions.

the constraint of $N = 0$ for the minimum starting height of the ball.

a prompt that an energy equation needs to be written as well.

The energy expression facility provides clues on the positions at which the energy expressions need to be written. Interaction is commenced by clicking the *Energy Clues* button. Once the clues have been given, the student is prompted to access the *Check Energy Expressions* window to write the expression.

7.2:6 Perform Simulation

This component of the program provides simulations of the ball for its rolling and sliding motion. The simulation window showing the simulated motion for rolling is given in Figure 7.19. The simulations can be used to study the motion of the ball for different starting heights relative to the radius of the loop, to check the final expression obtained for the starting height, and to study the dependence of the motion on the incline of the track and the gravitational intensity, g .

The simulation window can be opened by selecting *Perform simulation* and clicking the *Start* button in the Start Panel (Figure 7.1).

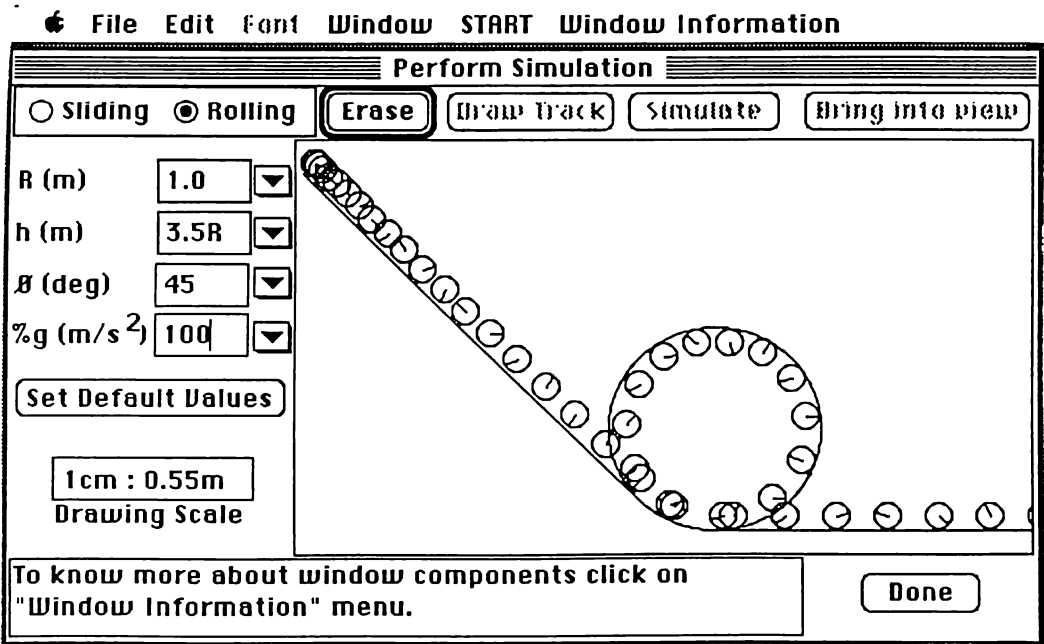


Figure 7.19: The simulation window showing a simulation for rolling motion of the ball.

The main features provided by the simulation window are:

1. a choice of sliding or rolling.
2. the facility to choose different values for the starting height (h) of the ball, the radius R of the loop performed by the ball, the value of the angle of incline of the track ($\varnothing$) and the value for the gravitational intensity or acceleration due to gravity (g). The values for these four quantities can be typed into the respective slots or chosen from pop-up menus (Figure 7.20).
3. a button which sets the values of R , h , $\varnothing$ and g to the default values of 1.0, 3.5R, 45° and 100%. These values provide an optimum size for the track and for the motion of the ball, and help in selection of values for simulation.
4. the *Drawing Scale* used to represent the track and the size of the ball provides information on the size of the actual track using values set for h and R .
5. the *Erase* button clears all drawing performed on the screen for further simulation.
6. the *Draw Track* button draws the track using the values provided for R , h and $\varnothing$.
7. the *Simulate* button commences the simulation and draws the positions of the ball at regular time intervals using the values of R , h , $\varnothing$ and g supplied.
8. the *Bring Into View* button opens a larger window for simulation if the angle chosen for drawing the track is such that part of the track is obscured.

Values of R	Values of h	Incline angle	% g
1.0	R	45	100
-----	-----	-----	-----
0.5	0.5R	10	50
1.5	1.5R	15	150
2.0	2R	20	200
2.5	2.5R	25	250
3.0	3R	30	300
3.5	3.5R	35	350
4.0	4R	40	400
4.5	4.5R	50	450
5.0	5R	55	500
-----	5.5R	60	-----
Other	6R	65	Other
	-----	70	
	Other	75	

		Other	

Figure 7.20: Pop-up menus for the simulation window.

9. the *Window Information* menu provides information on the items presented in the window (Appendix D1).
10. the *DONE* button closes the window and ends the simulation.

7.2:7 Answer Check

This component of the program provides a check on the final expression for the minimum starting height of the ball for rolling and sliding motion. This component provides the answer as a check (Figure 7.21) so that students who have solved the problem using pen and paper can check their answers.

This window can be opened by selecting *Answer Check* and clicking the *Start* button in Figure 7.1.

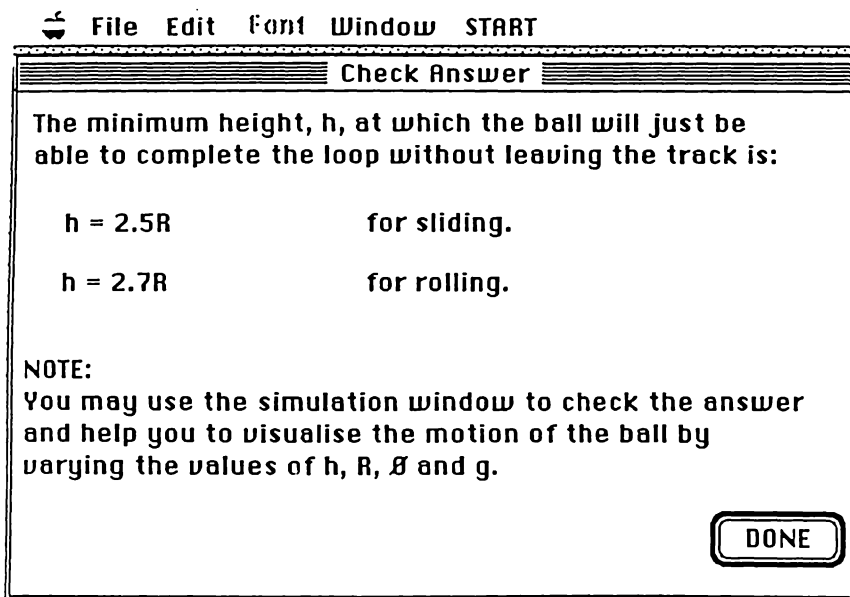


Figure 7.21: The *Answer Check* window.

7.2:8 Other general facilities

- (a) Another major component is a general worksheet enabling an edited and printable copy of the solution (Figure 7.22). The *Cut* and *Paste* functions in the *Edit* menu can be used to transfer working from other components of the program. With these and the *Font*, *Size* and *Style* menus the student can prepare a final copy of the solution which can be printed by selecting the *Print* item in the *File* menu.

- (b) A component which compiles a list of the interaction with the program has also been included to provide feedback on the facilities used. The interaction information compiled is discussed in Chapter 8.

This completes the discussion of the major components and facilities provided by the program. Information on use of the program is provided in the user manual (Appendix D1).

The following discussion focuses on the extent to which the program performs and provides the essential ICAI components of expertise module, student module, tutoring module, interface, and knowledge representation and access.

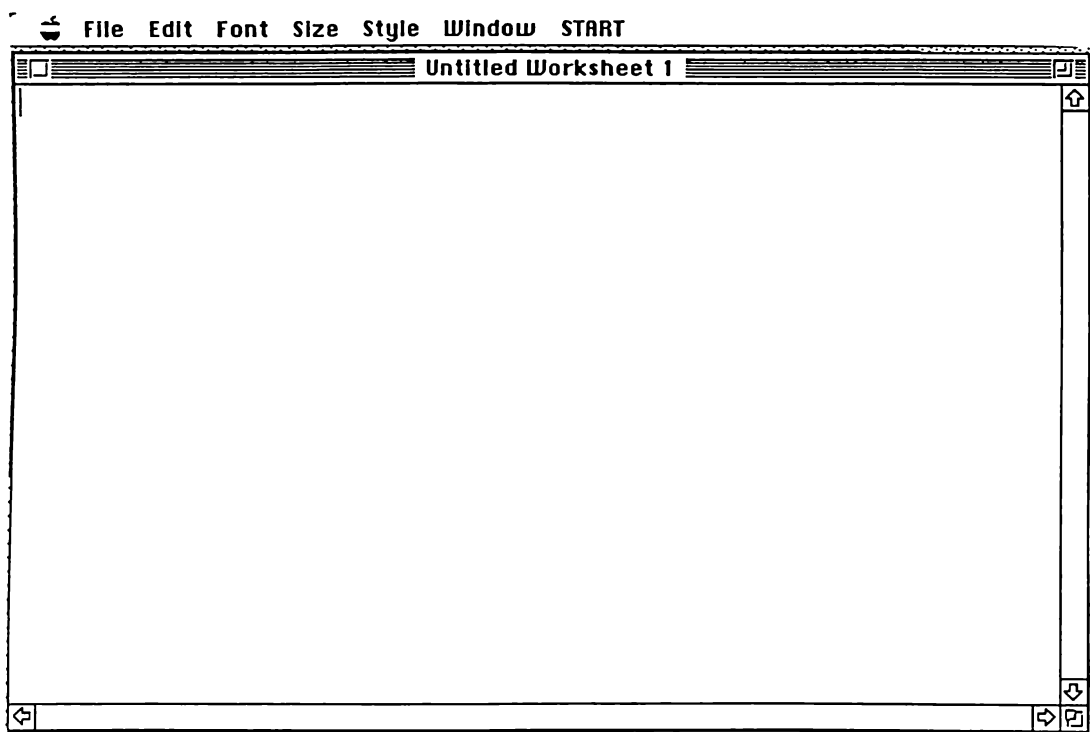


Figure 7.22: The untitled worksheet.

7.3 THE INTERNAL PROGRAM STRUCTURE

The internal structure of the program is discussed to show how the analyses in Chapters 5 and 6 are addressed. The expertise module, the student module and the tutoring module are not strictly separate entities, since all three are involved in assisting the problem-solving interaction. During problem-solving expert knowledge,

knowledge about the student and the information supplied by the student are used to supply information and to aid the interaction. The student makes decisions on which information to access and what activity to perform. Each of these three modules are discussed, followed by the interface, and the knowledge representation and access.

IF ACTION: Energy clues button clicked and no information is supplied. THEN PROMPT: Select a set of positions from Position 1 and Position 2. Select a different position for each of the positions. The selected positions should be those at which you wish to equate the energies.
IF ACTION: At the start and bottom of the loop are the two selected positions for Position 1 and Position 2. THEN PROMPT: The energy expression should be written for the start and the top of the loop. It may also be obtained by writing expressions for: (i) start and bottom of the loop; (ii) bottom and top of the loop; and (iii) combining the two expressions. To write the energy expressions, select <i>Write energy expressions</i> from the Start Panel.
IF ACTION: The start and the top of the loop are the two positions selected for Position 1 and Position 2. THEN PROMPT: The two selected positions are correct. Select Write energy expressions from the Start Panel to write the energy expression.
IF ACTION: A position is selected from only Position 1 or Position 2 THEN PROMPT: A position should be selected for both Position 1 and Position 2 if you wish to equate energies. A single position is sufficient if you wish to write the energy expression at a point. To check the expression select Write energy expressions from Start Panel.

Figure 7.23: Rules used and knowledge supplied by the expertise module.

7.3:1 Expertise module

The expertise module in this program refers to the knowledge required to solve the problem and to provide explanations. The knowledge has been obtained from the clarifications and assistance sought by students, and the knowledge needed to solve the problem (Chapters 5 and 6). Information on clarification is directly available using dialogue boxes which can be accessed through pop-up menus or buttons. For example the question clarification on rolling and sliding can be obtained from the *Question* window (Figure 7.2). Figure 7.3 gives an example of information accessed through a selection from a pop-up menu in the *Question* window.

The knowledge needed to solve the problem is supplied appropriately during interaction. For example, clues on positions of the ball at which to write the force and energy equations are provided after some input from the students and not directly. The feedback provided is dependent on the choices made or information supplied. For example when a student does not know the position of the ball at which energy expressions should be written, rules and information (Figure 7.23) and dialogue boxes (Figure 7.18) are provided.

The student does not have to pursue the recommended action, or any prescribed sequence of steps. For example, Figure 7.23 shows four ways for obtaining information on the positions at which energy expressions may be written.

7.3:2 Student module

This includes the information obtained on the points at which problem-solving is commenced, the general mistakes, the conceptions portrayed and the difficulties encountered by the participants during problem solving, discussed in detail in Chapter 5. The problem-solving interaction has been subdivided into writing force equations, writing energy equations, envisioning the motion, solving the question, performing simulation and checking the answer. This subdivision enables the program to use the appropriate information to provide guidance rather than tracking each step of the interaction. The program acts as a knowledgeable teacher (Chapter 2) providing necessary information to help the interaction. Rather than presenting a summary of the information collected and assuming the student will make the connections, the program has been developed to deal with the immediate information supplied by the student. The student is assisted to make the connections through help and hints supplied. Conceptions of students and mistakes generally made inform the program on the type of help and information to provide.

The interactions are focussed through subdivision. For example, the *Check Energy Expressions* window (Figure 7.11) subdivides the information needed for the

interaction into small independent slots of information. In this way the mistakes made by the students can be traced and anticipated.

7.3:3 Tutoring module

This module governs the interaction using information from the expertise module, the student module and the problem-solving strategies used by the students (Chapter 5). The information supplied by the student is compared with that of the expertise module to supply help at the general and the specific level. Rules which are inherent in the program interaction and not made explicit to the student determine the information supplied by the student, and provide the correct level of help during interaction. Figure 7.24 gives an example of the different information given to determine the force equation for a rolling ball at the top of the loop.

The tutoring rules are arranged so that if the student does not know how to commence, the prompt suggests starting with $F = Ma$. A similar prompt is given if the information supplied by the student does not make sense due to the presence of errors. The tutoring module searches for mistakes in the following order:

- The choice of position and direction.

- Mathematical mistakes (such as absence of an equal sign or the presence of unwanted symbols).

- The presence of information which can be replaced to simplify the expression.

The tutoring module does not correct mistakes due to incorrect signs but provides an equation closest to the format supplied. This enables the student to effect necessary corrections. During the interaction the tutoring module also supplies the following information:

- Hints on how to perform the next step.

- Clues on where and how to obtain more information about the hints and information supplied.

- A brief explanation of the information supplied.

IF: Position of the ball at which equation is to be written is not selected. PROMPT: Select a position of the ball.
IF: Position is supplied but no equation is written. PROMPT: Place the equation you wish to check in the <i>Check Window</i>. If you are unsure click on <i>Force Diagram</i>. Otherwise start with $F = Ma$ and click the <i>CHECK</i> button.
IF: No equal sign. PROMPT: There is no equal symbol in your equation.
IF: A number is included in the expression. PROMPT: The equation should not have any number in it.
IF: Angle symbol is included in the equation. PROMPT: Angle is 180^0 at the top of the loop giving a cosine value of -1.
IF: Wrong alphabetic symbol is used e.g. lower case r instead of capital R. PROMPT: Change the symbol e.g. r is the radius of the ball. Use R the radius of the loop.
IF: $F = Mg$ PROMPT: F is the resultant force towards the centre of the track. It is equal to the sum of N and Mg. For more information click <i>Force Diagram</i>.
IF: $N = Mg$ PROMPT: N is not equal to Mg. F the resultant force towards the centre of the track is equal to the sum of N and Mg. For more information click <i>Force Diagram</i>.
IF: $F = Ma$ PROMPT: F is the resultant force towards the track. Replace it by the sum of N and Mg. For more information click <i>Force Diagram</i>.
IF: $N + Mg = Ma$ PROMPT: <i>a</i> is the acceleration towards the centre of the track. Replace it using relations in <i>Need help with</i>.
IF: $N + Mg = Mv_T^2$ PROMPT: Your equation should include an R. The acceleration is the square of the velocity divided by R.
IF: $N + Mg = \frac{v_T^2}{R}$ PROMPT: Your equation is missing an M. Check your working.
IF: $N + Mg = \frac{Mv_T^2}{R}$ PROMPT: Check completed. If you wish to apply the equation to solve the problem, return to the <i>Start Panel</i> and select <i>Solve the question</i>.
IF: Information supplied does not make any sense. PROMPT: Change the equation or start with $F = Ma$ and click the <i>CHECK</i> button. For more information click <i>Force Diagram</i> or <i>Need help with</i>.

Figure 7.24: Possible interactions for writing the force equation for the top of the loop.

The tutoring module provides the minimum help required to carry on the interaction, providing options to pursue and clues to further action. The student is not required to proceed in a fixed manner and can solve the problem in his/her own manner. Stepwise information leads those who are unable to make headway on their own.

Next the interface used by the tutoring module to provide information and access to the interaction the student wishes to perform is discussed. The learning environment which fits the tutoring style is discussed later in this chapter.

7.3:4 Interface

The interface allows the student to communicate and interact with the program using menus (in the menu bar and pop-up menus), windows and buttons to move from one interaction to the next, and to access information. Menus and buttons open interaction windows and dialogue boxes. Diagrams, simulations and dialogue boxes present information during interaction. Buttons and pop-up menus enable the student to input symbols and write equations, providing subscripts, superscripts and symbols which the ordinary key board does not provide. Examples of these are v_T^2 , ω_B , β and ϕ .

The interface also provides access to other applications and to use the supportive Macintosh environment through the *Applications*, *File*, *Edit*, *Font*, *Style* and *Size* menus (Figure 7.22). These menus provide choices of opening and closing files, printing and page set-up options, and editing commands. The *Cut*, *Copy* and *Paste* editing functions in the *Edit* menu enable the student to transfer information and working performed from one interaction window to another. The *Applications* menu allows access to a calculator option and different keyboard symbols.

7.3:5 Knowledge representation and access

Knowledge representation and access to this knowledge is now discussed. Representation and access can be described as internal or external; internal referring to the knowledge structure and process within the program, external referring to the presentation of the knowledge to the students. External access to knowledge uses the interface discussed in the previous section and will be discussed before the internal representation and access.

External representation: The external representation of knowledge uses diagrams, written descriptions and explanations, simulations and tutoring information. The diagrams present information in pictorial form (Figure 7.3), written descriptions provide qualitative information in the envisioning window (Figure 7.15), and written explanations are used to provide information explaining the functions and meanings of items presented in a window (Figure 7.25).

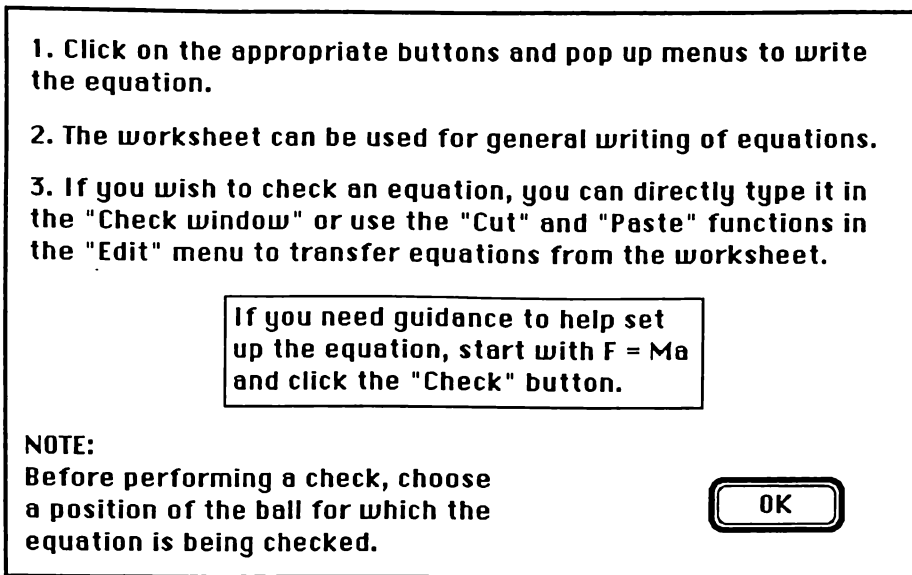


Figure 7.25: Information for commencing the *Check Force Equations* interaction.

Simulations depict the motion of the ball (Figure 7.19), and can be used empirically to obtain the minimum starting height of the ball for rolling, as well as sliding motion.

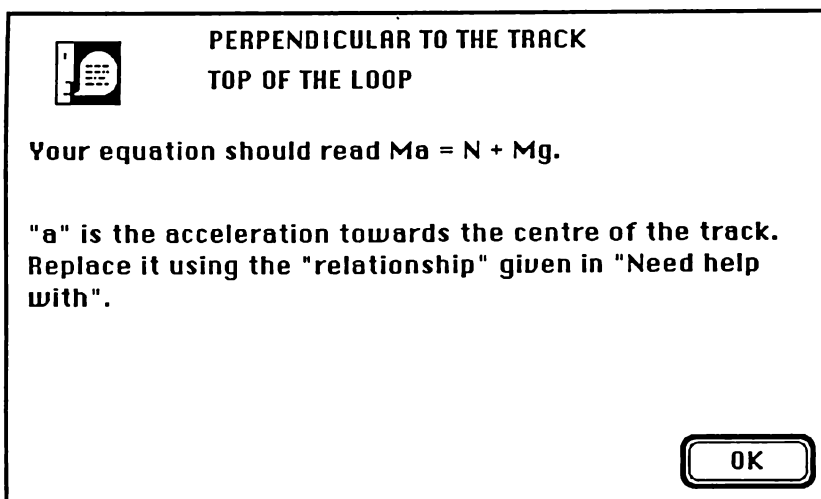


Figure 7.26: Dialogue box presented when performing a force equation check.

Tutoring information uses dialogue boxes (Figure 7.26) structured so that the top right hand corner presents information to confirm the purpose of the interaction. In the example, the student is informed that the force equation is being checked for the top of the loop in the perpendicular direction to the track. The first line of the message (i.e. Your equation should read $Ma = N + Mg$) affirms the information presented by the

student, since the program does not check for negative or positive signs. The dialogue has been formatted so that students have to compare the information supplied with that presented (if $N - Mg = Ma$ had been written, the onus is on the student to notice the difference). The remainder of the information provides a simple explanation, suggests a possible action, and provides cues on access to further information.

1	Position	At the bottom of the loop
2	Motion	Rolling
3	Energy	Rotational kinetic
4	Mass	
5	Square of radius	
6	Square of linear velocity	
7	Numbers supplied	
8	Angular velocity	
9	Moment of Inertia (I)	

Table 7.1: Frame for the rolling motion of the ball at the bottom of the loop.

Internal representation and access: Information is internally represented using frames as detailed in Chapter 3 where the example is for the total energy of the ball at the bottom of the loop (Tables 3.4 and 3.5). A further example of a frame for the rotational energy of the ball at the bottom of the loop is given in Table 7.1. First the information supplied by the student is used to fill in the appropriate slots in the frame. Rules are then used to access information in the frame, and to determine information not supplied or incorrect. The student is then appropriately prompted, using a dialogue

box similar to that given in Figure 7.26, until all the information slots are filled. If information for a non-related slot is supplied the student is accordingly prompted.

Some slots require information from other slots. For example, slot 9 uses information from slots 4, 5 and 7 to obtain the moment of inertia I , of the ball about an axis through its centre. Rules are used to prompt the student for more information when required. Figure 7.27 gives the rule used to prompt the student and the dialogue box used to present the information. The rules used to access information and to provide prompts follow the forward chaining procedure (Chapter 3).

IF	Numerator is 1 Denominator is 2 Moment of inertia is I Angular velocity is ω_B^2
THEN	Request for the substitution of the moment of inertia I

Figure 7.27: A rule and associated dialogue box to prompt the student.

The next section describes how the program takes into account the problem-solving processes and strategies of the participants.

7.4 INCORPORATING THE PROBLEM-SOLVING STYLES

This section describes the inclusion into the program of the 14 problem-solving processes as described in Chapter 5, and highlights how the problem-solving strategies are incorporated.

7.4:1 The activities performed

The 14 activities which the participants performed during problem solving are now discussed to show how they have been taken into account in the program development.

Analogous reasoning: This has not been directly taken into account by the provision of specific examples, however, the purposes for which the analogous reasoning was used: for determining the forces and energies at the top of the loop; has been included (Figures 7.10 and 7.11). Analogous reasoning was also used in attempting to understand the motion of the ball. The envisioning window (Figure 7.19) provides this.

Checking: Checking the logic of the steps performed is included in the interaction process when clues and hints to correct or change the information are supplied. Checking the answers has been provided through the *Check Answer* window (Figure 7.21), and through the CHECK buttons in the *Forces Equations*, *Energy Expressions* and *Perform Solution* windows. An example of an answer check for the force equation is given in Figure 7.7. The *Perform Simulation* window can also be used to check the final expression for the starting height of the ball.

Clarifying: This has been included through the dialogue boxes. In the example in Figure 7.26 it is made clear to the student that the equation should read $N + Mg = Ma$. If the student had supplied the correct equation then reiteration clarifies to the student that the interaction is proceeding according to plan. The next possible simplification of the equation, or the solution of the problem, is also clarified through clues given in the dialogue box.

Declaring quantities: This is included when providing clues and corrective information to the student. For example, when the student provides no value or zero for the translational kinetic energy of the ball at the top of the loop, then it is stated that the velocity is not zero at the top of the loop. If the student supplies the wrong symbol for representing a relation or quantity this is also stated. For example, if the student supplies the expression $\frac{1}{2} Mv_T$ for the translational kinetic energy of the ball at the top of the loop then it is declared that the wrong velocity symbol has been used. In this case, the correct symbol would be v_T^2 .

Pictorial representation: This facility is provided in the program in the *Question* window (Figure 7.2) to show the diagram of the track, to provide force diagrams, and for performing simulations. Static force diagrams are supplied using menu choices for the *Force Equations* and *Perform Solution* windows (Figures 7.6 and 7.16). Interactive drawing of force diagrams and acceleration is supplied by the *Force Diagrams* window (Figure 7.10). The resolved forces can also be drawn. Simulations are provided by the *Perform Simulation* window (Figure 7.19).

Resolving difficulties: The program is designed to resolve difficulties by providing a variety of information. When the student makes an error information is provided to address the difficulty encountered. The information could be a hint on where to obtain more information to make a substitution (Figure 7.27) or a statement pinpointing the mistake. This information is presented using dialogue boxes.

Qualitative analysis: This is provided in the *Envisioning* window (Figure 7.15), when supplying explanations for the clarification of terms (Figure 7.3), and when clarifying the clues given for the solution.

Qualifying: This is used to provide supportive information to the student, usually in the form of qualitative explanatory information. Examples of this include explaining the constraint $N > 0$ for the ball to stay in contact with the track, explaining the effect or role of the frictional force for the rolling motion of the ball, and explaining how the final expression for the energy of the ball is obtained.

Quantitative representation: This is catered through the provision of a keyboard and symbols which can be used to represent the physical quantities algebraically, and to provide expressions relating physical quantities. The internal representation uses this mode of storing information for checking and providing information (Table 7.1). Quantitative representation is also used to provide the final expressions to check working.

Question reading: This is included using the *Question* window (Figure 7.2).

Recapitulation: This has been incorporated in the checking mechanism and is dominantly used when checking energy expressions to inform the student of the correctly denoted energies and those requiring correction. Recapitulation is also used in encouraging consideration of the energy expressions once the force equation has been correctly written and vice versa.

Reference: This activity is included by providing expressions and relations, clues to write the force equations and energy expressions, and to substitute the condition $N = 0$ to obtain the minimum starting height of the ball.

Relating quantities: Quantities are related using algebraic expressions and previous working and knowledge. Windows for the different interactions can be accessed using the Window menu to use previous working performed. Previous knowledge has been incorporated into the internal structure of the program using frames and rules.

Symbols: These are provided to quantitatively represent the physical quantities (for example v , v_T^2 , ω_T , F , β), to supply information, and to perform checks on information supplied.

The 9 windows supplied by the program (see main features discussed earlier in this chapter) allow the student to move freely from one window to another and to perform

the 14 activities in random order. This fits the problem-solving flow charts of the participants (Chapter 5).

Next the incorporation of the problem-solving strategies is discussed.

7.4:2 Problem-solving strategies

The structure of the program is related to the specific problem-solving strategies of problem decomposition, forward strategy, means-end analysis, generate and test, envisioning and analogy.

The subdivision of the program into windows for writing energy expressions and force equations supports the problem decomposition strategy, through the choice of rolling or sliding to subdivide the solution of the problem.

The internal structure of the program encompasses forward strategy, generate and test, and means-end analysis. Rules use the forward chaining principle with one leading to the next. Forward strategy is used to gradationally lead to the solution. Means-end analysis is used to provide information leading to the solution to help bridge the gap between the current knowledge state and the final solution state. Generate and test is incorporated in enabling information to be placed into the slots to determine the best response from the program. The student could use the information or clues suggested by the program or type in further information independent of the advice.

A window has been specially supplied to help the students in qualitatively envisioning the motion. The simulation window can be used to envision the motion of the ball. The force diagrams window provides help in envisioning the forces and the acceleration.

Analogy is used when similarities to circular motion are mentioned and applicable circular motion relations are used. Although analogous reasoning has not been explicitly included, information students sought through reference to examples has been included.

Next the extent to which the program provides and supports a constructivist environment is discussed.

7.5 LEARNING ENVIRONMENT PROVIDED BY THE PROGRAM

In this section the discussion focuses on the features of a constructivist ICAI system (Chapter 3) included in the program. A summary of the 7 features discussed in Chapter 3 are:

1. a facility enabling the student to be an active participant.
2. enabling the student to make learning decisions.
3. being like a knowledgeable teacher.
4. being able to challenge and discuss ideas to deal with contradictory previous knowledge.
5. compiling a student model during interaction.
6. making explicit the links to other topics.
7. provision for the student to apply the ideas developed for consolidation .

The provision of 9 windows enables students to choose the preferred interaction and to be an active participant. As mentioned earlier, the student does not have to follow the advice given by the program or follow the steps suggested and is able to move from one window or interaction to another through the use of the *Windows* menu. The information needed can be accessed through various windows. Information supplied by the student can be transferred from one window to another using the *Copy*, *Cut* and *Paste* functions supplied in the *Edit* menu. The student is able to make learning decisions such as using the program to:

- clarify ideas and obtain help.
- obtain help in solving the problem through using the energy, force or solution windows.
- perform a simulation to understand the motion.
- obtain a qualitative description.
- practice applying the energy principle and writing energy expressions.
- practice writing force equations.
- check the answer to the problem.
- use the environment provided to perform the solution.

The information used by the program to provide help has been obtained after extensive data collection on conceptions, mistakes and knowledge of students. This enables it to provide help like a knowledgeable teacher.

The program provides minimal help allowing the student to remain in control of the situation. The student has to reconcile the difficulty encountered using the information supplied. The student is also provided the option of pursuing further help if unable to reconcile the information supplied to obtain an understanding of the situation. To an extent the program challenges the ideas of the students by identifying mistakes to help them reconcile contradictory ideas.

The program compiles the interaction of the student but this is not used to govern the interaction. It is used to determine the features of the program which are used and for program evaluation. The student model is included in the program structure and it takes into account the mistakes and conceptions of the students.

The Force Diagrams, Check Force Equations and Check Energy Expressions windows enable access of information for application to related problems for consolidation.

Links are made between the different windows, between the force equations and the energy expressions to obtain the solution, and between different physical quantities. Links to other topics are not made since the program is for a single problem. It has not been prepared for the teaching or learning of different topics.

Except for 7, it is felt the features a constructivist based ICAI system should have are included in the program to some extent. It is premature to include links to other topics at this stage since the current aim of the program development was to help the students to solve the selected problem. The links to other topics are provided in appendix C which gives all the relevant problem-solving information. This information can be used to develop the program further for making links to other topics for the teaching and learning of rotational dynamics.

7.6 CONCLUSION

The program has been described in detail and its salient features highlighted to show the variety of purposes for which it can be used, the main purpose being to solve the rotational dynamics problem. The structure of the program has also been related to the problem-solving activities performed during the data collection. The program has been

analysed to determine the extent to which it provides a constructivist learning environment.

In the next chapter, the data collected when students solved the problem for which the program was developed and a related problem is presented and analysed. The analysis is used to evaluate the program as well as the aims of the study.

CHAPTER 8

ANALYSIS AND EVALUATION OF DATA ON PROGRAM USE

8.1 INTRODUCTION

This chapter analyses the data collected while students solved the selected and a related problem (Chapter 4), a questionnaire completed after attempting each question, from interviews about these questionnaires, from observations of the problem-solving, and from the record of interactions with the program. General information on student use of the program is first presented. Then information obtained from the program evaluation questionnaire and substantiated with interviews is presented to show the:

- general student opinion of the program.
- difficulties encountered in using the program.
- features the students liked.
- features students would use.
- limitations of the program.

Next the information obtained from the second questionnaire on the related problem is discussed to show whether the students found it difficult, how the program helped them and the similarities they perceived between the two problems.

While the students were solving the problems, they were observed and a record of their interactions was compiled by the researcher and the program. The students were not aware of the program compiling a record. This data is analysed to highlight the

- features used by the students to obtain assistance.
- extent to which the students were able to recall information.
- difficulties encountered and conceptions portrayed.
- differences between the transcripts compiled by the researcher and the program.

The chapter concludes with a discussion of the research aims of the study (Chapter 1) in relation to the data analysed.

8.2 GENERAL INFORMATION ON PROGRAM USE

The students first attempted the problem using pen and paper. When they felt ready to start consulting the program they were shown how to commence and taken through the features of the program. It was suggested that they initially browse through the program to familiarise themselves with it (the 4 girls browsed through each window meticulously whereas the 14 boys were in a hurry to commence interaction). Any queries students had about the program were answered by the researcher throughout the session. No session lasted more than 2 hours, inclusive of attempting the question on paper, browsing through the program and performing the solution.

During the interaction with the program it was noted that the bulk of the students were interested in solving the problem to obtain the answer. They were not very interested in the other information which the program offered. Upon successful completion of the solution to the problem, the students enjoyed themselves by observing the simulated motions of the ball.

None of the students would have successfully solved the problem without the help of the program as was expected due to the difficulties faced by students during the initial data collection stage (Chapter 5). With the aid of the program, all 18 students successfully solved the problem. Students generally encountered difficulty in performing simple algebraic manipulations which the program did not perform for them.

To determine their opinions of the program and how it helped them, a questionnaire was administered. The information obtained is discussed next. The questionnaire is included in Appendix A6.

8.3 ANALYSIS OF THE PROGRAM EVALUATION QUESTIONNAIRE

The responses to the questions in section A provide the general student opinion of the program features. These are discussed first followed by the responses to each of the questions in section B. Excerpts of the student answers are also quoted. The excerpts are labelled F1 to F4 to denote answers of female students and M1 to M14 to denote the answers provided by male students.

8.3:1 General student opinion of the program features

Statements	Response				
	5	4	3	2	1
1. The program was easy to use.	0	12	5	1	0
2. I was able to get all the information required.	5	10	3	0	0
3. It was easy to access the information needed.	1	8	9	0	0
4. The program helped me to learn how to solve the problem.	9	6	2	1	0
5. The program was flexible to my needs.	2	8	8	0	0
6. I felt in control while interacting with the program.	1	9	5	1	1
7. The hints given by the program were helpful.	13	5	0	0	0
8. The simulations provided were helpful.	14	2	2	0	0
9. The diagrams provided were helpful.	8	6	2	2	0
10. The envisioning provided was helpful.	5	6	3	3	1
11. The answer check was helpful.	13	5	0	0	0
12. The solution provided by the program was helpful.	6	7	3	2	0
13. The program helped me to better understand rotational dynamics.	4	7	7	0	0
14. The program provided too much help.	0	0	3	13	2
15. I would consult the program for other tutorial questions if it was possible.	4	11	3	0	0

Response 5: Strongly agree - 1: Strongly disagree

Table 8.1: The number of students selecting a response for each statement.

Table 8.1 summarises the responses of the students. The individual responses are given in Appendix E1. A general discussion of the responses is next presented.

Items 1, 2, 3, 5 and 6 provide information on the use of and access to information. Only 1 student felt the program was not easy to use and 2 did not feel in control while interacting with the program. These 2 students had difficulty solving the problem and attempting to connect the information supplied by the program. Nine students felt that information was easy to access while the other 9 were undecided, giving a response of

3. Ten students felt the program was flexible to their needs while 8 gave a response of 3.

The responses to items 4 and 7 on the helpfulness of the program in providing hints and learning is encouraging. All 18 felt the hints provided by the program were helpful and 15 felt the program helped them learn how to solve the problem. This was expected as students were unable to solve the problem without the help of the program. However, it is encouraging to note that the students felt they had benefited from the interaction with the program.

Items 8 to 12 provide opinions on the main features of the program. Two students did not find the diagrams very helpful, 4 did not find the envisioning helpful, and 2 did not feel the solution provided by the program was helpful. The single student who indicated the response of 1 for envisioning did not use the facility. Generally it can be concluded that the features were found helpful by the students.

Only 3 students were indecisive as to whether the program provided too much help. This is encouraging as the program was developed to encourage students to think. Feedback from student F3 is given below indicating that too much help was provided. She is an above average grade student and did not have difficulty using the program to solve the problem.

I felt the program did most of the work. This was with the help given and I did not really need to think. I think the help section could give a little help and then if you need more help, have another in depth help section.

(F3)

When asked whether they would consult the program for other tutorial problems if possible, 15 students responded favourably and 3 were unsure. This response should be discussed in context with the:

- need to consult such a program.
- perceptions of the students as to whether they benefited.

This assumes that they wish to solve the problem or have to solve it as part of their course work. Item 13 gives the perceptions of students as to whether they benefited. Eleven felt they had been helped while 7 felt they had somewhat benefited. If they perceived benefit then there is a greater chance of consulting a program for further problems. The response to item 4 indicates 15 students felt that they learnt how to solve the problem. Comments of students in response to other questions discussed

later in this section support the view that the students would indeed consult the program.

In the next section the difficulties encountered by the students in interacting with the program are discussed.

8.3:2 Difficulties encountered while using the program

In response to the question *List any difficulties you encountered while using the program*, unfamiliarity was mentioned as the major difficulty. However, as students used the program difficulties encountered reduced. The excerpts of 14 student answers given below directly refer to unfamiliarity with the program:

Without the manual, the program was a bit confusing. (F1)

Initially getting familiar with the program in going from one part of the program to the next. After a while it was okay. (F2)

Finding out how to begin answering the question. But once I had started and had got used to the method of checking every step it became clearer on what to do. I found it hard to work out which window to work out the formulae in but then I realised it didn't matter as I could transfer the formulae to another window. (F3)

The main difficulty I had with this program was initially setting up the equations ie knowing what equations went where, what equations do I use. Once this was overcome manipulation was easy. (M1)

I had trouble when trying to decide which section I should next refer to. Also, it took me the full time to properly get used to the format etc. There was not really much explanation by the program as to what I was trying to find. (M2)

Remembering to select "position of the ball", recording equations etc. Moving through the program, changing from a window to another was difficult. (M3)

Getting used to the program took a while as I was not sure quite how it worked to start with. (M5)

The start panel could be a little more helpful but overall it was pretty good. I was at a bit of a loss once I had my 2 expressions as to what to do next, maybe the program could prompt you along a little more to obtain the final expression. (M6)

How to get started. Although force equation, energy equation and simulation were adequately helpful only the solve problem feature was helpful in actually solving the problem. (M7)

It took quite some time to begin to understand what some of the questions actually wanted from me. It was a matter of trial and error before I could work in union. (M8)

No difficulties encountered except in learning how to use the program. (M9)

Knowing exactly what to do when the expression entered into the *Energy expression* in the solve question window was incorrect. Knowing whether position 1 and position 2 in the solve question window were interchangeable. I didn't expect Start position to be at position 2. (M10)

While using the program I discovered several difficulties on how to start and use things. Without your help I found it difficult to use or discover the help functions available to me to be used. (M11)

It took a while to get used to. A manual would be helpful. (M14)

The other students mentioned unfamiliarity indirectly by suggesting inclusion of features already present in the program. An example of this is given below. The student did not realise that the external keyboard could be used to input the numbers.

The user interface was a bit awkward for inputting formulae especially numbers were hard to input ie having to input formulae because it is hard to get at the numbers with the mouse. (M4)

The difficulties encountered by students have been addressed in a brief manual on the use of the program features (Appendix D1).

In the next section the features of the program the students liked are summarised.

8.3:3 Features which the students liked

This section presents the response of students to the question *What aspects and features of the program did you like?* Rather than summarise the responses, each of the individual responses for the 18 participants are given to provide a general flavour.

The help features and the use of buttons on the screen. (F1)

Simulations. Check for steps of working. Answer checks for the force and energy equations at different positions. (F2)

Having the simulation was very good as it allowed you to see what would actually happen if you set the experiment up. It gave you hints on where to start and what to do next. You could check each step you had done to see if you were on the right track or not. (F3)

Prompted but did not provide too much help. Unlike similar programs it encouraged one to think. Simulation was particularly good, allowing the user to see the effects altering various variables have to understand better the motion involved. Program was well organised and generally fairly user friendly. Description of variables was great. (F4)

The checking feature was useful and also the hints that this feature gave were very useful. The simulation aspect is a enjoyable way of discovering the effects the various variables have. (M1)

I especially liked the on screen help service. It was very well set up and really helped me with my calculations. (M2)

The check feature this enabled time reduction ie if formula entered was incorrect the use of check quickly highlighted the error, which is very useful. The simulation was very useful. (M3)

Help was good but sometimes a little inflexible. Good to differentiate between v , τ and other positions of the velocity and angular velocity. Like the way it starts small and builds up, helps understanding. (M4)

That it hinted of what I needed to do to get the required answers to the questions. That it had a simulation we could run to see that our answer worked. After getting used to it, it was relatively easy to use. (M5)

The perform simulation is very good. The way the check button gives you hints and helps you along is a very good idea. (M6)

I am not any good on computers so no aspect did I like. The program was very good but hard for me to get through. I felt safer using paper and writing my own equations. It seemed harder for me to find the file that solved the problem and how to use it, than it was to write and equate equations on paper. (M7)

The force diagrams were a large help in solving the problem. I thought the simulations were very good to help visualise what is happening once I had solved the problem. (M8)

Simulations provided excellent experimental demonstration of application. Simple instructions and hints were helpful. Windows and graphical user interface makes this user friendly. (M9)

Being able to constantly check the expressions being entered in all the different windows. The diagrams were also very useful, especially the force diagrams. I also liked the simulation part. As well as being useful it was also very interesting just to experiment with. (M10)

I liked the simulation as you could change all the different values involved and see the way the changes affected the ball rolling down the slope. Also once I discovered how to use the help with formula they were very helpful with the simplifications and the like. (M11)

Answer checking, force diagrams, simulations and help sections. (M12)

The ability for the program to tell you where you were going wrong. The simulation where you can see that your answer is correct. (M13)

The help features, the simulations. If you couldn't understand a problem you could enter a basic formula and it would coach you from there. Force diagrams. (M14)

Except for the response of M7, the rest were very encouraging. It is difficult to determine how much these feelings were due to novelty. At the interviews conducted immediately after completion of the questionnaire, the students were asked whether the

novelty of the program affected the feedback. None of them thought so. Liking the features of the program enhances the chance of them using it.

Next the purposes for which they would consult the program, if it was possible to do so, for other assigned question are discussed.

8.3:4 Purposes for which the program might be consulted

The students were requested to select the purposes for which they would consult such a program if it was possible to do so for other assigned tutorial problems. They were presented with the following 7 choices: to check the equations written, to check the solution to a problem, to consult the envisioning, to perform the simulation, to consult the force diagrams, to check the answer, to obtain hints to solve the problem, and the option of mentioning others as well.

The students could make any number of choices they wished. Their responses are summarised in Chart 8.1. One student was unsure whether he might consult the envisioning or the force diagrams. This has not been included in Chart 8.1. Four students mentioned some other purposes. These are given below.

To generally practice the problem.

To use the program to work through the problem and obtain an answer to it.

To obtain an understanding of the process involved in solving a problem.

To use the formulae given in the program.

Envisioning was the least popular choice (Chart 8.1). Next was consulting force diagrams, consulted by 7 students. When students were asked the basis for their choices two major criteria surfaced. These were:

The features which helped to solve the problem.

The features which helped to provide more information for understanding the solution.

These criteria explain why *Check equations written*, *Check solution* and *Obtain hints* were the popular choices (Chart 8.1).

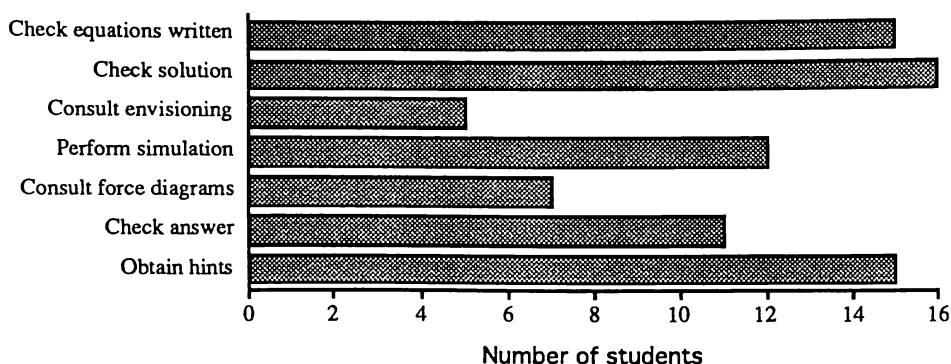


Chart 8.1: Student choices of purposes for which they would consult the program.

The students did not feel envisioning helped to provide insight for the solution of the problem. They were of the opinion that the simulation provided better envisioning and understanding of the motion of the ball.

For the choice of consulting force diagrams, the students were asked whether they had considered both types of force diagrams i.e. those supplied through the menus and the interactive force diagrams which could be drawn by accessing the *Force diagrams* window. Four of the 7 students who had not chosen *consulting force diagrams* indicated that they had considered the static force diagrams available through the menus and not the interactive force diagrams when providing the response. These 4 stated they would consult the force diagrams window as it helped to understand and resolve the forces. The other 3 stated they knew the forces acting and the force diagrams did not help much.

In the next section, the responses to the question *What are the program limitations you encountered during your interaction?* are discussed.

8.3:5 Program limitations encountered

No major concerns were mentioned by the students. This can be inferred from the responses of the students who provided feedback. These are given below.

The program does not rearrange equations. (F1)

It didn't do step by step mathematical calculations but went straight to the answer. (F3)

Would have been more useful if simulation was explained more, ie enlarging or reducing g, oscillation that occurs when $h = R$ in the program. (F4)

None, but possibly an example already written in the boxes on screen would help you to understand better. (M1)

There may have been too many screens. Possibly if they had been all placed on one screen and a scrolling facility used, it would have been easier to find my way around. (M2)

The use of *normal* algebraic expressions i.e. brackets, decimals etc. (M3)

Inflexible at times about the way you solve the question. Inability to square expressions in brackets may be limiting later on with other problems. (M4)

A little hard to use in some places but mostly alright. (M6)

The check in the solution was very systematic. It however did not keep up with what you were doing. When I left a file all of what I had done in it was gone when I came back to the file. (M7)

Mathematical expressions not simplified. Should include a wide variety of problems. (M9)

Not being able to enter v_T , v_B and so on from the keyboard. This is just something to basically speed up entering the equations. (M10)

With the question I solved, I did not find any limitations. (M11)

Speed of the windows opening and closing. Speed of the checks. (M13)

None, but more help would be helpful. (M14)

The feedback of F3, M3, M7 and M10 are considered.

F3 refers to the algebraic manipulations required to substitute the velocity expression from the force equation into the energy equation or vice versa. The students needed this to obtain the value for the minimum starting height. The program has intentionally been constructed not to perform these calculations to provide the student with the opportunity to practice some mathematical manipulations.

The comments of M3 result from the use of fractions in the program. Decimals were not included because fractions such as $10/7$ do not convert to simple decimals. Furthermore, during the equation checking, the numbers of decimal places would be a problem. Truncating the decimals would reduce the precision of the calculations. If the program is expanded to include further problems then this feature might be included depending on the problems.

In the case of M7 the window in which the information was typed was closed, causing the information to be lost. When it was next accessed it was opened anew rather than being brought to the front from the background. The student did not understand the difference between closing a window and allowing it to remain open in the background

but not visible. The student was at that stage not familiar with the cut and paste facility supplied by the program for saving working. Saving working was not easy as the information in each slot had to be separately cut and pasted. The program might need improvement to this facility for performing assignments. In its present form it can be used to produce a printable copy of the solution.

The concerns of M10 had to do with representing the symbols such as v_T or v_B using buttons. This would make them visible on the screen like the other symbols such as F and M. In the present format the velocity symbols are accessed through pop-up menus in the window and descriptions of the symbols in the menu items help students make the correct choice.

The other comments provided by the students do not require any clarification or discussion. None of the feedback provides any concerns for program use in principle.

Comments: The information supplied by students in response to *Please supply any information pertaining to the program but has not been covered by the questions* was either repetitive of information given for the other questions or merely general remarks such as:

Perhaps the question should appear first then prompt the user from which level s/he would like to start from. (M3)

Next the information obtained through the second questionnaire on the related problem is discussed.

8.4 ANALYSIS OF THE PROBLEM EVALUATION QUESTIONNAIRE

Only 14 of the 18 students attempted the related problem. One female and 3 male students pulled out of the study due to academic pressures. The 14 students given the problem to solve were told that they did not have to use the program. All 14 students consulted the program.

After attempting the problem they were given the problem evaluation questionnaire (Appendix A8) to determine whether they found it difficult to solve the related problem, how the program helped them and the similarities they perceived between the two problems. The 3 questions used to obtain the information were:

- Did you find it easy to solve the problem? Please explain.

- Did your consultation with the program help you in solving this problem?
- What similarities did you find between the problem you have just attempted to solve and the loop-the-loop problem you tried to solve using the program?
 - (a) When you read the problem.
 - (b) While solving the problem.
 - (c) After solving the problem.

The responses of the 14 students to these 3 questions are next presented and discussed.

8.4:1 Difficulty in solving the problem

The main responses of the 14 students are presented to show that all of them found the problem difficult to solve.

I found it hard to begin with, as I was stuck as to how to go about the problem, and that the two situations seemed unrelated, but it started to make sense as time went on. (F1)

I found it difficult to use the ideas from the other problem and use them to solve this problem. (F2)

No. Had problems starting, took equations at wrong points initially. Had difficulty interpreting data from diagrams realising some similarities. Had some difficulties adapting problem from first and second question. Unsure at what to take as height when ball leaves large ball. (F3)

No, I could not solve the problem. The reason why is that I relied too heavily on the help function in the loop-the-loop problem. I did not really understand the derivation of the force and acceleration components throughout, and still could not after using the force diagrams. I felt that I did not really know where to start in the problem. (M1)

No, I found it hard to relate the portions of the problems to each other, although finding the parts was easier. The other problem was finding what bits I needed and what I didn't. (M2)

No. Could not remember all of the physics equations, my maths was not so great. (M3)

The problem was quite hard but with a little thinking and a start from the computer (basic equations) it was alright. It helps to read the question and refer to the old question. (M4)

No. Assumed the ball would lose contact at 90 (wrong). Could not understand why N was not equal to $mg\cos\theta$ due to movement. Could find no connection between point on plane apart from top and bottom

in terms of forces or energies at these points seemed a very non fixed value. In terms of rolling I could not understand why no extra forces were involved and that only velocity changed due to rotational energy. (M5)

At first glance the problem looked very difficult. I had problems producing the relations needed to solve the problem and it was after thinking, writing down various facts and acquiring equations from the program that I managed to get a suitable solution. The biggest problems I had were finding force relations for different points in the loop, and eliminating one of the unknowns for the final expression (which were general physics and mathematics problems). (M6)

Not really. Problems encountered included difficulty in determining force expressions although energy expressions were easier. Application of the model from the previous problem was a great help, but the algebra involved was a killer. (M7)

No. It took a while to spot some of the differences between the problems. How the forces changed when they were on the inside and outside of the circle. The main difficulty I had was the minus sign in the force equations. (M8)

I knew I had to use the top to write the energy equation. I did not know which other point to take. I remembered from the last problem that I needed two points. I used the program to help me. (M9)

I found it difficult to use the ideas from the other problem and use them to solve this problem. (M10)

No. It took a while to work out how to use the program to your advantage, once this idea was latched onto this simplified it a lot. When you were not using the computer to your advantage it was difficult. Also the algebra was a challenge. (M11)

The comments of M1, M5 and M8 were further clarified during interviews. M1 was referring to the solution of the previous problem where he used the information from the problem and did not really understand it. He realised this when he tried to apply the knowledge to the related problem.

M5 had the conception that the marble would lose contact with the sphere when it had travelled a quarter of a circle at the position where its radius vector made an angle of 90° with the vertical (Figure 8.1). Another conception portrayed was that since there was no movement towards or away from the centre of the circle there cannot be any acceleration in that direction. This meant that the forces (N and $mg\cos\theta$) in opposite directions should equate (Figure 8.2). These are discussed in detail in a later section.

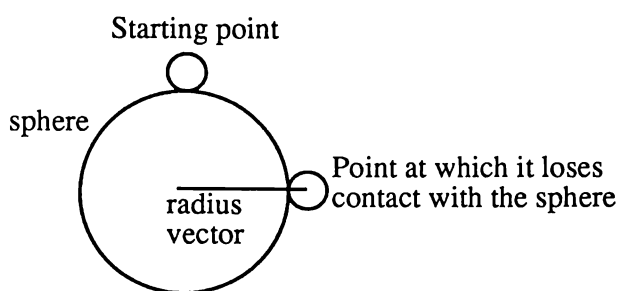


Figure 8.1: Marble loses contact with the sphere when it has travelled a quarter of a circle along the sphere.

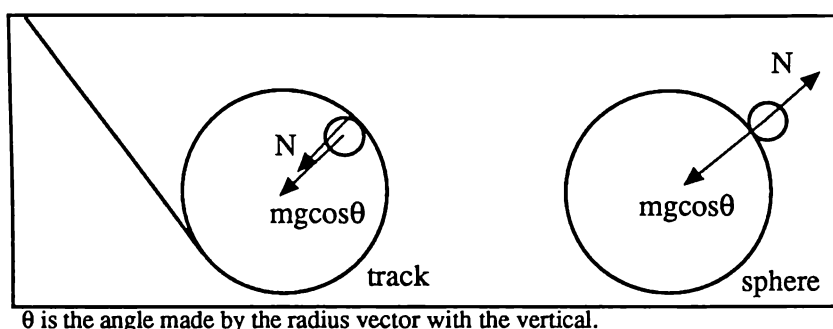


Figure 8.2: The forces in the radial direction for the ball moving in the track and the marble on the sphere.

M8 had difficulty resolving the change in direction of the reaction force of the sphere on the marble N . In the first problem the reaction force of the track on the ball acted towards the centre. In the related problem the reaction force of the sphere on the marble acted away from the centre (Figure 8.2). For the case of the track the force equation in the radial direction gave $mg\cos\theta + N = ma$ while in the case of the related problem the equation was $N - mg\cos\theta = ma$ where a is the acceleration towards the centre.

The comments of the 14 participants confirms the choice of a problem of reasonable difficulty. If the students found it easy they would not feel the need for consulting the program. At the interviews the students were asked whether the problem was too difficult. None of the students felt it was so but 3 felt that the problem was not really similar to the previous problem.

Next the responses to the question *Did your consultation with the program help you in solving the problem?* are presented and discussed.

8.4:2 Help accorded by the program

All except 1 female (F2) and 2 male (M1 and M3) students felt that the program helped them to obtain information for the solution of the problem. Of the other 11 students, 8 managed to successfully solve the problem, and 3 who were unsuccessful felt the program had helped them. The responses of the 14 participants are given below to show the majority felt the program was helpful.

Yes as the force diagrams helped me to write equations to solve the answers. The program gave me clues as to how to alter my equations. (F1)

No, I could not see the similarities even when I used the program. Did not know how the problems could be the same when the two starting points and ending points were different. (F2)

Yes diagrams were useful (particularly those such as force diagrams) though I did not look at this properly first time. Program was fairly adaptable to new situation because it explained at each step what variables were e.g. when $N = 0$. Program would have helped more if force energy equations would have been equated for different question but presumably program will later be general. Help with algebraic parts would have helped although this is not essential, it would avoid mistakes and misunderstanding. Simulation helped a lot again as did the information the program could give on the motion in problem 1 as this information was applicable to problems 1 and 2. (F3)

No not really, apart from being able to gain access to the force and acceleration components at different quadrants in the loop I could not really see the similarities between these two problems, so could not effectively use the first problem to help me with the second one. It seemed to me that all the equations given were backwards due to the force of N acting toward the centre of the loop. (M1)

It was helpful in that it allowed me to see the similarities between the problems and find out portions of the equations that I felt necessary. It also helped me decide what I needed and what I didn't. I found using the force equations check very helpful. (M2)

Yes but only to look at force diagrams and find the equations I needed. I only required background information and did not use the program to work anything else out. (M3)

The program helps by giving you the basic equations and hints (only slightly) at the correct path. It made solving the problem easier but also gave me a way of starting as often I start wrong and have little confidence. (M4)

I understood that the forces did not cause the ball to leave the slope. But the induced velocity component caused this. Found out y could be found in terms of the changing angle in equation. The rolling of a ball only caused internal change in mass positions, no longer considered as a point. Program did not offer sliding or rolling window on forces, so assumed sliding forces = rolling forces. I found that angle

placement when drawing force diagrams is critical to calculations.
(M5)

The program gave me an insight into how the forces related to each other, and also, the diagrams showed the forces present. I used that to verify my assumptions which was very useful. I also used the energy expression part to derive an expression for the energy of the rolling/sliding ball and therefore an expression for v^2 which could be substituted into the force expression acquired previously. (M6)

Yes. Force and energy expressions were derived from the relevant sections which I would not have been able to do without the computer (particularly the check). The loop simulator was required for determining forces/directions similar to those in the question. (M7)

Yes it helped me work out some of the formulas and the force diagrams were helpful in getting to grips with the problem. Again it took a while to pick up some of the differences between the problems e.g. the position points when using the force and energy bits. (M8)

Yes it helped to write the energy expressions and remind me of $N = 0$ when ball leaves the track. The force diagrams were very helpful. (M9)

Solving the problem again on the computer helped me to see how the equations could be used for this problem. The force diagrams showed me the forces were same but N had a different direction. (M10)

Yes you could see how to solve the first, then using the same technique solve the second. The program also checked your work, giving hints what was wrong and the program equated the formula (i.e. cut out a lot of unnecessary formula rearranging). (M11)

The above responses support the data in Chart 8.1 on the purposes for which they would consult the program. It is possible that students might say that they would consult the program but not do so when the situation arose. It was encouraging to note that all the students consulted the program.

A text was not supplied during the session for the related problem. No students requested the text. At the interviews they were asked whether they would have preferred to consult a text. No student felt the text would have helped as much as the program did. Their answers implied that the text only helps when information on relations and expressions is required and not on information for solving problems. They felt that the program provided both aspects, and suggests that students would consult a program if it was available for other problems.

Another positive aspect of the program was the help given to students at a loss on how to start. Six students stated that the program helped them to focus on aspects of the information which they did not understand. They felt this would help them ask their tutors for information rather than saying they did not know how to solve the problem.

In the next section, the similarities observed by the students prior to attempting, while attempting, and after attempting the solution are presented and discussed.

8.4:3 Similarities between the two problems

This section reports similarities the students perceived between the two problems. This was explored to determine the transfer of learning when solving the related problem. The level of transfer is discussed in a later section. First the similarities perceived by the students prior to attempting the solution are presented.

The energy and force equations would be similar. The force vectors would be different because the little ball was on the outside of the loop. (F1)

Nothing apart from the presence of a large circle and a small circle. (F2)

Realised force equations would again be the same for sliding or rolling. (F3)

The main difference was that at the top of the loop in problem 1 the ball was moving, whereas in the second it was stationary. (M1)

They both had 2 different sized balls in them. They both had to do with points at which contact was broken. (M2)

Realised that the force diagrams (just up the other way) for most of the equations would be the same but I did not think the program would help all that much as it gave the answer and all the equations relating to the answer for the first question. (M3)

The first thing I realised was that R was in both and the $F = N - Mg$ (or $+$) relationship was in both. The thing we were looking for was different though. (M4)

That the initial height has the same energy values at the top of the circle. In problem 2 the ball rolls on the outside instead of on the inside. (M5)

There was a rolling ball and a larger object to roll in or around. That we had to take account for the radii of both the larger object and the small one. The small ball started from rest. (M6)

Both questions involve a solid ball rolling and sliding on a circular surface. The main similarity was the factor of whether or not the ball would leave the surface and whereabouts this might occur. (M7)

They both had an interaction between a small and a large circle. They both have rolling and sliding. (M8)

The energies involved would be the same. The force equation would be quite similar. (M9)

In both cases the ball started from rest and there was a top point in the loop or large ball. (M10)

That they both involve one large and one small circle. (M11)

Apart from 5 students (F1, F3, M3, M4 and M9) only surface similarities were perceived. This is coherent with the reports of other researchers that novices classify problems using surface features (Chi et al, 1981). The 5 students who did not use surface features successfully solved the problem.

Next the similarities noticed by the students while attempting the solution of the problem are presented.

Similar but some vectors were reversed. (F1)

There are energies and forces involved in both. (F2)

Realised energy at the top of loop in problem 2 is the same as in problem 1, but that there was no translational kinetic energy at top of loop in problem 2. Could adapt force components for a position “generally in loop” for problem 2 by seeing motion was in opposite direction. Saw that $0.81\cos\beta - R$ could be substituted for y but only when diagram was drawn. (F3)

The basic force diagrams could be used for both problems except for the small problem of reversing the direction of N . (M1)

Both used force and energy equations. They both used normal force and what happened when N was zero. (M2)

The steps used to get the final answer were the same so I could remember back to the last time when I used the computer to figure out what to do. (M3)

The force equation as in the first does not change for sliding or rolling. The approach to find the velocity (as in both) both uses the same energy approach to find expressions, then an equation. (M4)

The normal force was altered. Made equal to zero when ball left surface. Energy at the top of circle in 2 = energy at top of circle in 1. (M5)

That the normal force N was 0 at some stage during the problem. Gravity was unimportant (assuming there was any). There was a $10/7$ and a $2R$ in the energy expression. (M6)

Both questions had similar energy expressions (although forces varied) which I expected due to the nature of the experiment. I found the motion inside the loop not too far different from the motion expected outside the loop which helped a great deal. (M7)

The similarities between the force involved, N and the perpendicular forces. (M8)

The energies involved were exactly the same. The force equation had N reversed but because it became 0 it did not matter. So the force equation was the same except now it was not for the top of the loop but for a point on the side of the sphere. (M9)

Energy equation and force equation had to be used but the points at which this was done was different and this confused me. If the points had been the same then the problems would have been similar. N kept appearing in the equations for the force. Couldn't remember why it did not appear last time. (M10)

That you used the same formula and worked through it the same way with the same component form of forces. The normal components are opposite but when the small circle leaves them both $N = 0$. (M11)

Comparing the comments on the similarities observed prior to and while attempting to solve the problem shows that the program helped the students by focussing attention away from surface features. The students used physical principles and quantities, as well as solution steps, to compare the questions. This is encouraging as it implies that they shifted from using surface features. Although they moved beyond surface features 6 of them still did not solve the problem.

Next, the similarities perceived by the students after they had attempted the solution are presented. The responses of the 8 students who solved the problem are reported since those unsuccessful in solving the problem made no new comments.

They are the same in the fact that you use the same technique, same formula and the same reasoning to work through the problem and get an answer with similar variables. (F1)

Realised only when helped why $N = 0$ when ball leaves loop because I did not initially read help properly. Saw that the loop part of problem 1 motion no different from loop part of problem 2 motion, just that a point in the loop needed to be taken in this problem rather than when $N = 0$, the angle when little ball leaves path could be worked out. (F3)

I could have used the program to calculate a lot of my expressions. (M3)

As in the first the sliding requires less of the component for the same reason, there is no friction and no energy loss. Escape velocity reached sooner (or minimum height for the ball to stay on the track). Therefore, rolling ball requires more energy to attain escape velocity and the angle is greater so more time on the sphere. (M4)

Same energies involved i.e. rotational, translational and potential. The moment of inertia expression was the same. The factor $10/7$ kept appearing in the expression. The problem is not as hard as it appears. (M5)

As with the first problem, both answers for rolling and sliding were not hugely different. (M7)

How the start point in the loop-the-loop one can be related to the top of the sphere, and how the point in the loop and a point on the outside of the sphere. (M8)

Both use the force equation at the point where the ball loses contact. Energy equation is written for the start and the point of losing contact. The condition $N = 0$ is applied for loss of contact and the two expressions combined to obtain the answer. (M9)

Only F1 and M9 explicitly mention the solution strategies as the similarities between the two problems. M5 and M8 imply such information, and when asked at the interviews they described the strategies.

Next the interaction records of the students compiled by the researcher and the program are discussed and compared.

8.5 INTERACTION RECORDS COMPILED

In this section the transcripts of the interaction records compiled by the program and the researcher are used to highlight the:

- features of the program used to help solve the problem.
- recall and use of information in attempting the solution.
- difficulties encountered and conceptions portrayed.
- differences in the transcripts.

8.5:1 Features used to help solve the problem

Only 4 students requested clarification of features of the program. The others knew how to access features, or discovered them through using the menus and buttons. Four performed stepwise solution of the first problem before commencing the solution to the related problem. Six others partially solved the first question as well. Chart 8.2 gives a summary of the number of students who used the different features of the program.

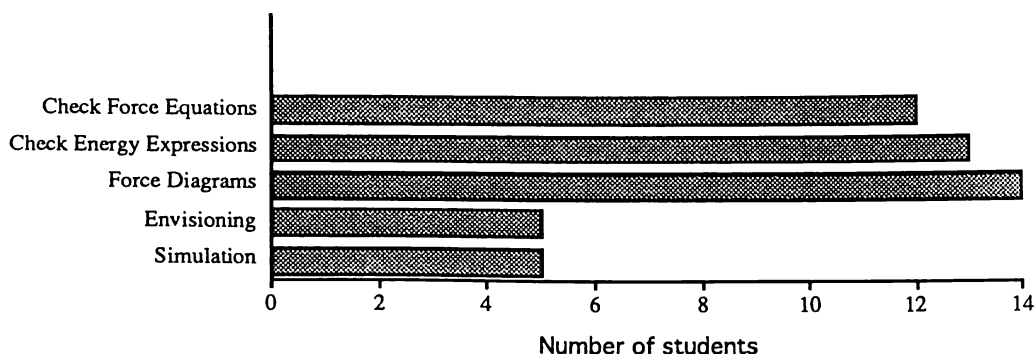


Chart 8.2: Features of the program used by the students to solve the related problem.

Interactively drawing the force diagrams, checking the energy expressions and obtaining the final expression, and checking the force equations were the most used features. Envisioning and simulation were each used by 5 students to solve the previous problem again. None of the students used envisioning and simulation for the solution of the related problem because they were specific to the previous problem. In the case of the force diagrams, energy expressions and force equations the students could write these for different positions of the ball.

For the force diagrams and equation writing the students either knew they had to consider the forces at *A general point in the loop* or they experimented with different positions of the ball until they hit upon the correct position of the ball at which to perform these. They then attempted to adapt the force equation and apply the information to the solution of the related problem.

For the energy expressions students again experimented or knew which positions of the ball to consider. Two main methods used were:

Writing the expressions for the top of the loop and a general point in the loop.

Writing the expressions at the start and a general point in the loop.

In the first case the velocity at the top of the loop was substituted as 0 to adapt the motion of the ball in the previous problem (Table 8.2). In the second case the energy expression obtained for the motion of the ball in the loop was similar to that of the motion of the marble on the sphere. No modifications to the expression were needed in this case.

$v^2 - v_T^2 = 2g(2R - y)$	(Wrote the equation for the top and a general point)
$v_T = 0$	(Adapted the top of the loop as the start)
$v^2 = 2g(2R - y)$	(Obtained the final expression)

Table 8.2: Equating the energies at the top and a general point and adapting it.

Next the information recalled and used to solve the problem is discussed.

8.5:2 Information used and recalled

Information used: Table 8.3 provides a summary of the information successfully used by the participants. This information was either recalled from previous problem-solving experience or accessed using the program. Only 8 participants successfully solved the related problem, 5 of whom could have solved the problem without the aid of the program. These 5 students also used the program to check their working. The solution steps to the problem are given in appendix E2.

All 14 students managed to obtain the energy expression at the start of the loop. Only 10 managed to obtain the energy expression for the point at which the marble leaves the sphere, in terms of the height (y) of the marble at this point. Those who determined the energy expression were able to obtain the force expression. These 10 had the rest of the information correct (Table 8.2). Two were unable to obtain the relationship between the height (y) of the marble and the angle of the radius vector with the vertical at the point of leaving the sphere (Appendix E2).

Information recalled: Most of the information used by the students to solve the problem was obtained from the program. A summary of the key information recalled by the students and recorded by the researcher is given below.

1. All the students recalled that the energy expressions had to be written for 2 points. Only 6 recalled that it should be written for the point at which the marble loses contact. The others realised this through information obtained from the program.
2. Twelve students recalled that they had written a force equation for the previous problem. From these 5 were aware of the point at which to write the force equation. In total 10 students wrote it correctly as shown in Table 8.2.

Solution steps	F1	F2	F3	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11
Energy at the start.	√	√	√	√	√	√	√	√	√	√	√	√	√	√
Energy at the point of losing contact with the sphere.	√	X	√	X	X	√	√	√	√	√	√	√	X	√
Equating the energies to obtain the velocity expression.	√	X	√	X	X	√	√	√	√	√	√	√	X	√
Force equation at the point of losing contact with the sphere.	√	X	√	√	X	√	√	√	X	√	√	√	X	√
The normal reaction substituted as zero into the force equation.	√	X	√	√	X	√	√	√	X	√	√	√	X	√
Combined energy and force equations.	√	X	√	√	X	√	√	√	X	√	√	√	X	√
Obtained relationship between y and the angle.	√		√	X		√	√	√	X	√	√	√		X
Obtained the final expressions for sliding and rolling.	√		√	X		√	√	√		√	√	√		X

Females: F1 - F3 Males: M1 - M11
 X: Incorrect information. √: Correct information. Blank space: Information not supplied or obtained

Table 8.3: The information supplied by the students for the solution of the related problem.

3. Only 5 students recalled that the condition $N = 0$ had to be applied to simplify the force equation. The other 5 obtained the condition from the program.
4. The 10 students who obtained the force and energy equations correctly knew that they had to substitute for the velocity to obtain the expression for the angle.
5. Nine students recalled that the force equation was the same for sliding or rolling motion of the marble.

Next the difficulties encountered by the students and conceptions portrayed are discussed.

8.5:3 Difficulties encountered and conceptions portrayed

The difficulties encountered by students included performing algebraic manipulations, determining the radius vector from which to measure the angle, not realising that for a general point in the loop the program gave the force equation for an acute angle, determining the direction of N , and determining the radius of the circle performed by the marble. Each of these are discussed in the order they have been mentioned.

Algebraic manipulation: Nearly all the students struggled with the algebraic manipulation required at the end. They had difficulty obtaining the algebraic expression relating the height (y) to the angle made by the radius vector with the vertical. They encountered further difficulty in simplifying the expression obtained for the angle after substituting for y . Some students took as much as half an hour to simplify the final expression and obtain the value of the angle at which the marble leaves the sphere. The excerpt below captures the mathematical difficulties encountered by students.

I felt I could not solve the equation mostly through my lack of good maths skills. I often have trouble with substitution in equations and also with trigonometry. I think the program helped me with this in some ways, but might have helped more in taking the equations further. It was useful to use. (M11)

Measurement of angle: Six students had difficulty realising that the angle could be measured from the start (Figure 8.3). They thought that the angle had to be measured from the radius vector passing through the bottom of the sphere. Four managed to reconcile this difficulty.

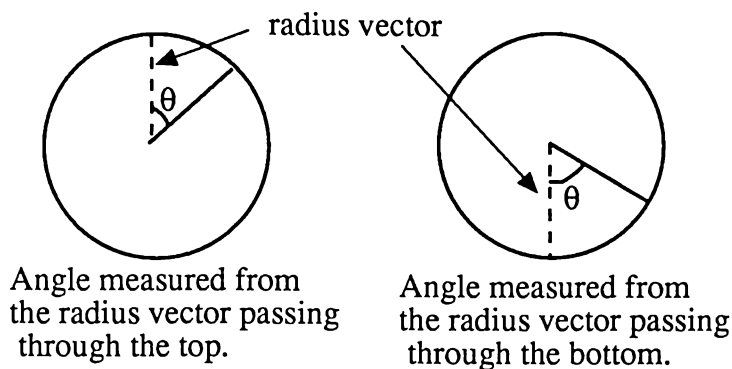


Figure 8.3: Measurement of the angle from the radius vector passing through the top and the bottom.

Direction of N: Four students had difficulty determining the direction of N, being unsure whether it was always towards the centre. Two resolved this using the force diagram for forces on the incline, where N is shown acting perpendicular to the track and not towards the centre. The other 2 substituted for N and obtained the correct expression without understanding the direction.

Radius of circle performed: Seven students had difficulty using the geometry of the figure to determine the radius of the circle performed by the marble. Three used the radius of the sphere rather than the sum of the radius of the sphere and the marble.

Some students forgot that the program provided help for the first question and that information might have to be modified. This difficulty was resolved when they performed a check, and the program prompted that a mistake had been made.

Next the conceptions portrayed by the students are mentioned.

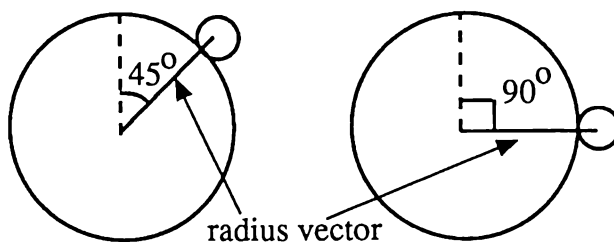


Figure 8.4: Conceptions of points at which the marble would lose contact with the sphere.

Conceptions portrayed: The only incorrect conceptions portrayed by the students were with regards to the point at which the marble would lose contact with the sphere. Three students thought that the marble would lose contact when the radius vector was at 90° (Figure 8.4). All 3 managed to resolve this difficulty when considering that the

velocity of the ball would make it lose contact before then. Five others thought that the marble would leave when the radius vector made an angle of 45° . Two realised that their prediction was wrong upon successfully solving the problem and obtaining a different answer.

Next the transcripts of the interaction records compiled by the researcher and the program are discussed.

Excerpt of program compiled transcript.	Comments to provide clarification of the interaction.
Start panel	At the start of the program the <i>Start panel</i> is selected.
Solve question check	Opens the <i>solve question</i> window.
Check force	Clicks the <i>check force</i> button. Program will prompt for selection of <i>position</i> since it has not been done.
Check energy	Clicks <i>energy check</i> button. Program will prompt for selection of <i>positions</i> 1 and 2 since these have not been selected.
Start panel	Returns to <i>Start panel</i> .
Check force equations	Opens <i>check force equation</i> window.
Normal direction	Selects the <i>normal</i> direction.
On the incline	Selects to write the force equation <i>at the incline</i> .
$F = Mg\sin\phi$	Types in this equation.
Force diagrams	Opens the <i>force diagrams</i> window.
At the start	Selects to draw forces <i>at the start</i> .
Resolve forces	Requests for <i>resolved forces</i> to be drawn.
Check force equations	Returns to the <i>check force equation</i> window.
$F = Mg\cos\phi$	Types in this equation.
$N = Mg\cos\phi$	Prompted by program to substitute for F and replaces it by N.
Solve question check	Goes to the <i>solve question</i> window.
At the start	Selects the <i>Start</i> as the position.
Check force	Clicks the <i>check force</i> button and is prompted to write the force equation for the top of the loop.
Check force equation	Returns to the <i>check force equation</i> window.
Force diagrams	Accesses the <i>force diagrams</i> window.
At the top of the loop	Selects <i>top of the loop</i> at which to draw the forces.
Resolve forces	Requests for resolved forces to be drawn.
Check force equations	Returns to the <i>check force equation</i> window.
At the top of the loop	Selects to check the equation for the top of the loop.
Normal direction	Selects the <i>normal</i> direction for the force equation.
$F = Mg$	Types in this equation. Prompted by program to start with $F = Ma$.
$F = Ma$	Types in this equation. Prompted by the program to replace F.
$Mg + N = Ma$	Prompted by the program to replace a.
$Mg + N = \frac{Mv_T^2}{R}$	Substitutes for a. Will be prompted by the program that the check is over and to proceed to the <i>Solve question</i> window if further simplification is required.

Table 8.4: An excerpt from a program compiled transcript.

8.5:4 Differences in transcripts compiled

The interaction records of the students and the transcripts recorded are compared to note the similarities and differences in information collected. An excerpt of a program compiled transcript is given in Table 8.4 as an example of the information collected by the program. The first column includes the transcript and the second column provides the actions of the student. Table 8.5 summarises the information which can be obtained by the two methods. The extent to which the two methods address each of the information collection aspects mentioned in Table 8.5 are discussed below.

Aspects of data collection	Researcher	Program
Perform observation.	Yes	No
Document verbalisations.	Yes	No
Obtain clarification.	Yes	Yes
Document features and information used.	Yes	Yes
Make or enable inferences.	Yes	Yes
Connect students written information with record compiled.	Yes	Yes
Document conceptions.	Yes	Yes
Note mistakes made.	Yes	Yes

Table 8.5: The information collection aspects addressed by each method.

Perform observation: The program could not observe student behaviour. In contrast the researcher can document when the student ponders, pauses or sits in front of the screen, attempts to make sense of the information, and switches between the program to pen and paper solution. The time spent on any one aspect of the problem solving is missed by the program, but can be recorded by the researcher.

Document verbalisations: Students commented to themselves and verbalised their difficulties to another person who might offer help. This was documented. Some excerpts of student verbalisation documented are given above, all of these were to themselves.

The program misses these verbalisations but records information accidentally used by the students (see response of M5 below). This may cause incorrect inferences. These verbalisations can be used to note aspects of the program requiring extension.

How do I carry on? (F2)
I know the force equation will be the same for rolling and sliding but I will still check to make sure. (F3)
I am just browsing through to get ideas. (F4)
Now where do I go next. (M3)
I wish the program could do this for me. I hate these algebraic things. (M1)
There is no 45° point to choose. I will choose a general point in the loop. (F1)
No this is not the window I wanted. This is the energy window, I want the force window. (M5)
Damn, why do I always forget to choose the position of the ball before checking. It would be good if the computer knew what I was doing. (M7)

Obtain clarification: The researcher could immediately seek clarification from the student but care was needed so that student interaction was not hampered. This was resolved by seeking clarification immediately upon completion of the interaction. In the case of program compiled transcripts, the clarifications could only be obtained after analysis and the student might then not recall the reasons for performing a certain action.

Document features and information used: The program is superior to the researcher when the student rapidly switches windows or types and changes information. The researcher had difficulty keeping track of the minor details.

Infer information: The inferences drawn by the researcher enrich those drawn from the program transcripts. The researcher can use observations, verbalisations and immediate clarification to correct inferences made from the program transcripts. Examples have been given in the discussion on verbalisations.

Making connections with written information: The information written by the student helps understand the interaction. It is difficult to accurately link written information with the program transcript. The researcher could note the stage of the interaction at which the information was written.

Document conceptions: Student conceptions inferred through the diagrams drawn, equations or statements written, verbalisations, discussions and information used has to be discussed further with the student for confirmation.

The program transcripts can be used to infer the conceptions (correct or alternative) of the students. For example, if the student types the force equation $N = Mg$ for the top of the loop, then the conception might be that the acceleration at the top is zero. This could be because the student has the conception that since there is no motion in the radial direction (Figure 8.5) then the acceleration must be zero. To confirm that this is not an error of omission, clarification needs to be obtained through discussion of the situation.

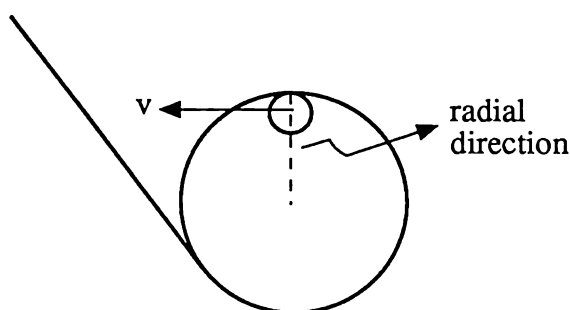


Figure 8.5: The horizontal motion of the ball at the top of the loop.

Conceptions portrayed through diagrams are more obvious and easily confirmed. Even then the situation has to be further probed through a discussion with the student. The two conceptions portrayed during the solution of the related problem (Figure 8.4) with regards to the position at which the ball would leave the sphere were documented using observations. The program would not have documented these as the choice of 45° or 90° was not available. These can be easily inferred from the written solutions of the students. Verbalisations of students provide greater opportunities to document their conceptions.

Mistakes made: The program can capture mathematical errors, generally arising due to numerical manipulations, algebraic manipulation of symbols, incorrect use of symbols to represent physical quantities, and incorrect use of relations between physical quantities. The checking mechanism of the program determines such errors. The excerpt below shows errors documented by the program when checking the students force equation for a general point in the loop.

Excerpt	Clarifying comments
$F = Ma$	Starts with this equation suggested by the program. He is then prompted to substitute for F.
$N + Mg = Ma$	Substitutes and is prompted that the component of Mg should have a cosine function.
$N + Mg\cos\phi = Ma$	Corrects the error and is prompted that the angle symbol is incorrect.
$N + Mg\cos\beta = Ma$	Replaces the angle symbol and is prompted to substitute for a.
$N + Mg\cos\beta = \frac{Mv^2}{R}$	Changes the sign by comparing the information supplied and correctly substitutes for acceleration. Is prompted that the velocity symbol is incorrect.
$N - Mg\cos\beta = \frac{Mv_T^2}{R}$	Substitutes the correct velocity symbol and completes the check. (M1)

These errors are also documented in the transcripts compiled by the researcher. Diagrammatic representational errors are not captured by the program since it has a fixed set of correct force diagrams which students can access. Observations by the researcher and the written transcripts of the students enriched this data.

Next the information analysed in this chapter is reviewed to discuss the extent to which the research aims of this study have been fulfilled.

8.6 EXTENT TO WHICH RESEARCH AIMS HAVE BEEN MET

Each of the aims are discussed in the order they are stated in Chapter 1.

What assistance do students seek in interacting with a computer-assisted system?

The data analysed shows that students will consult the program if they need information to solve a problem. The feedback obtained from the students (section 8.3:5) and the use of the program by them for the solution of the related problem suggests this conclusion. The features most used were those directly needed for the solution, writing energy equations, writing force equations and performing the solution of the problem. The simulation feature was also consulted to check the answer and to understand the motion of the ball. The features students indicated they would consult (Chart 8.1) and the features consulted by them (Chart 8.2) confirms students will use features deemed useful.

The feedback obtained suggests that students do not wish to obtain too much help. They wish for sufficient information to enable them to solve the problem or to help check the solution to the problem. The comments of the students in section 8.3:3 confirms this claim.

This research aim was adequately fulfilled since it was determined that the students would consult the program for activities which would help solve problems. Fifteen indicated that they would consult such a program if it would help them to solve tutorial questions (item 15 in Table 8.1).

To what extent does the system allow the students to control their interaction?

The program has different windows for a variety of interactions, providing ease of accessing and using information. The interactions do not have to be followed stepwise and information is available to commence the interaction at different levels. Help is only provided when requested or when some incorrect information is detected.

Only 2 students felt that the program did not sufficiently allow them to control the interactions (item 6 in Table 8.1). These students had difficulties understanding the information supplied and pursuing the hints during interaction. The responses to items 1, 2, 3, 5 and 6 in Table 8.1 indicate the program was easy to use, information was easy to obtain, access to information was easy, program was flexible to needs and it enabled sufficient control. The comments given in section 8.3:3 supports the claim that the student needs were served.

The limitations mentioned by students in section 8.3:5 suggest some inflexibility in the handling of mathematical expressions such as decimals and brackets. Apart from this the students had no other concerns. The program appears to allow sufficient flexibility for the students to control the interaction. Total control of the interaction is not possible as the responses of the program are limited by the information which has been included and the rules which govern its interaction.

Do students find it encouraging and worthwhile to consult such a computer-assisted system?

All the 14 students consulted the program (including 5 who could solve the problem without help from the program). Evidence that the students considered the program worthwhile is obvious from the positive responses indicated:

to the items in section A of the questionnaire (Table 8.1);

in stating the features of the program which they liked (section 8.3:3);
and

in mentioning the helpfulness of the program in providing helpful
information.

Students who did not manage to solve the related problem still indicated that the
program helped them.

This aim has to be considered together with the first and second aims discussed in this
section. If the program provided adequate assistance, allowed students to control the
interaction to some extent, and the students felt they benefited from the interaction,
then they will consider it worthwhile and encouraging to consult.

Which features of the system are deemed good by the students?

The features which helped to solve the problem and to provide more information for
understanding the solution were deemed good (Chart 8.1). Of the 7 purposes for
which the students stated they would use the program the envisioning feature and the
force diagrams features were the least popular choices. However, all 14 students used
the interactive force diagrams (Chart 8.2). Only 5 used the envisioning feature to solve
the related problem even though 11 indicated it was helpful in response to item 10 in
Table 8.1.

The students used the features which allowed them to perform an activity with minimal
help less than those which provided help to resolve the difficulties encountered. The
simulation window was the most popular as it enabled the students to observe the ball
successfully performing the loop, performing a projectile motion path when
unsuccessful and performing oscillations when the starting height was less than the
radius of the loop.

This research question was more than adequately answered.

What program limitations do students encounter?

The limitations encountered are detailed in section 8.3:5. No major concerns were
mentioned by the students which was expected since the students felt the program was
helpful, catered to their needs and allowed them to control the interactions.

Some minor concerns of individual students were:

The inability of the program to rearrange equations.

The program not handling decimal numbers during the force and energy equations check.

The slow processing speed of the program.

The provision of insufficient help for the simulation window.

Since the students did not encounter any major limitations the program has potential for future development and use in the classroom or laboratory.

Can students transfer their learning to the solution of related problems?

The discussion in section 8.5:2 shows that the direct recall and transfer of specific information was not high but that of general information was good. Recall upon obtaining cues through program use was encouraging. All 14 students recalled the need to write energy expressions at two points, and 12 that a force equation was needed. Nine recalled that the force equation for sliding and rolling was similar. Specific recall was not very high as only 5 recalled that:

$N = 0$ was the condition for the loss of contact;

the force equation should be written at the point where contact is lost;
and

the energy equation should be written for the start and the point of losing contact.

Transforming the knowledge from the first problem to the related problem was found difficult by all except 5 students. That only 8 students managed to solve the problem when all 14 had successfully solved the related problem shows incomplete transferability of knowledge to the related problem, even though the program provided information and clues. Table 8.3 shows that 10 students managed to successfully use the program to:

write the energy expressions at 2 points and equate them to obtain the energy expression.

write the force equation for the point at which the marble loses contact with the sphere.

determine the constraint that $N = 0$ when the marble loses contact with the sphere.

This confirms that the program provides information for the solution of the related problem. The sample was too small to conclusively determine whether the program enhanced transferability of information, which needs to be further researched.

How effective is a computer-assisted system in interactively collecting data on individual student knowledge, conceptions and problem-solving styles?

The program compiled transcripts have been detailed and compared with those compiled by the researcher (section 8.5:4). One aspect where the program is superior to the researcher is in documenting features and information used while attempting the solution. It accurately documents the stepwise interaction. The steps performed can be used to infer the problem-solving styles of students. The written transcripts of students can be used to further document the knowledge and infer the problem-solving styles.

The program compiles information from which the conceptions of students can be inferred. The information documented needs to be discussed with the students to confirm the conception.

The program can be effectively used to collect information and to investigate problem-solving styles. Conceptions of the students can be inferred, but interviews are necessary to confirm these.

8.7 CONCLUSION

The data collected using the program evaluation and problem evaluation questionnaires, the transcripts compiled by the researcher, and the program has been analysed. This has been used to evaluate the opinions of students on the program features. The features and information used by the students for solving the first and related problem have been discussed, and the conceptions of students and knowledge used to solve the problems have been described.

The research aims have been met with the proviso that the transfer of learning requires further research. A further development would be to increase the choices available to students to enable the program to capture the conceptions portrayed by students.

The next chapter summarises the findings of the study, discusses their implications, and recommends future work.

CHAPTER 9

CONCLUSION

A summary of the findings is first presented followed by a discussion of their implications. The chapter concludes with recommendations for future research.

9.1 SUMMARY OF FINDINGS

The problem-solving information shows that the solution to the problem can be commenced at several points and can be performed in a variety of ways. The knowledge needed, and that which students used to solve the problem, details the assistance which needs to be provided. The analysis shows that knowledge needs to be represented in descriptive form, as basic relations and in diagrammatic form. The conditions of applicability of the knowledge to enable its effective use have been documented.

The 14 activities (Chapter 5) effectively capture the problem-solving processes and strategies used by students. The activities can also be used to depict the differences between novices and experts, which are coherent with the findings of other researchers. Problem-solving flow charts of these activities depict the problem-solving styles, and show that information is accessed and used in an idiosyncratic manner. This indicates that the structuring of knowledge into hierarchical or other interconnected patterns of information units will not serve the problem-solving needs. A learning environment based on the generative learning theory, using a constructivist environment is implied, using transmission to supply information when needed or requested.

The knowledge normally used to help students (documented using experts and teachers) provides the minimum knowledge necessary to explain, perform the solution, and to provide an understanding of the solution. Documentation of the complete solution using the fundamental principles and quantities of physics has shown the vast amount of related information which indirectly impinges on the solution. This knowledge is not considered when the solution is normally performed since it would be tedious and uneconomical. However, the conditions of applicability and the validity of assumptions may be used to provide deeper understanding, and the starting framework for the knowledge base of the program. Although all the

knowledge has not been included the program provides availability for further reference, and possibilities for extension of the knowledge base.

Students consulted the program when they had difficulties and to check the solution. They will use the program if they perceive it to serve their needs, if they do not encounter difficulties in consulting it, and when it has features which they wish to consult. They generally preferred to use the program to check their solution, to observe simulations, to obtain an understanding, to check the answer, and to obtain hints. Students did not want the total solution to the problem, preferring the program to provide sufficient help to enable them to perform the next step or to make suggestions. The students used the program for consultation to obtain information and to solve the problem but preferred pen and paper to document it.

Use of the program to obtain help in solving the related problem shows that it is helpful in providing information and cues for recall. This indicates that the program has sufficient general features to help students with problems of a similar nature. Students were not very effective in transforming knowledge from the first problem to the second. The data collected on the transfer of learning was insufficient to provide conclusive evidence that this was enhanced by the program.

The information collected by the program is effective in documenting the information and features used by the students, but is less effective in compiling conceptions of students. It does however collect information from which conceptions can be inferred, and confirmed through discussions with the students.

It can be concluded that the program was successful in providing helpful interaction to the students, and that students will consult such a program when the problem is sufficiently difficult to warrant consultation.

Next some implications of the findings of this research for learning are discussed.

9.2 IMPLICATIONS FOR LEARNING

Role of computer: The computer should not be used as a stand alone substitute for teaching and learning but as a supportive and complementary tool. It can be used to clarify queries, provide a platform to obtain further information, to help students realise aspects they do not comprehend, to document student interactions, and to obtain insight into the problem-solving behaviour of students. All of these can be used to interact with each student's difficulties. Similar programs could be developed for

other problems, and students could spend time working with these programs depending on the amount of help they require.

Program features: Students will use a program if they feel it is helpful, easy to use, and serves their needs. Since students only wish the right help when consulting the program, information represented in the program should be split into small units. It should not be based on a hierarchical organisation of knowledge due to the idiosyncratic manner in which students perform the problem-solving activities. The program has to perform a variety of functions, allow access of information in a variety of ways, and be based on a learning environment within which students feel comfortable, which appreciate their ideas, and is aware of their conceptions so that they can be appropriately helped.

Program development: If similar programs are developed they should be for problems of sufficient difficulty to merit student consultation. The development of the knowledge base requires information not only of the knowledge needed to solve the problem but also the knowledge used by students, the mistakes made by them and their conceptions.

Role of the teacher: With the aid of the computer, learning can be the responsibility of the student since the interaction is student initiated and not teacher centred. The teacher can serve an advisory role of guidance, providing hints on how to pursue the interaction only when appropriate.

Prior knowledge of the students has to be used to determine the knowledge to be included in the program. The vast related knowledge which impinges on the problem needs to be known by the teacher who has to decide how the student should be provided with this related knowledge when the student is in difficulty.

The teacher will have to monitor use of the program so that it is equitably used by students in proportion to their needs. The teacher can use the transcript compiled by the program during the interaction to provide clues to the help required.

Finally some recommendations for future work are made on issues which have arisen out of this study.

9.3 RECOMMENDATIONS FOR FUTURE WORK

The following are suggestions for research and development:

1. Include further problems into the program.
2. Consider how much related knowledge could be incorporated, how this would be accessed by students, and how this would hamper the problem solving interaction.
3. Study with a larger group of students, over more problems, the transferability of information.
4. Introduce more information and choices to explore the level to which the program will capture alternative conceptions.
5. Explore whether a program which includes alternative conceptions will help students to change them.
6. Investigate the use of the program to evaluate student understanding.
7. Investigate further the value for developing problem-solving skills by requiring students to solve sets of related problems and describe similarities and differences between them.
8. Investigate interaction with the program by groups of two students rather than by individuals.

APPENDIX A

INSTRUCTIONS AND QUESTIONNAIRES.

APPENDIX A1

INSTRUCTIONS TO PARTICIPANTS FOR THE THINK- ALOUD SESSIONS.

1. Your voluntary participation in the data collection will be kept strictly confidential.
2. The information is sought to assist my D. Phil. research into a computer-assisted learning program.
3. The aim of this exercise is to collect data on how the problem may be solved; obtain information needed to solve the problem; and capture your train of thought as you attempt to solve the problem.
4. I would like to remind you that this exercise is not a test of your ability to solve the problem. I am interested in how you go about solving the problem.
5. You may take as long as you wish to solve the problem. No time constraint will be applied.
6. Feel free to refer to the text provided at any point during the problem solving session.
7. The tape recorder will be on throughout the session.
8. It is important that you think aloud and verbalise your every thought and action throughout the session. Make obvious every thought and action so that your train of thought can be documented.
9. If you are unsure of anything at any time during the session, please feel free to seek clarification from me.
10. I will give you a polite reminder if you stop verbalising or if your voice seems to fade.
11. I would like you to try a warm up question before commencing on the actual task. This will help you to get used to the idea of thinking aloud and verbalising your every thought and action.

At the end of the session:

1. Please do not discuss the questions you have done with any other person.
2. Thank you for your assistance and help. Your contribution is greatly appreciated.

APPENDIX A2

INSTRUCTIONS TO PARTICIPANTS FOR THE PEN AND PAPER SESSION.

1. Your voluntary participation in the data collection will be kept strictly confidential.
2. The information is sought to assist my D. Phil. research into a computer-assisted learning program.
3. The aim of this exercise is to collect data on how the problem may be solved and the information needed to solve the problem.
4. This exercise is not a test of your ability to solve the problem. I am interested in how you go about:
 - solving the problem;
 - the examples you use or refer to solve the problem;
 - the steps you take to obtain the information for solving the problem; and
 - the steps leading to the solution.
5. Please detail difficulties encountered while attempting the problem.
6. If you managed to overcome the difficulty, explain what helped you to do so and how you overcame it.
7. If the difficulty hindered your progress in the problem solution, elaborate as to why you were unable to resolve it.
8. Please complete the task within a week. Feel free to refer to your text, lecture notes or any other information which will help you to solve the problem. You may attempt the problem as many times as you like.
9. You must perform the solution by yourself. Please do not seek help from any other person.
10. Please do not discuss the question you have done with any other person.
11. Thank you for participating in the data collection. Your contribution is greatly appreciated.

APPENDIX A3

INSTRUCTIONS TO TEACHERS

Assume that your students have done the topic(s) required to solve the attached problem on rotational dynamics. You have assigned this problem for homework and a majority of them are unable to solve it. They have requested you to perform the solution on the board during one of their Physics lessons. I would like you to write a solution taking the following points into consideration.

1. Include all the information and steps which you will present to the students so that they will comprehend the solution presented.
2. Include all the verbalisations you would use in explaining the solution steps while performing the solution.

The length of the solution presented is irrelevant to the exercise. Feel free to include more than one solution to the problem.

At the end of the solution please include all the questions, clarifications and explanations you anticipate students will request as you perform the solution on the board and how you would perform the clarification.

APPENDIX A4

INSTRUCTIONS TO PARTICIPANTS PRIOR TO PILOT USE OF PROGRAM.

1. Your voluntary participation in the data collection will be kept strictly confidential.
2. The information is sought to assist my D. Phil. research to test the workability of the computer-assisted learning program.
3. The aim of the exercise is to determine if there are any bugs in the program and the aspects of it which need improvement.
4. I would like you to attempt to use all the features of the program and to note on the sheet of paper provided features of the program which:
 - hindered your consultation in any way.
 - you feel should be modified or removed.
 - are missing and you feel would enhance the program.
 - helped and enhanced ease of consultation.
5. This exercise is not a test of your ability at solving the problem or using the program. The aim is to obtain feedback on the program.
6. You may take as long as you wish and use the program as many times as you wish.
7. If you are unsure of anything, please feel free to seek clarification from me.
8. Please do not discuss your session with others as they might be requested to use the program as well.
9. Thank you for your kind assistance and help.

APPENDIX A5

INSTRUCTIONS TO PARTICIPANTS PRIOR TO USING THE PROGRAM FOR SOLVING THE PROBLEM

1. Your voluntary participation in the data collection will be kept strictly confidential.
2. The information is sought to assist my D. Phil. research for improving and evaluating a computer-assisted learning program.
3. This exercise is not a test of your ability to solve the problem by using the program. The aim is to collect data on how:
 - the program assists you to solve the problem; and
 - you interact with the program for assistance in obtaining information and to help solve the problem.
4. You may take as long as you wish to solve the problem. No time constraint will be applied.
5. If you are unsure of anything at any time during the session, please feel free to seek clarification from me.
6. Please note on the sheet of paper provided:
 - features of the program which hindered your consultation in any way.
 - actions you wanted or expected the program to perform but it could not.
 - features of the program which helped and enhanced ease of consultation.
7. I would like you to first attempt solving the problem on your own before going on to consult the program. Once you are ready to do so, I will explain the functioning and usage of the program to you.
8. You may browse through the program until you feel comfortable to use it to solve the problem.
9. Please do not discuss your session with others as they might be requested to use the program as well.
10. Thank you for your kind assistance and help.

APPENDIX A6

PROGRAM EVALUATION QUESTIONNAIRE

Program Evaluation Questionnaire

This questionnaire is being administered to obtain feedback to evaluate the problem solving program you have just used. If you are unsure of the information requested by any of the questions, please ask me. The questionnaire is divided into two sections.

Thank you for your kind assistance and help.

Section A

Below are 15 statements about the program. Please indicate your response by circling the appropriate number on the scale beside each statement. Please answer **ALL** the questions.

Statement	Scale				
	Agree strongly				Disagree strongly
1. The program was easy to use.	5	4	3	2	1
2. I was able to get all the information required.	5	4	3	2	1
3. It was easy to access the information needed.	5	4	3	2	1
4. The program helped me to learn how to solve the problem.	5	4	3	2	1
5. The program was flexible to my needs.	5	4	3	2	1
6. I felt in control while interacting with the program.	5	4	3	2	1
7. The hints given by the program were helpful.	5	4	3	2	1
8. The simulations provided were helpful.	5	4	3	2	1
9. The diagrams provided were helpful.	5	4	3	2	1
10. The envisioning provided was helpful.	5	4	3	2	1
11. The answer check was helpful.	5	4	3	2	1
12. The solution provided by the program was helpful.	5	4	3	2	1
13. The program helped me to better understand rotational dynamics.	5	4	3	2	1
14. The program provided too much help.	5	4	3	2	1
15. I would consult the program for other tutorial questions if it was possible.	5	4	3	2	1

/Section B follows

Section B

I would be grateful for an answer to all the questions, with as much detailed explanation of your answers as you can provide. The spaces provided do not indicate the length of the answer expected.

1. List any difficulties you encountered while using the program.

2. What aspects and features of the program did you like ?

/Section B cont.

3. Assume there exists a program which you may consult for all your tutorial problems assigned. For which of the purposes mentioned below would you consult the program? You may make more than one choice.

- A To check the equations written.
- B To check the solution to a problem.
- C To consult the envisioning.
- D To perform the simulation.
- E To consult the force diagrams.
- F To check the answer.
- G To obtain hints to solve the problem.

Other purposes:

4. What are the program limitations you encountered during your interaction ?

/Section B cont.

5. Please supply any other information pertaining to the program but has not been covered by the questions.

APPENDIX A7

INSTRUCTIONS TO PARTICIPANTS SOLVING THE RELATED PROBLEM.

1. Your voluntary participation in the data collection will be kept strictly confidential.
2. The information is sought to assist my D. Phil. research to test the workability of the computer-assisted learning program.
3. I would like to remind you that this exercise is not a test of your ability to solve the problem. I am interested in how you go about solving the problem.
4. You may take as long as you wish to solve the problem. No time constraint will be applied.
5. You may refer to the program at any point during the problem-solving session. I wish to remind you that the program does not solve this problem.
6. If you wish to consult the program and are not sure how to access information for reference, please ask me.
7. Please detail difficulties encountered while attempting the problem.
8. If you managed to overcome the difficulty, explain what helped you to do so and how you overcame it.
9. If the difficulty hindered your progress in the problem solution, elaborate as to why you were unable to resolve it.
10. Please do not discuss your session with others as they might be requested to use the program as well.
11. Thank you for your kind assistance and help.

APPENDIX A8

PROBLEM EVALUATION QUESTIONNAIRE

This questionnaire has been developed to evaluate your attempt to solve the problem. If you are unsure about any of the questions, please ask me. I would be grateful for an answer to **all** the questions, with as much detailed explanation of your answers as you can provide. The spaces do not indicate the length of the answer expected.

1. Did you find it easy to solve this problem ? Please explain.

2. Did your session with the program help you in solving this problem ? Please explain.

/cont.

3. What similarities did you find between the problem you have just attempted to solve and the loop-the-loop problem you tried to solve using the program ?

(a) When you read the problem:

(b) While solving the problem:

(c) After solving the problem:

APPENDIX B

THINK-ALOUD TRANSCRIPTS AND PROBLEM-SOLVING FLOW CHARTS.

APPENDIX B1

Think-aloud transcripts: Included are the transcripts of an expert solution, and a successful, a partially correct and two unsuccessful novice solutions for problem 6. Each protocol is presented as two columns. The first column gives the verbal protocol and the second column gives the working written by the participants as they performed the think-aloud problem solution. The researcher's verbalisations and comments are included within rectangular brackets. Non verbal actions performed by the participant are placed within parenthesis. Diagrams drawn by the participants to aid their solution attempt are given at the end of each transcript.

APPENDIX B2

Problem-solving flow charts: Two novice (N1 and N8) frequency and sequential flow charts are given. The numbers in the frequency flow charts represent the number of times an activity was accessed from another, and those in the sequential flow charts represent the sequence in which the activities were performed starting with 1.

APPENDIX B1.1

TRANSCRIPT OF EXPERT E2

A uniform ball of radius r rolls without slipping along the loop the loop track shown in the figure. It starts at rest at height h above the bottom of the loop.

First of all, the height. The first impression I get is you use the conservation of energy.

You are given the initial height above the bottom of the loop. So energy argument is obviously going to be in here somewhere.

If the ball is not to leave the track at the top of the loop , what is the least value h can have in terms of the radius R of the loop ? What would h have been if the ball were to slide along a frictionless track instead of rolling ?

Rolls without slipping, so its initial energy is going to be just mass, m g times h . Its at rest.

[please keep verbalising]

To not leave the track at the top of the loop, then at the part where it is at the top of the loop the centripetal force, perhaps force due to acceleration because its circular motion around the loop, I have to exactly balance at least exactly balance the force of gravity downwards.

Now rolls without slipping. So its also rolling so we going to have energy connected with its rotational motion. So let's see.

I guess, the particular point on that slope as it comes down is going to have a certain amount of potential energy until it reaches the bottom.

It will have kinetic energy too due to its linear motion and it will have kinetic energy tied up in its rotational motion. So, the energy at a point is going to be the sum of those three quantities. So if I write that down.

Initial energy = mgh

A half $m v^2$ plus a half $I \omega^2$ squared where I is the moment of inertia and ω is the angular frequency at which it is rotating plus its potential energy which is, $m g$ times a quantity x , where x can be its position above the bottom of the loop. So the sum of those three things will have to equal the initial potential energy. So that's going to equal $m g h$. As it slides down this track.

Now that's it. Now, consider that, when its energy when it is at the top of the loop, at that point you have to find out what its energy is.

But, I can't avoid it. At that point on top of the loop, its got to be moving with a certain velocity and the centripetal force should be $m v^2$ squared over radius R . So its in circular motion going round this loop. So that's the force due to its circular motion round the loop. We probably want to let that equal m times g . Will want to be greater than or equal to that value.

So we are asked for least value up here so we look for the minimum point. Least value of h . We want to know the point at which the centripetal force just balances the gravitational force at the top of the loop.

So that tells you what its velocity at the top of the loop must be, this is square root of g over R

So let's see square root over R .

[please keep verbalising]

I am just trying to keep track of what I am trying to do here.

To get the least value of h in terms of the radius R of the loop.

Now this r , we can obtain the rotational energy term half $I \omega^2$ squared.

It starts, from zero. ω starts from zero, just trying to think how this comes into it

So the ball is rolling at all times. It starts off from rest. Its rolling at all times.

$$\frac{1}{2} m v^2 + \frac{1}{2} I \omega^2 + m g x = m g h$$

At the top of the loop

$$\frac{m v^2}{R} = m g$$

$$v = \sqrt{\frac{g}{R}}$$

.....equation (i)

$$\omega = \frac{v}{r}$$

So the radius of the little ball is small r . So should be able to relate ω to the velocity of the ball. From that I guess ω will be v over r I think.

[please keep verbalising, you are fading off]

How to establish the ratio relationship between the angular frequency at which this ball is rotating and the velocity of the ball.

The linear velocity of the ball, just trying to think of that to see if its going to be. What's the angular velocity simply, the linear velocity divided by the radius of the ball.

$$\omega = \frac{v}{r}$$

Rolls without slipping, let's see, how do we do this.

Just trying to picture this not sure that, establish the relationship between the linear and the angular velocities of the ball as it rolls along

So, this is probably obvious but I am having trouble.

Working this for myself, the angular velocity at which this ball is spinning is just the linear velocity divided by the radius.

I'm not, I am just trying to establish for my own sake whether it is that simple relationship between the angular velocity ω is just, is the velocity just r times ω where r is the radius. When you are looking at circular motion.

Just trying to establish if it is the same case for when the ball is rolling along. I am sure I have done this before so should be able to just.

I'm just having a bit of a block I think, trying to. I've got the feeling that it should be obvious but I just can't see at the moment.

r times ω for this ball, little r the radius of the small ball. I know I have done it before I just er am trying to think if it is obvious without having to prove it to myself.

I would like to convince myself that it is the case and I am quite sure that it is.

Let's see, but I am not quite sure how I, so where is the distance?

Drew figure 1

d, v t. It's got distance from there to there and corresponds to the circle where it tells us we're rotating around. That angle equals theta and that distance so distance, so theta is distance d divided by radius r so now d theta by dt is just v over r

$$d = vt$$

$$\theta = \frac{d}{r}$$

$$\frac{d\theta}{dt} = \frac{v}{r}$$

Simple but took me a while. It is just something which I have convinced myself of first before I wanted to go on further. I was pretty sure it was the case but I just wanted to make sure. So that relates omega to the linear velocity of the ball v divided by r so the total kinetic energy or the total energy can now be written entirely in terms of its height and its linear velocity at any particular point.

$$\omega = \frac{v}{r}$$

So we going to have a half m v squared plus half I, moments of inertia we'll have to work out next, v squared over little r squared plus m g times x which is m g times the original height and the moment of inertia of the sphere.

$$\frac{1}{2}mv^2 + \frac{1}{2}I\left(\frac{v^2}{r^2}\right) + mgx =$$

$$mgh$$

Can I ask you that or do I look it up.

[you are on your own]

Checks book. Ok, two fifths m r squared and its little r and put that into this equation half m v squared plus a fifth m v squared. Let's see m g x is equal to m g times h

$$I = \frac{2}{5}mr^2$$

$$\frac{1}{2}mv^2 + \frac{1}{5}mv^2 + mgx =$$

$$mgh$$

Now that's the general equation for that energy at a particular position, this particular height and the place at which we are interested in is at the top of this loop

So it is going to be at the point where the quantity x which is just its height at a particular point at a particular instance. x is just going to be 2 times the large radius of the loop.

$$x = 2R$$

So what is going to be the height of that point. Now the point for which we worked out before. The centripetal force at the top of the loop has to be just balanced by the gravitational force if the thing is going to stay at the top of the loop.

So that tells us what v is going to be, that is the square root of g over R . So at the top of the loop x is two R and the velocity should be square root of g over R . That's for the ball to stay on, stay in contact with the loop.

So should just be a matter of plugging the rest of the things into that general equation to work out what the height should be in terms of capital R the radius of the loop.

So a half $m v$ squared plus a fifth $m v$ squared plus $m g$ times its height which is two times R . It is equal to its initial at the top before it is let go. So we can solve that for height h .

So we've got something is not quite right. The dimensions are not right here at the moment. Got one over R in these two terms here, the last one square root. Let's see v equals, I have made a mistake at the top. The velocity should be the square root g times R .

Just tracing my steps back, to where the mistake was obvious. The dimensions were not correct so it is not actually one over R .

So that should be ok now. So we just have to add a few things here. That should make a fifth, so we've got $m g R$ times a half, that's a fifth plus two so that is $m g h$

The $m g$ cancels out, so a half plus a fifth plus two times R which is equal to twenty seven over ten times R .

So I'll have a quick check to make sure I haven't made any silly mistakes. Well I think that's about right. So according to that calculation that's what h should be. Now the second part, what would h have been if the ball were to slide along a frictionless track instead of rolling ?

In that case you just neglect the part of the energy associated with the spinning ball that is the term half $I \omega$ squared. So that just wouldn't come into it.

$$v = \sqrt{\frac{g}{R}}$$

.....equation (ii)

$$\frac{1}{2} m \frac{g}{R} + \frac{1}{5} m \frac{g}{R} + 2mgR = mgh$$

[Changes equation (i) to]

$$v = \sqrt{gR}$$

[Changes equation (ii) to]

$$v = \sqrt{gR}$$

$$\begin{aligned} \frac{1}{2} mgR + \frac{1}{5} mgR + 2mgR &= mgh \\ h &= \left(\frac{1}{2} + \frac{1}{5} + 2 \right) R \\ &= \frac{5 + 2 + 10}{10} R \\ &= \frac{27}{10} R \end{aligned}$$

So following that through we just take out this value.
Just take out one of the terms in this energy equation.
The half I omega squared term.

Following that through, think that you would find that,
let's see you just relate this quantity over fifth so if
that's the ball sliding then I suspect the height will just
be a half plus two times R. Which, so five over two
times R

So I think that's all there is to it. If it is sliding then its
just a particle. Then can think just in terms of its linear
velocity at all points and should just be a simple case of
dropping out the energy term associated with its
rotational motion and all the previous work we should
just carry through. Just omitting that term and that
should be the case.

Just thinking whether that, no doesn't make a great
difference to the final result twenty sevenths tenth to
twenty five tenths times R. Its not a big difference.

Just wondering if that is, makes sense. I guess, most of
its energy is contained in its linear motion and so and so
its potential energy, so rotational motion doesn't have
too big an effect on the final result. Well, that's it.

Sliding ball
$$h = \left(\frac{1}{2} + 2\right)R$$
$$= \frac{5}{2}R$$

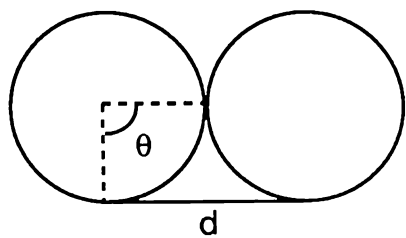


Figure 1

APPENDIX B1.2

TRANSCRIPT OF A NOVICE SOLUTION (N3) WITH THE CORRECT ANSWER BUT ERRORS IN WORKING WHICH CANCELLED.

It starts at rest at height h above the bottom of the loop
If the ball is not to leave the track at the top of the loop
what is the least value h can have in terms of the radius
 R of the loop
What would h have been if the ball were to slide along a
frictionless track instead of rolling.
So we've got to include the effects of friction and use
conservation of energy.
Draw a diagram and label the parameters that we know
The small ball is of radius little r . It starts off at height.
The loop has got radius big R
That then is about it.. Let's go now then. Its a uniform
ball. It's not mean its uniform density. Uniform.
Should be. Still not specified correctly uniform ball ,
something like that? Uniform density. Ok its uniform
density then and conservation of energy tells us, means
that
What do we need? We need to know the height of the
top of the loop that would give the potential at the top of
the loop.
That would give us the energy at the top of the loop.
We need to know the velocity of the ball at that point
and we need to know its rate of rotation so we can work
out the frictional, rotational energy
So the loss of potential energy for the ball to arrive at
the top of the loop is h minus two R times mg . mgh
minus mg times two R . So that's the potential energy
lost and that must equal the kinetic energy gained plus
the rotational energy

Draws Figure 1.

Conservation of energy

Loss of PE for the ball to
arrive at the top of the loop
is
 $(h - 2R)mg$

I write everything out. The kinetic energy gained plus the rotational energy. That's an expression for the kinetic energy gained.

The kinetic energy here does not, at the top of the loop is not zero. It must pass around it and, and the ball is not to leave the track at the top of the loop.

So therefore at the top of the loop, for the critical condition. Well something have to do, critical condition

The least value h can have so that's the critical condition. $m v^2$ over R must equal mg

The force upwards must equal the force downwards, point the gravity downwards, acceleration upwards

That tells us that v^2 is equal to $g R$ and therefore half $m v^2$ is equal to $m g R$ over two. So the kinetic energy gained at the top of the loop.

What's the rotational energy of the ball being the same?

Rotational energy, rotational energy, now I've got it.

Rotational energy. The ball was radius, velocity v will equal, now what's going on here?

Its travelling at, let's see now the ball's travelling at a velocity v . Therefore the thing is spinning at a velocity v omega r v equals omega r , and the energy is equal to I omega squared equals $I v^2$ over R

To calculate the rotational energy we need to know what the moments of inertia of the ball is and of course it is two fifths m . Do you know its R or R^2 ?

[You are on your own. You may refer to the text]

(Checks book) Ok so its two fifths $m R^2 v^2$ over R equals two fifths $m v^2$

Now argue this one out for me. Do I proceed in faith?

Is that correct?

[you are on your own]

Oh! you are not going to tell me. Alright well we'll see if its right. See what we get in the silly answer.

Rotational energy is $I \omega^2$. Omega equals v the velocity, that's got to be right.

Must equal the Kinetic energy gained + the rotational energy.

At the top of the loop

$$\frac{mv^2}{R} = mg$$

$$v^2 = gR$$

$$\frac{1}{2} mv^2 = \frac{mgR}{2}$$

$$v = \omega r$$

$$\text{Rotational energy} = I\omega^2 = I v^2 \dots\dots\dots \text{equation (i)}$$

$$= \frac{2}{5} m R^2 v^2$$

$$\dots\dots\dots \text{equation (ii)}$$

So that's the rotational energy. So we bring those two things together. They are equal to evidently. h minus two R brackets $m g$ must equal $m g R$ over two plus two fifths m, v squared R .

Now what was v squared? v squared was $g R$ equals $m g R$ over two plus two fifths $m g R$ squared. Interesting that's not dimensionally correct

Can't have it both wrong. v squared equals $g R$ so we've got $g R$ squared

[Please keep verbalising]

Yes, yes. What I've got is not dimensionally correct $m g R, m g R$ squared and not $m g R$. So I've got a factor of R wrong somewhere. Let me see now. So therefore h equals $m g R$ two plus two fifths $m g R$ takeaway two $m g R$. That's plus two $m g R$. h is equal to two fifths plus two is equal to seven half is

This is five versus seven tenths, so this is equal to two point seven $m g R$. Right, that's good enough for me. Something about this R is obviously wrong. That's over R squared. That's over R squared. That is v . So we just, that's two $m v$ squared, two fifths

Why haven't I got an R now there? Same thing I've got an R here that's right. v squared over R squared is I . Teta is $m R$ squared. So therefore the rotational energy is two fifths $m v$ squared. Yes, that's fine. That's dimensionally correct. Two fifths $m v$ squared but v squared is $g R$.

That's all. Right job done. Two point seven $m g R$. That's the answer.

Would you really be happy if I checked.

[It is entirely up to you]

Does it mean it is wrong?

[That is not what I mean. You are on your own.]

You can check everything so very often. Its so straight forward sometimes. What are we going to do if we check it. All we got to do is that part anyway.

$$(h - 2R)mg = \frac{mgR}{2} + \frac{2}{5}mR^2v^2$$

.....equation (iii)

$$v^2 = gR = \frac{mgR}{2} + \frac{2}{5}mR^2$$

.....equation (iv)

$$mgh = \frac{mgR}{2} + \frac{2}{5}mR^2 + 2mgR$$

$$= 2.7mgR$$

Divided eq (i) by R^2

Divided eq (ii) by R^2

Cancelled R^2 in eq(iii)

Changed R^2 in eq(iv) to R

$$h \geq 2.7mgR$$

.....equation

(v)

Let's do the second part where it says no friction. Let's do that first and see how it compares. I suspect 0.7 is due to friction so we'll just get two m g R.

Without friction, let's see now. Let's do it in a different way and redefine some variables.

Let's call this distance from the top of the loop to where we're going to release the ball from h dash. Then m g h dash by conservation of energy.

Thought that there is friction in the loop this least value h here is the lower value so h must greater than or equal to 2.7 m g R. So it must be equal to 2.7 m g R.

Assuming that m g R is correct so it is great enough. So m g h dash is equal to whatever it would be half m v squared. But we've seen that v squared is equal to g over R. That's a half m g R is v squared equals a half m v squared equals m g and m g equals m v squared over R.

Ok, weight downwards, force of acceleration upwards so therefore v squared equal to g R. Let's check that one.

So therefore h dash is equal to R over two

Interesting. Why have we got this m g here. Ah, of course I didn't cancel that one

2.7 m g cancels out from the previous expression. So that's useful as h dash is R over two. Its dimensionally correct in the first place. Right, all this seems rather vague. See whether we can twist this round a bit. Got a minus sign in the wrong place.

h is equal to, this is h dash. So h is equal to, h is equal to 2.5 R where h is the height.

That fits in nicely as we have got h equal 2.7 with the rotational energy, 2.5 without I am happy with that. I consider this question finished.

Draws figure 2.

Conservation of energy

$$\begin{aligned} mgh' &= \frac{1}{2} mv^2 \\ &= \frac{1}{2} mgR \end{aligned}$$

$$h' = \frac{R}{2}$$

Cancels mg from eq (v)

$$h = 2.5R$$

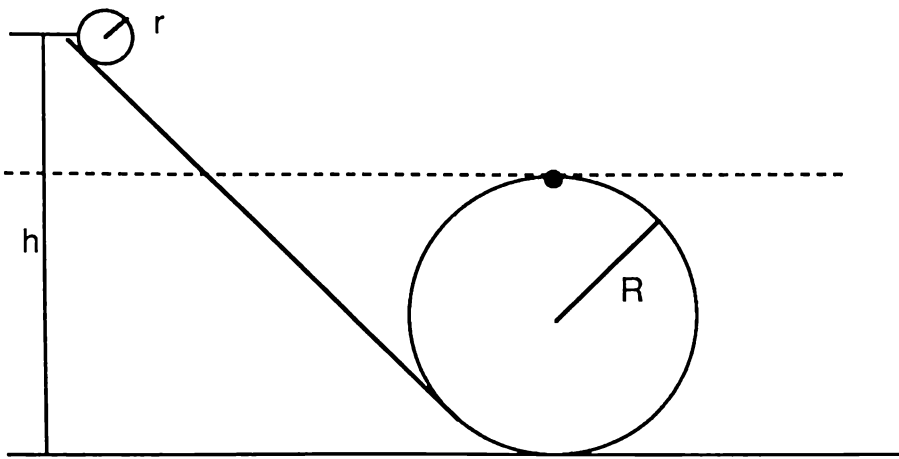


Figure 1

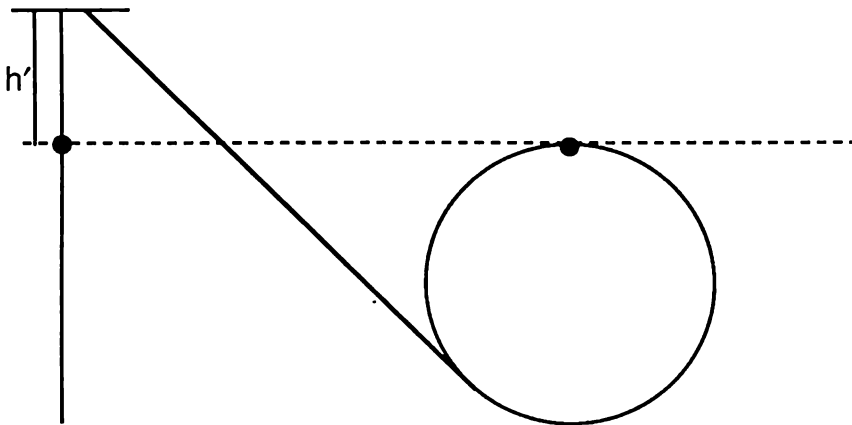


Figure 2

APPENDIX B1.3

TRANSCRIPT OF A PARTIALLY CORRECT NOVICE SOLUTION (N7).

Uniform, radius, rolls without slipping. Well look for something that can be useful in the area of energy
Rolls without slipping,, loop the loop. At height h
okay so we got energy bottom of the loop the ball
Is not to leave the track at the top of the loop the loop.
Height it can have in terms of radius of loop. Least value h can have.

Okay its definitely energy, kind of tricky

[please keep verbalising]

Look for the conditions, and finally it will be the, it will have condition that ball is not to leave the track at the top of the loop.

Okay its tricky, it v is more than, one energy that's okay.

Okay, going around its going to get a force. That will be equivalent to the centripetal force which is gives me.

Can't remember that

[you may refer to the book]

I should be able to remember that v^2 over a , m
 v^2 squared over the distance of force. So m , I hate this question.

Let's just formulate a bit. Okay using the initial conditions. Rotation potential is known but towards the, not sure what conditions to be looking for.

It won't drop off it hardly, hasn't got zero energy its got what's it got. I don't know that there. Of course I forgot that zero energy, then it wouldn't even make it up to the top of the loop it would just fall off or something like that. So, I'm not sure.

Well one thing I can do, its changing gravitational potential and its $m g$ times the change in distance and that's h minus two R and its the energy.

$$mg(h - 2R) = \Delta E$$

Where's, where's that, never seen before. That's the thing okay it makes angular arguments. I don't understand this final condition

If it is not to leave the track at the top of the loop. I'm not sure how to formulate that at all

No, no idea to look it up. Using a force idea, $m v$ squared upon r . I might already have that there. I hate this question

Okay the minimum condition to be that when it does reach the top then it can start to leave or something also, go down there.

Okay, so go back to energy, see if anything to do with energy. So the minimum energy the ball can have at the top to get there

[please keep verbalising]

Seems to have gone blank really. Seems to have, bad attitude towards this problem.

This is the same problem. Okay the forces here. When reaches the top angular speed got to be greater than $m g$. Well take the minimum case as equal to $m g$. See if going to help me at all, this er doesn't tell me too much (checking book)

Guess in that case v equals square root of $g R$ and okay that can give us energy condition. So minimum energy is. Should look at angular rather than linear, its tricky. Total kinetic energy equals half $I \omega$ squared plus half $m v$ squared and ω equals v upon r where v is linear.

Okay, so I'll just plug in everything and see how its going. Okay v , that's root $g R$ upon r . Half $m v$ squared, so that's $g R$. That's minimum energy condition I hope.

So, just equating ideas. There's a change in potential, has to equal minimum, get rid of that square. Kinetic, that's just $g R$ upon r squared.

Okay what am I actually looking for? what's the least value h can have. Okay, all seems to be completely wrong but its all I can think of so I'll just solve for h

$$\frac{mv^2}{R} = mg$$

$$v = \sqrt{gR}$$

$$E_k = \frac{1}{2} I \omega^2 + \frac{1}{2} mv^2$$

$$\omega = \frac{v}{r}$$

$$= \frac{1}{2} I \left(\frac{\sqrt{gR}}{r} \right)^2 + \frac{1}{2} mgR$$

$$mg(h - 2R) = \frac{1}{2} I \frac{gR}{r^2} + \frac{1}{2} mgR$$

I don't mind equating the equations its relatively easy.
Is it wrong?

These seem to cancel not very much, doesn't seem very hopeful. Its rather messy sort of equation. Try and look for things that are simpler makes it seem more sensible, okay

[please keep verbalising]

Add two R to this side this has got h into all the other variables, still very messy

[please keep verbalising]

Okay adding two R to inside. I'm putting into the bracket. Taking out the half, and this one becomes here plus four R. inside the bracket. Its very messy but what can you take out now. Doesn't really look very good. I upon m r squared plus m plus four

Okay, what does that tell me? I don't like it at all, m r squared. Nothing in particular.

I hope my logic is right but doesn't continue I think I'm on the wrong wavelength because I don't see how I can simplify that to something sensible.

Can't see anything wrong, its terrible but can't be helped I'll rewrite, do something like this and I'm confident of my logic and everything but and you know just sort of give up this idea and that must be it. I have to start again or something., I definitely have to start again. Well some of it seems to make sense to me the way I've worked that out but I don't like my answer that's all.

In terms of radius r. All these other things are constants which helps but I suppose that makes sense.

Well, what would h have been if the ball just slides along a frictionless track instead of rolling

Okay it means there's just linear kinetic energy. So can take out that factor of half I omega squared which would. Does it mean we have to equate it. That's what it means doesn't it

$$h - 2R = \frac{1}{2} \left(\frac{IR}{mr^2} + mR \right)$$

$$h = \frac{1}{2} \left(\frac{IR}{mr^2} + mR + 4R \right)$$

$$= \frac{1}{2} R \left(\frac{I}{mr^2} + m + 4 \right)$$

Okay, going back. This is exactly the same as before and taking out that rotational factor. Must equate that, simplifying energy h equals half R plus, well that's something seems to make sense.

Two and a half R . That's nice and simple. Is that enough?

I don't know why the first bit didn't sort of fall into place like that? Although I don't usually like the feeling but its quite silly just looking for this thing to come out nice and everything cause its hardly real life

I reckon this problem is a fake so, that's okay I think. I think the first part is not what they are after but

It seems messy although it does makes sense because everything is in terms of r and all that.

$$(h - 2R) = \frac{1}{2} mgR$$

$$h = \frac{1}{2} R + 2R$$

$$= 2\frac{1}{2} R$$

APPENDIX B1.4

TRANSCRIPT OF AN UNSUCCESSFUL NOVICE SOLUTION (N8).

A uniform ball of radius r rolls without slipping along the loop the loop track shown in the figure. It starts from rest at height h above the bottom of the loop. If the ball is not to leave the track at the top of the loop, what is the least value h can have in terms of the radius R of the loop. What would h have been if the ball were to slide along a frictionless track instead of rolling ?

Okay, at any point its total energy is half $I\omega^2$ plus half mv^2 which has components plus mgh wherever. h is still h which is always equal to its initial energy mgh .

So when it gets to the top of the loop. Make it so its better, its $2R$, h equals $2R$

Okay so mgh equals half $I\omega^2$ plus half mv^2 plus $mg(2R)$. That equal, ball is not to leave the track, what is the least value h can have.

The energy pulling it down $mg(2R)$ is the force that's acting down there. Half mv^2 is that way half $I\omega^2$ is what's rolling there.

So h equals half $I\omega^2$ plus half mv^2 plus $mg(2R)$ over mg

I don't like this mg is the force going down the force keeping it going that way is the kinetic energy divided by its mass

I think, I don't know I think I'm stuck.

I'm thinking of how far the $I\omega^2$ is different to the mv^2 because it hasn't got any, it has got a unit because I for a uniform ball

(checking book). There's a table in here. It should be in its index got a table here, moments of inertia, two seven six. There you go I equals two fifths mr^2 squared times two thirds mr^2 squared

$$\frac{1}{2} I\omega^2 + \frac{1}{2} mv^2 + mgh' = mgh$$

$$h' = 2R$$

$$mgh = \frac{1}{2} I\omega^2 + \frac{1}{2} mv^2 + mg(2R)$$

Drew figure 1.

$$h = \frac{\frac{1}{2} I\omega^2 + \frac{1}{2} mv^2 + mg(2R)}{mg}$$

A uniform ball so its two fifths omega squared will be v squared over r squared so they cancel plus half m v squared plus m g two R all divided by m g equal.

We got now v squared times no we've got half m v squared and half m v squared which is m v squared plus two m g R over m g and can cancel out the m's. v squared plus two g R over g.

That would be my answer.

Now what will v be, it starts from rest, v is going to be zero because that's the least, no can't be zero.

I know what I've done with this, stated the obvious, it doesn't give you the least value for h. It gives you the value of h

which depends on v

The smallest that can be is if v was zero. I don't like that because that's going to make this thing to cancel down to two R which means that it starts at the same height that it goes up to and that wouldn't work

If it starts at the same, it just rolls back up to that height because its got rotational energy. So its not going to have enough energy to keep moving kinetically. Which is why v zero if it happens.

I'm not sure what to do with this to get the least value for h.

[please keep verbalising]

I've already hit the least value of h. It just depends on that v so I'm stuck. I don't know. I think I'm on the right track but I don't know where to go.

What would h have been if the ball were to slide along a frictionless track instead of rolling.

Then we won't have half I omega squared, we just have h equals half m v squared plus two m g R over m g which equals, now what does it equal ?

$$I = \frac{2}{5} m r^2$$

$$\omega^2 = \frac{v^2}{r^2}$$

$$h = \frac{\frac{1}{2} \frac{2}{5} \frac{v^2}{r^2} + \frac{1}{2} m v^2 + m g 2R}{m g}$$

$$= \frac{m v^2 + 2 m g R}{m g}$$

$$= \frac{v^2 + 2 g R}{g}$$

$$h = \frac{\frac{1}{2} m v^2 + 2 m g R}{m g}$$

Divide out the m we get half v squared plus two g R
over g. So v squared over two g plus two R.

$$= \frac{\frac{1}{2} v^2 + 2gR}{g}$$
$$= \frac{v^2}{2g} + 2R$$

Alright, I've got v squared in this. The place I get
stuck is finding

out what v is in both those parts.

Okay, I don't know how to carry on. I'll stop there.

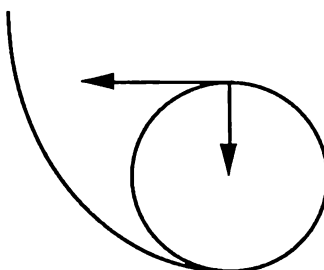


Figure 1

APPENDIX B1.5

TRANSCRIPT OF AN UNSUCCESSFUL NOVICE SOLUTION (N1).

A uniform ball of radius r rolls without slipping along the loop the loop track shown in the figure. Okay, is that raised to the height h above the bottom of the loop? If the ball is not to leave the track at the top of the loop, what is the least value h can have in terms of the radius R of the loop. What would h have been if the ball were to slide along a frictionless track instead of rolling ? I was looking at the picture again, my first idea would be to use conservation of energy. Right, centre of rest, where it starts from rest got a certain amount of potential energy. At the bottom of the loop that will be converted all to kinetic energy. As it goes back up potential energy again then kinetic energy as it rolls of. The first intuitive idea that I had would be that h would have to be the diameter of the loop at least in order to get round. Okay, uniform ball, there are two parts to the question. One's differentiation between rolling without slipping and its rolling as if it was a frictionless track so its sliding. Learnt something similar before, trying to recall it. Want to find minimum h such that the ball is not to leave the track at the top of the loop. Two parts, one is if a ball rolls without slipping, two frictionless	$h_{\min}$ ball not to leave the track at the top Ball rolls without slipping frictionless track
---	---

Seem to recall a similar problem. Just like a thorny part to it, had a subtlety in that. I think when the, see a ball rolling without slipping, I don't think there's any energy lost due to friction.

Remember solving a problem like this before, we've got one contact point. Might be profitable to start off with the second part. So if you would have a rotational component that comes into it and if the inertia of the ball and etc plus the gravitational just to show the any components right. Then the second part of the ball. If we just slide along a frictionless track and its not rolling then you would be able to, be able to reduce the ball to just a point mass sliding along a track, a frictionless track in which case you could say that by energy conservation.

There's no friction and its a closed system then you would have to start off at twice R, in order to, but if you start at exactly twice R, the ball, reach the bottom of the loop, start going up the loop, at the top of the loop will recover all the lost potential energy, collect potential energy and it will stop exactly at the top of the loop in which case it will fall down.

Has to be slightly greater than twice R. The second part I will put down for now h minimum got to be greater than two R.

There's a problem with that because let's say if it is just a little bit greater than two R then you could have the ball starts falling but not necessarily stays on the track rather than just stopping at the top.

So long as energy is taken up it sort of goes through, do a sort of chord across the circle. I can't write on this can I ?

I'll just redraw it, play around with it. The ball would experience a centripetal force as it were. If that force becomes less than zero then it will leave the track.

Drew figure 1.

$$h_{\min} > 2R$$

Drew figure 2.

So the second part we could that in terms of the centrifugal force

[please keep verbalising]

Not sure, this thing, how to face the problem in terms of the centrifugal. Somehow I am not happy with that. Its just sort of intuitive. How I feel for it, this ball will come across there and will just drop part of the way. The way you have to phrase it to see that it remains in contact all the time we have to observe how much force there is perpendicular to the circle or along the radial direction of the circle.

Out that way, call that N, so if is N is less than zero then ball leaves track.

If want to connect angular velocity, could some normal component

A similar system will be say a string with a mass on the end twirling.

We want to know what's the normal force that way rotate.

Now in part one you have a ball. Seem like a ball along the track perpendicular so one point of contact. There shouldn't, in this case I don't think there is any friction, any energy lost to friction either because of the rolling motion.

In terms of the view of the ball the whole apparatus is simply going round it. So, just trying to remember how, probably solve this case. The ball has got certain inertia. A point on the circumference will rotate with same omega instantaneously.

Centripetal, mass m g it falls down that, would provide a certain torque on the ball. So this sphere is going to resolve into perpendicular and normal components.

$$T - mg\cos\theta = ma_c = \frac{mv^2}{r}$$
$$a_c = \frac{v}{r}$$

Drew figure 3.

If $N < 0$ ball leaves the track

(simple pendulum analogy)

Drew figure 4

Labelling figure 4.

That's N and that will be your acceleration along the track a.

I remember there was a very nifty way of solving it. Being frustrated with it can't remember it. Think you can take it from the reference point on the contact, take it instantaneously then you would have, that's little r the radius of the ball. In which case torque, instantaneous torque, the point of contact will be a certain big I times A I think

$$\tau = IA$$

Now let's, so starts at rest, the initial, it has got initial potential energy which was one half h. What's the formula for potential energy in terms of height. Is it one half m g h ? No its just m g h isn't it? m g h.

$$E_{\text{pot}} = mgh$$

See if, potential at the bottom here's going to be, have a rotational energy in the ball and you going to have a linear kinetic energy.

At the top we going to convert it to potential energy okay. Read the question again make sure I have not missed anything out.

A uniform ball radius r, rolls without slipping the loop track in the figure. Starts at rest at a height h above the bottom of the loop. If the ball is not to leave the track at the top of the loop what is the least value h can have in terms of the radius R of the loop? What would h have been if slide instead of rolling?

So initial information we have I would say would be about potential energy which would be m g h (Checking book)

$$P_i = mgh$$

Conservation of energy. Let's see, rotation in space. Let's look at two O five and this two O four, two O five, see a picture of a pendulum here. Sort of relating back to this case here.

In order for the ball not to fly off it will exert certain force upwards which is due to the tendency of this to go to a linear motion. It has got to be balanced by a certain tension inwards like that, these two will be equal and opposite.

Over here I have found sort of similar problem that ties up the mass. Finds out the tension, just read the question of that one, see if it is applicable

In a way this is a sort of bob of mass m attached to a string of length l . It is pulled aside so that the string makes an angle θ with the vertical and it is released from rest. What is the speed at the bottom of the string and the tension of the string there?

Forces acting at the bottom, neglecting air resistance, force of gravity which is conservative and tension T which is perpendicular to the motion and does no work

The mechanical energy of the bob is therefore conserved in this problem. What is its gravitational potential energy, to be zero at the bottom of the swing.

At original position at height h above the bottom. The mass has a potential energy mgh and no kinetic energy because it is at rest.

As it swings down the potential energy is converted into kinetic energy, gives us speed at the bottom, half $m v^2 = mgh$.

According to figure seven point two the distance h is related to the original angle θ by that, the distance, alright does, take the force from that. The tension at the bottom is found as in example five seven.

At this point the mass has centripetal acceleration v^2/r . Due to acceleration v^2/l . Forces acting, T up mg down, result of the force gives that.

(Referring to the simple pendulum example in the text)

This problem is also solved using Newton’s laws, one twenty four, actual circle is not constant, acceleration tangential, the circle is not constant, it varies with theta and therefore time is difficult all this to multiply.

(Looking at problem five seven) Just trying to figure out the notation of the book. Example five four, six five seven. A block on a string moves in a vertical circle and string makes an angle theta with the vertical. Find the acceleration and tension in the string.

Suppose in the case of a frictionless case the model of this is a string going round like that got a certain kinetic energy and we want to see how long it takes to the top. The kinetic energy at the bottom will be given by the potential energy at the top.

You want to write tension and you want the tension of the string to be not less than zero instead of being the limiting case which just goes round.

Have a radial component of Newton’s second law. Tension minus $m g \cos \theta$, mass v , tension $m g \cos \theta$, m let’s see equals to $m v^2$ squared over r . It is centripetal acceleration. Right so $\cos \theta$ ’s will be minus $m g$.

Question four part two, model has string and the mass on the end of the string gives us the tension. Start off with total energy equal to that, potential energy which equals $m g h$ at the bottom of the swing. Bottom total energy will be just kinetic energy one half $m v^2$ squared $m g h$.

Now from that, we can find out what v is at the bottom perpendicular to there, which have v is going to be two g , the square root of two $g h$. Now in line with that analogy you can relate the v at the bottom to the tension If you have enough, check set up the right way. So equation’s got, that’s tension minus $m g \cos \theta$ will equal $m v^2$ squared over R .

Drew figure 5.
 Start off with
 $E_T = P_T = mgh$
 At bottom
 $E_T = E_k = \frac{1}{2} m v^2 = mgh$
 $v = \sqrt{2gh}$
 $T - mg \cos \theta = \frac{m v^2}{R}$

Equation page one thirty six, so the bottom cos zero is one so tension minus m g equals m v squared. You want

[please keep verbalising]

See if this set up is the way I want it, tension in string cosine theta m v. Find the acceleration and the tension, set up in the same way. So theta from the vertical.

So the tension is going to be equal to m v squared over R plus m g, that's, the cos theta hundred and eighty degrees will be minus one, so going to be minus m g and force perpendicular to it is going to be equal and opposite to the tension so its going to minus T. So its going to be minus, plus there, N is equal to m g minus m v squared upon R.

Now when that is less than or when that is equal to zero that would be the minimum, that would be the minimum time at which it is remain in contact.

See if I missed anything. So that will give you a v squared will equal m g, minus m g times R. v squared is equal to two g h. Two g h

[please keep verbalising]

I was just following the argument, see if I have done it correctly, think so. So at the condition where the normal force, normal component of the force is exactly zero at the top.

That will be achieved when you start off, when you start off from a total distance of two h or two R which is just the diameter over here so we get R is two, is one half R.

I think I tied myself up there because that will mean that if it started off a quarter of a distance up there it will make it across. So I think made a mistake somewhere. R would be two h, so h will be a half R

$$T = \frac{mv^2}{R} - mg$$

$$-T = -\frac{mv^2}{R} - mg$$

$$N = mg - \frac{mv^2}{R}$$

$$v^2 = mgR = 2gh$$

(cancels the m)

$$R = 2h$$

$$h = \frac{1}{2} R$$

This is, my original intuitive feeling was that it was going to be greater than two R where I was sceptical of whether it could be equal to two R but it is definitely not half R. So its gone the wrong way.

Following that one again, see how T minus m g cos theta equals m v squared upon R. So that times that m g cos theta minus that.

[please verbalise slightly louder]

So from that we can say that v squared is going to equal to, at the top, so I am going to look, what am I solving for?

I know the v is greater at the bottom so I am confused at the moment. I think I might have made my model wrong.

Do you know the initial velocity at the bottom? You want the tension, the tension in terms of the function theta, which is the same speed as that.

A vertical circle, when strand makes an angle theta with a vertical, speed is v.

Example five six, find the forces on the value v T.

These two are similar. They are basically the same thing In fact it is exactly the same, there's the force of gravity and the square root of R g. This pretty much the same thing.

This problem I, the level of speed the water can have at the top of the circle. N equal zero, okay the same, m g equals v T half.

Okay, so the answer in the book, hang on here v T or the minimum velocity is the square root of R g and the weight m g.

A bit lost, it could be. Yes it is two R, following this one hold on.

[please keep verbalising]

$$T - mg\cos\theta = \frac{mv^2}{R}$$

(Refers to the example of a pail being swung in a vertical circle)

$$v_{\min} = \sqrt{Rg}$$

It is greater than two R . It start off with potential energy at the bottom is all converted to velocity say if you start. If starts at exactly at two R then it will convert to velocity, the top here is all converted to potential energy see, cannot have any kinetic energy the velocity is zero.

Following by the example here is that the minimum velocity that you can have is still a positive number. So it, h has to be greater than two, two R in order to have a slight velocity there for this thing to go round without dropping when it reaches the top and the minimum velocity at the top is square root of $R g$

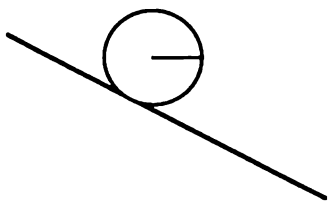


Figure 1

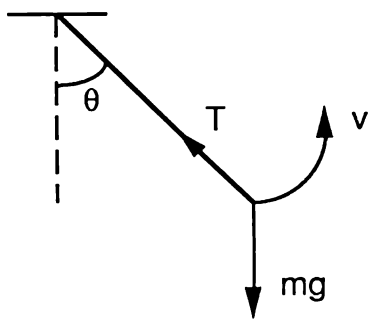


Figure 2

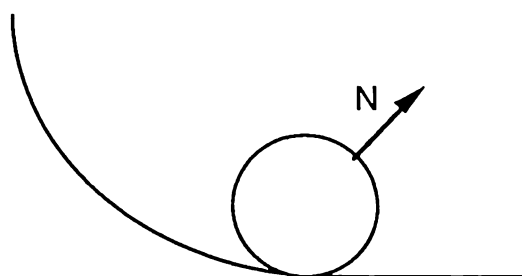


Figure 3

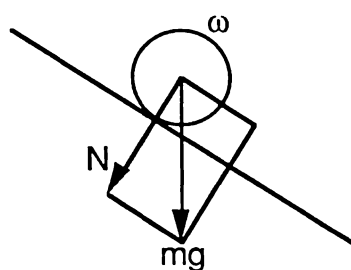


Figure 4

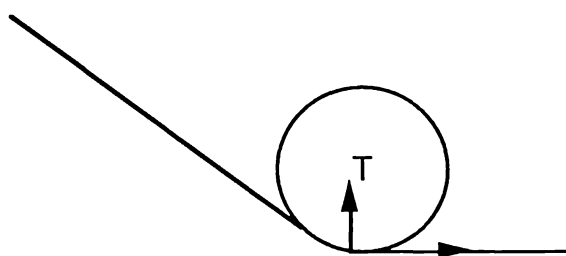
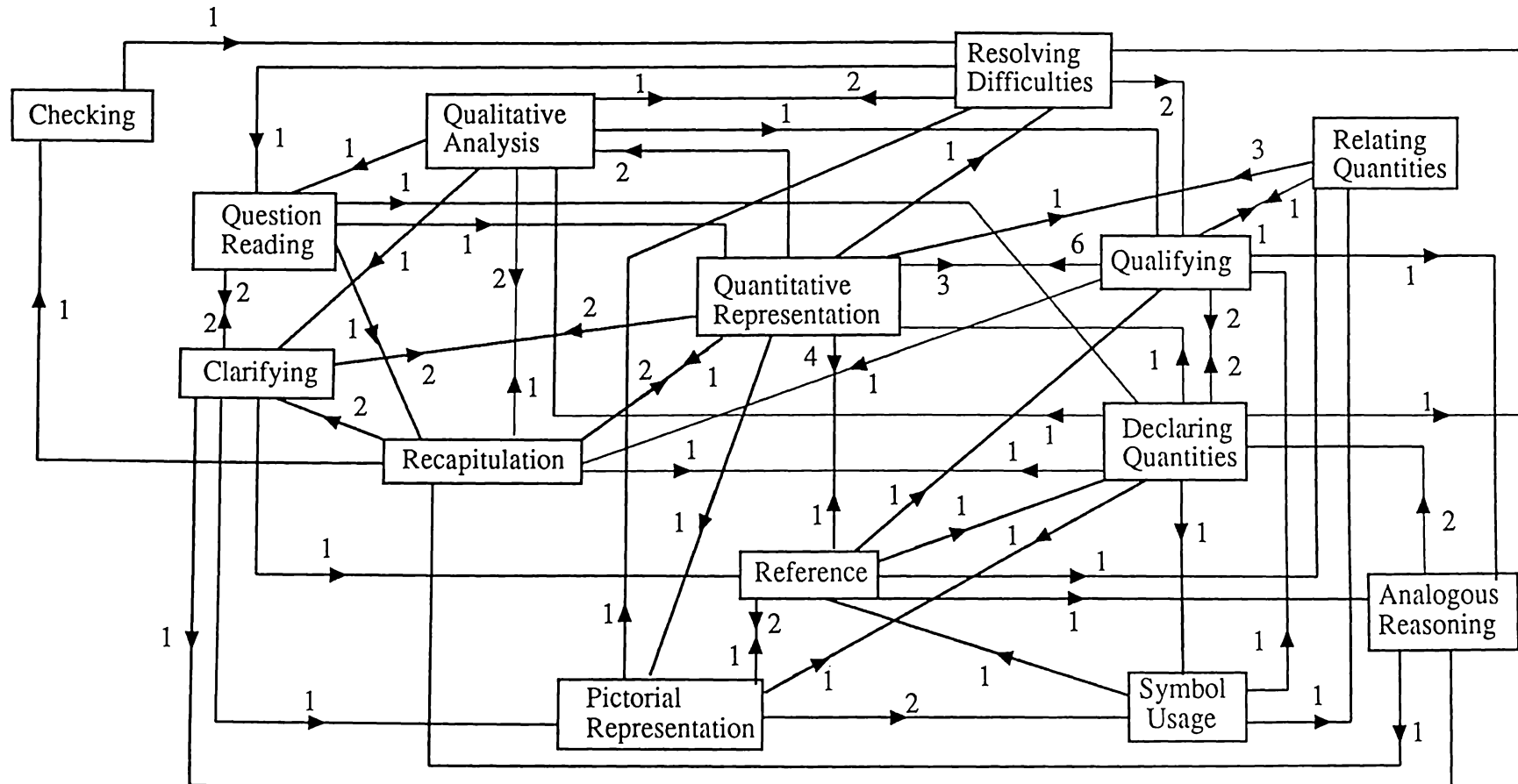


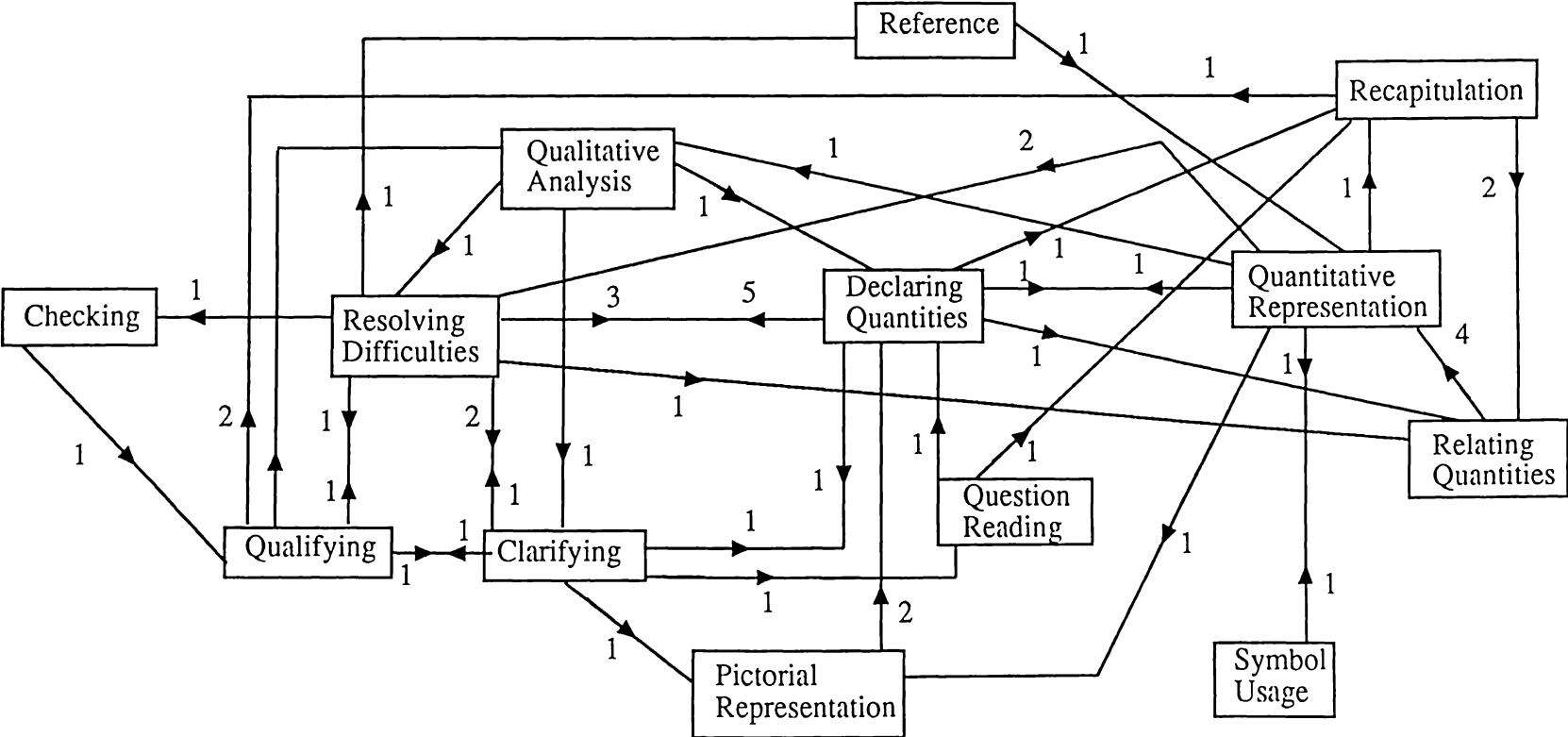
Figure 5

FREQUENCY FLOW CHART OF N1



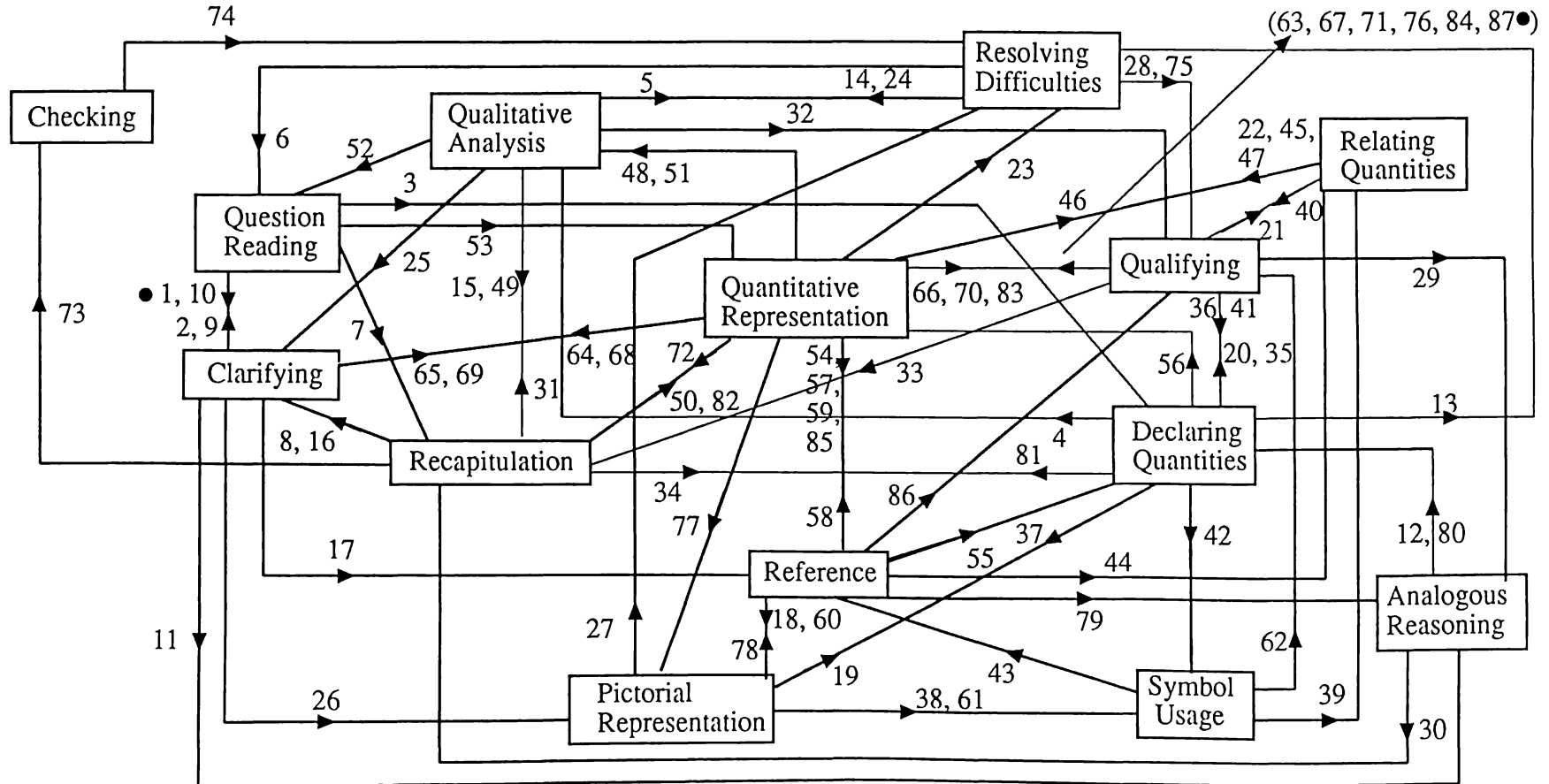
APPENDIX B2.2

FREQUENCY FLOW CHART OF NOVICE N8

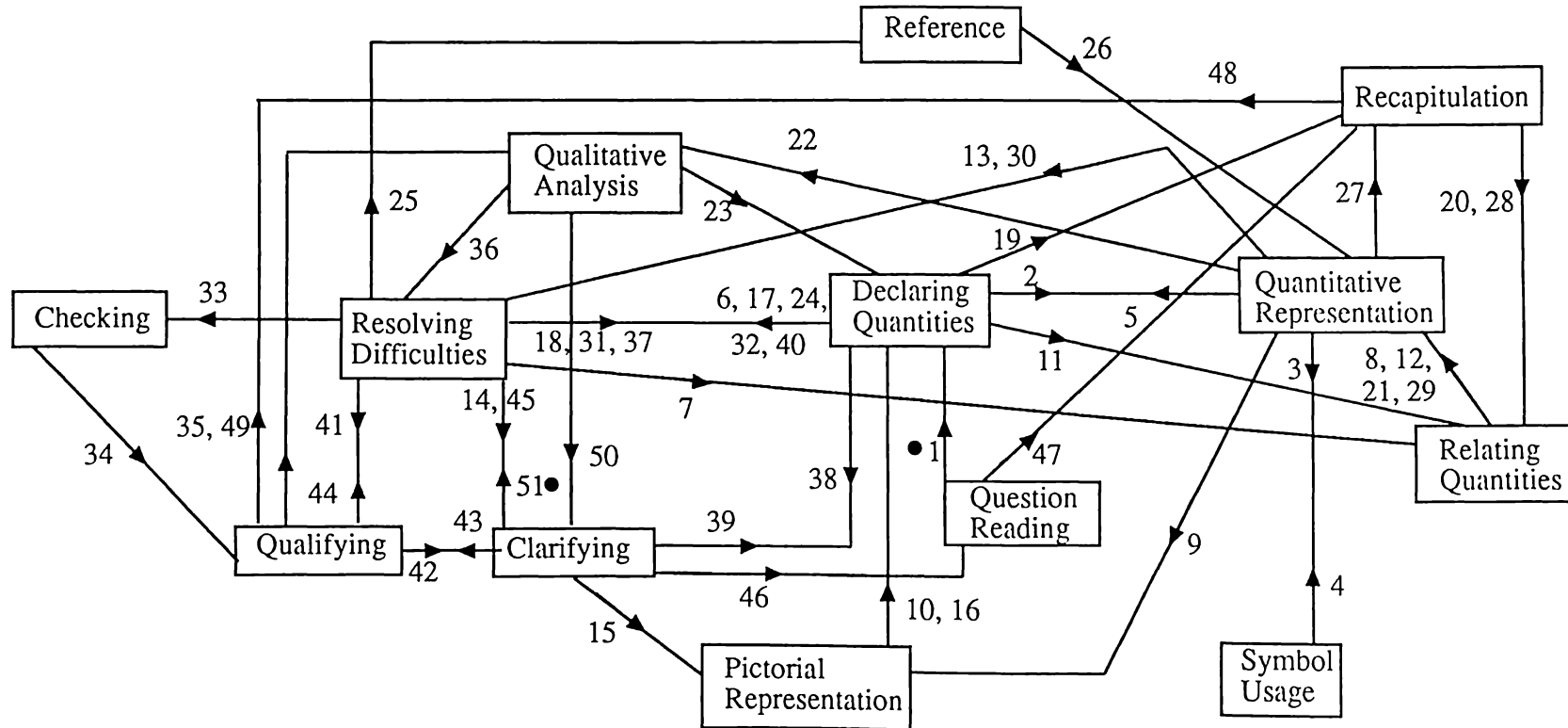


271

SEQUENTIAL FLOW CHART OF NOVICE N1



SEQUENTIAL FLOW CHART OF NOVICE N8



APPENDIX C

RELATED INFORMATION FOR THE COMPLETE SOLUTION OF THE PROBLEM.

References used to compile the information included in this appendix:

1. Physics for Students of Science and Engineering by A. L. Stanford and J. M. Tanner.
2. Physics by P. A. Tipler.
3. Physics by R. Resnick and D. Halliday.
4. Elementary Classical Physics by R. T. Weidner and R. L. Sells.

APPENDIX C1

C1.1 THE TRIGONOMETRIC RATIOS

For any right angled triangle:

The side opposite the right angle is called the hypotenuse.

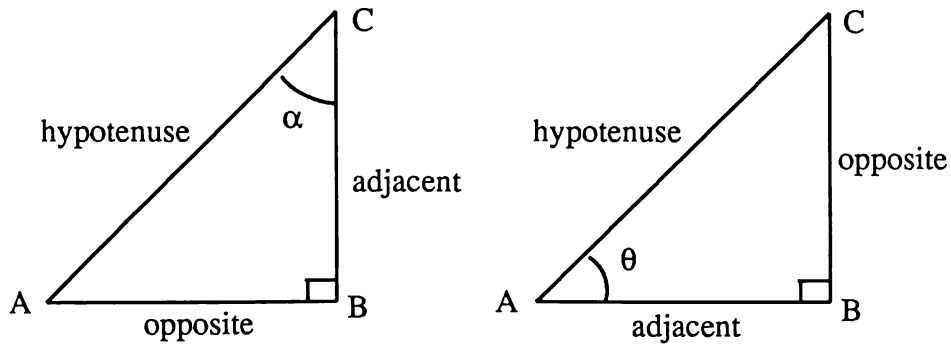


Figure C1.1: Sides of a triangle

The side opposite the angle α is AB. For this angle, BC is the adjacent side. For angle θ , BC is the opposite side and AB is the adjacent side.

The sine and cosine definitions are:

$$\text{sine of an angle} = \frac{\text{opposite}}{\text{hypotenuse}}$$

$$\text{cosine of an angle} = \frac{\text{adjacent}}{\text{hypotenuse}}$$

$$\text{e.g. } \cos \theta = \frac{AB}{AC}$$

$$\cos \alpha = \frac{BC}{AC}$$

$$\sin \theta = \frac{BC}{AC}$$

$$\sin \alpha = \frac{AB}{AC}$$

C1.2 RESOLUTION OF FORCES

Since forces are vectors the rule of vector addition is used to resolve them into components. This involves using a set of coordinate axes usually mutually perpendicular. Figure C1.2 shows a three dimensional set of mutually perpendicular

axes. These are called rectangular axes. In the problem, the motion of the ball is two dimensional. Therefore, it is sufficient to use two axes as shown in Figure C1.3.

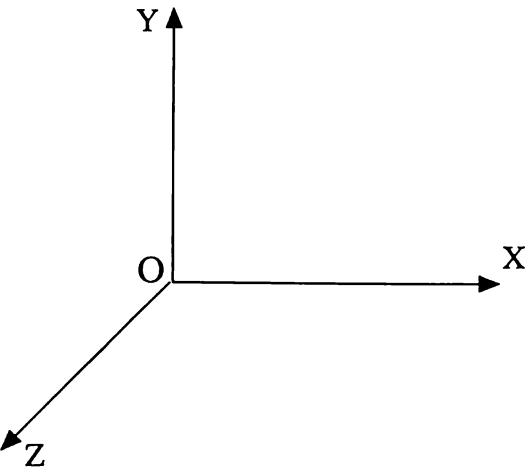


Figure C1.2: Rectangular axes

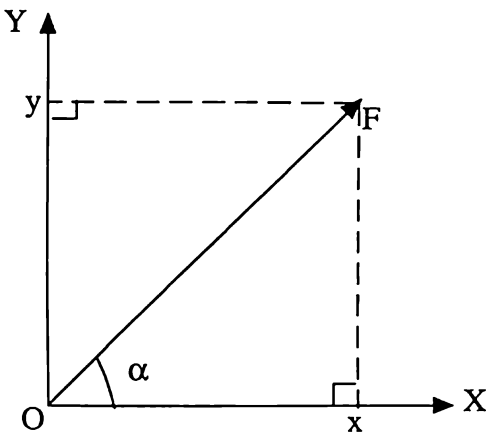


Figure C1.3: Two dimensional representation of rectangular axes.

Resolution is the process of replacing a two dimensional vector by two mutually perpendicular vectors using vector addition. For example, for a single vector **F** making an angle θ with the x-axis, **Ox** and **Oy** represent the two mutually perpendicular vectors along the x and y axes respectively (Figure C1.3) which add to give **F**. Obtaining these mutually perpendicular vectors is called resolving the vector into its components. For the vector **F**:

The vector **Ox** is called its X-component and its magnitude is $F\cos\alpha$

The vector **Oy** is called its Y-component and its magnitude is $F\sin\alpha$

Information on trigonometric ratios is given in Appendix C1.1.

A vector can have many sets of components since the X and Y axes can be arbitrarily chosen. To simplify the job of resolution, the following rules of thumb may be applied to assist in the choice of axes:

(i) Choose a rectangular set of axes; and

Either

(ii) Choose an axis to lie along the direction of motion; or

(iii) Choose the axes such that as many of the vectors lie along one of the two directions.

For the problem, the rectangular axes have been chosen along the normal and the tangential direction to the loop (Figure C1.4). N lies along the normal and will not have a component in the tangential directions since $N\cos 90^\circ$ is zero. The component of mg along the tangent is $mg\sin\theta$ and along the normal is $mg\cos\theta$ as shown below.

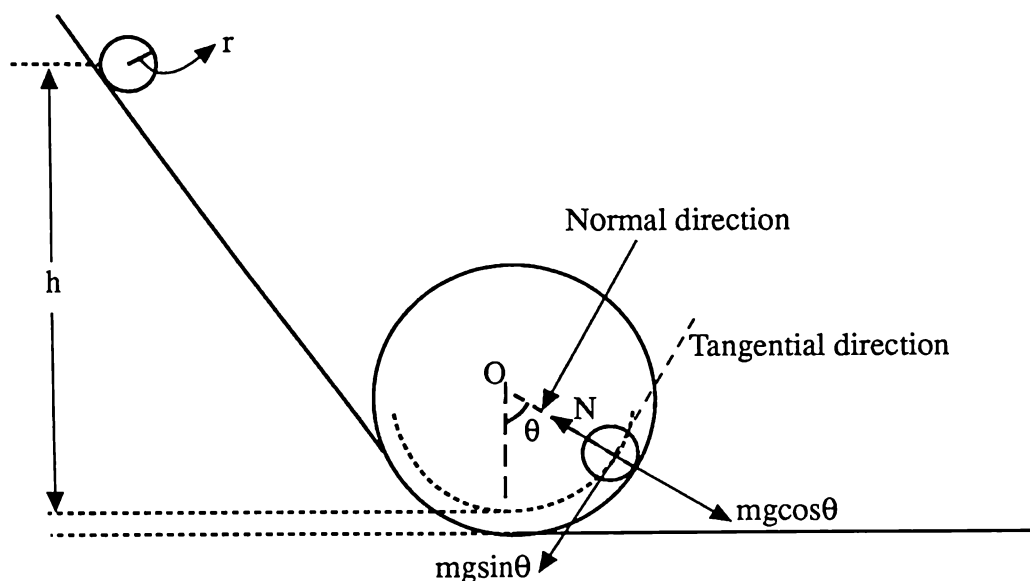


Figure C1.4: Resolution of the forces acting on the ball.

C1.3 GRAPH OF COSΘ

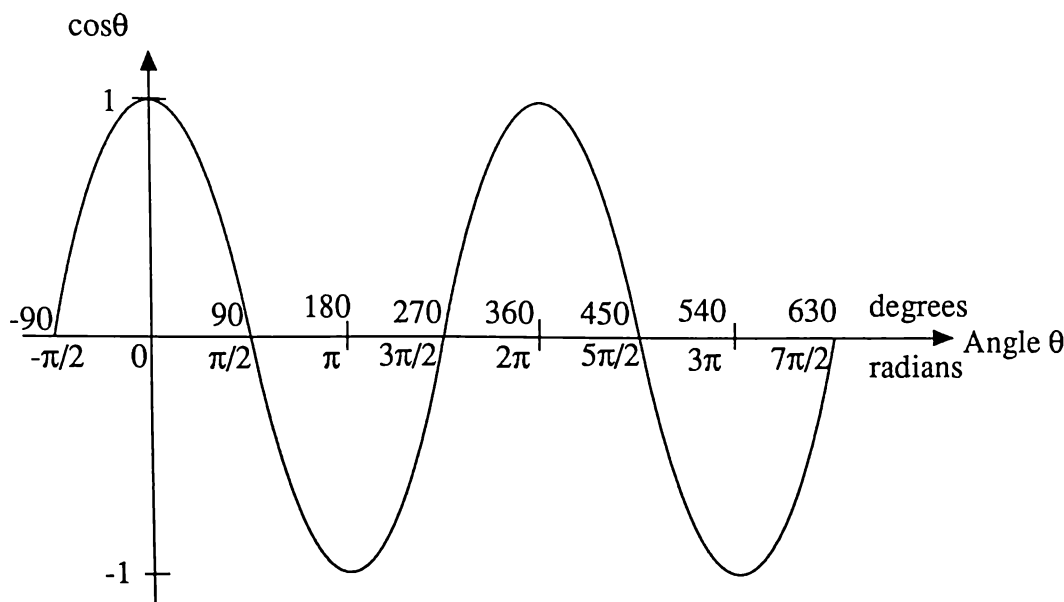


Figure C1.5: Cosine function

From Figure C1.5, $\cos\theta$ has a negative value for values of θ from 90° to 270° or $\frac{\pi}{2}$ to $\frac{3\pi}{2}$ radians.

C1.4 DOT AND CROSS PRODUCT

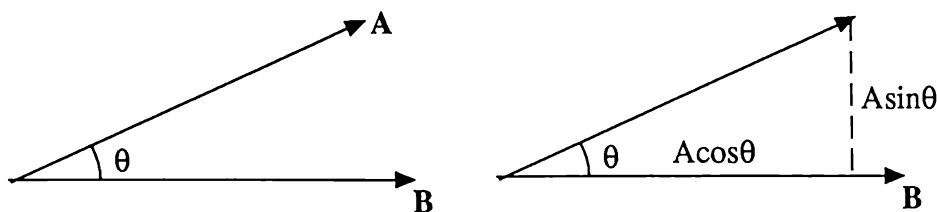


Figure C1.6: Resolving the vector to obtain a product.

The dot product of two vectors **A** and **B**, forming an angle θ with each other is denoted by $\mathbf{A} \cdot \mathbf{B}$ and is defined as $AB\cos\theta$

The dot product of two vectors give a scalar quantity i.e. it has a magnitude but no direction.

The magnitude of the cross (vector) product of two vectors **A** and **B**, forming an angle θ with each other is denoted by $|\mathbf{A} \times \mathbf{B}|$ and defined as $AB\sin\theta$

The vector product of two vectors points in a direction perpendicular to the plane containing the two vectors (Figure C1.7).

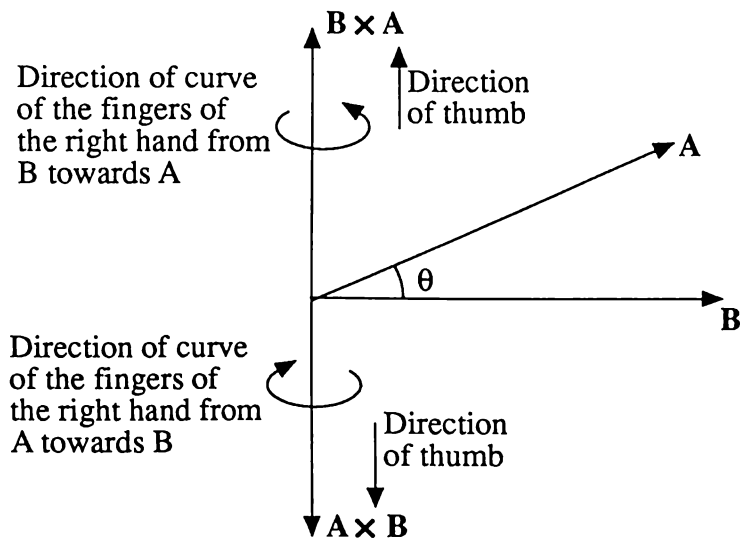


Figure C1.7: Direction of the vector product.

The right hand rule is used to determine the direction of the vector product. In Figure C1.7, $\mathbf{A} \times \mathbf{B}$ points in the direction of the thumb of the right hand when the fingers curve in the direction from $\mathbf{A}$ towards $\mathbf{B}$. The direction of curve of the fingers is given in Figure C1.7 by the arrowed curve. $\mathbf{B} \times \mathbf{A}$ points in the opposite direction as the fingers curving from $\mathbf{B}$ to $\mathbf{A}$ points the thumb in the opposite direction

C1.5 THE DERIVATIVE OF A QUANTITY.

The graph below depicts a general change in a physical quantity (z) over time. The slope of the line gives the change of the quantity over the time interval δt .

$$\text{Slope of graph} = \frac{z_f - z_i}{t_f - t_i}$$

$$\delta z = z_f - z_i$$

$$\delta t = t_f - t_i$$

$$\text{Slope of the graph} = \frac{\delta z}{\delta t}$$

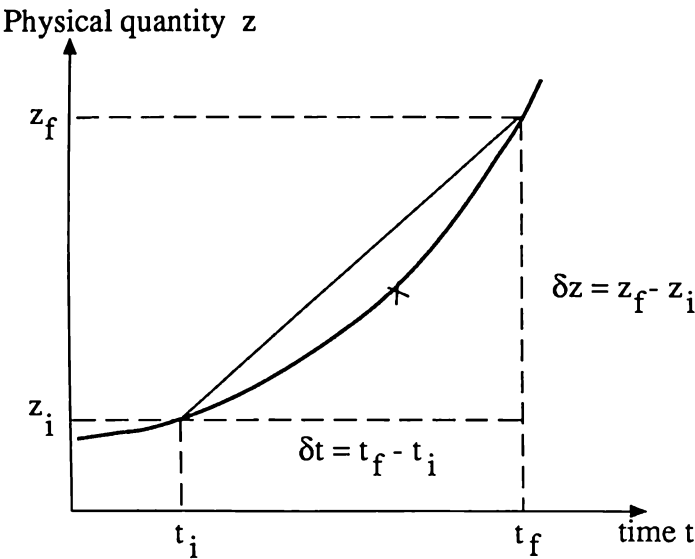


Figure C1.8: The change of a certain physical quantity z with time.

The rate of change of a physical quantity z over time is denoted by $\frac{dz}{dt}$ and is defined as

$$\frac{dz}{dt} = \lim_{\delta t \rightarrow 0} \frac{\delta z}{\delta t}$$

For a sufficiently small time interval the change may be considered as instantaneous. Then $\frac{dz}{dt}$ gives the rate of change of the quantity at an instant of time. This can be graphically depicted by drawing a tangent to the graph at the

instant of time. In Figure C1.8, the point X marks the centre of the interval δz . As $\delta z \rightarrow 0$ the two points z_1 and z_f converge towards the point marked X. Drawing a tangent to the graph at this point gives the rate of change at that time instant.

C1.6 EXPLANATION OF THE INTEGRAL

Let’s take a physical quantity A, which may be divided into n smaller parts. Let $\delta a_1, \delta a_2, \delta a_3, \delta a_4, …, \delta a_n$ represent the n parts which form the total quantity A. Then

$$\begin{aligned} A &= \delta a_1 + \delta a_2 + \delta a_3 + \delta a_4 + \dots + \delta a_n \\ &= \sum_{i=1}^n \delta a_i \end{aligned}$$

If A is divided into infinitesimally small parts then

$$\delta a_i \rightarrow 0$$

giving an infinite number of parts which constitute the physical quantity. Then

$$A = \lim_{\delta a_i \rightarrow 0} \sum_{i=1}^{\infty} \delta a_i$$

The summation is replaced by the integral when the sum is taken over an infinitesimal intervals, since

$$\lim_{\delta a_i \rightarrow 0} \sum_{i=1}^{\infty} \delta a_i = \int_0^{\infty} da$$

Therefore,

$$A = \int_0^{\infty} da$$

APPENDIX C2

DERIVATION OF $F = ma$

Newton's second law of motion states that the rate of change of momentum of a body is proportional to the resultant of the forces acting on it and takes place in the direction of the resultant force.

$$F \propto \frac{dp}{dt} \quad \text{where } F \text{ is the resultant force; and}$$
$$\frac{dp}{dt} \text{ is the rate of change of momentum.}$$

(Appendix C1.5 provides an explanation of the derivative)

$$F = k \frac{dp}{dt} \quad \text{where } k \text{ is a constant.}$$

By definition, the momentum of an object is given by

$$p = mv \quad \text{where } m \text{ is the mass of the object; and}$$
$$v \text{ is the velocity of the object.}$$

$$F = k \frac{d}{dt} (mv)$$
$$= k \frac{dm}{dt} v + km \frac{dv}{dt}$$

For a constant mass,

$$\frac{dm}{dt} = 0$$

$$F = km \frac{dv}{dt}$$

Acceleration is the rate of change of velocity. So,

$$a = \frac{dv}{dt} \quad \text{where } a \text{ is the acceleration}$$

$$\therefore F = kma$$

By definition, a Newton (the unit of force) is that force which will produce an acceleration of 1ms^{-2} when applied to a mass of 1kg. This gives

$$1\text{N} = k(1\text{kg})(1\text{m s}^{-2})$$

$$k = 1$$

$$\therefore F = ma$$

APPENDIX C3

C3.1 DERIVATION OF $s = r\theta$

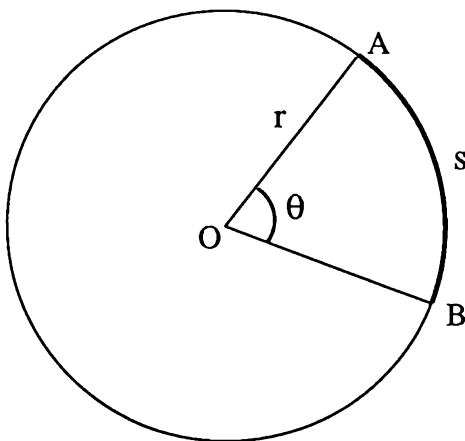


Figure C3.1: An object performing a circle of radius r

As an object performs circular motion, it undergoes an angular displacement of θ radians when it moves a length s along the arc AB. For one complete revolution, the object will undergo an angular displacement 2π radians or 360° . The arc length traversed will be $2\pi r$, the circumference of the circle i.e. an angular displacement of 2π radians corresponds to a displacement of $2\pi r$ along the circumference of the circle. Therefore, an angular displacement of θ radians will correspond to

$$\frac{\theta}{2\pi} 2\pi r = r\theta$$

along the circumference of the circle.

$$\therefore s = r\theta$$

C3.2 DERIVATION $v = \omega r$

Consider an object traversing a circular path (Figure C3.1). As the object moves from position A to B, the radius vector turns through an angle θ radians. The distance s moved by the object is given by

$$s = r\theta \quad (\text{Appendix C3.1})$$

Differentiating with respect to time t ,

$$\frac{ds}{dt} = r \frac{d\theta}{dt} \quad (\text{Appendix C1.5})$$

where $v = \frac{ds}{dt}$ is the tangential velocity; and

$\omega = \frac{d\theta}{dt}$ is the angular velocity about an axis through the centre O and perpendicular to the plane of rotation.

This gives,

$$v = r\omega$$

(Note: Linear velocity (v) is defined as the rate of change of linear displacement i.e. $\frac{ds}{dt}$ and angular velocity (ω) as the rate of change of angular displacement i.e. $\frac{d\theta}{dt}$)

C3.3: EXPLANATION OF ROLLING

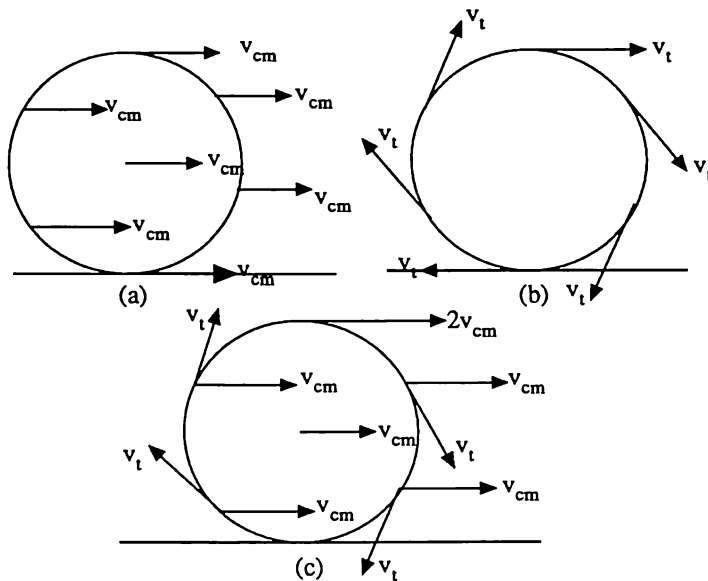


Figure C3.2: Tangential and centre of mass velocities of a rolling object

If the ball just slides then as shown in Figure C3.2a, each point on the ball will have a translational velocity, say, v_{cm} . If the ball is rotating about an axis through its centre of mass then each point on the ball will have a tangential velocity, say, v_t (Figure C3.2b). Only points on the circumference of the ball have been considered in the diagram. If the ball is rolling without sliding then at an instant of time, the point A

at the bottom of the ball is instantaneously at rest on the surface. This happens when $v_{cm} = v_t$. When the ball rolls, apart from the point instantaneously in contact with the ground, all points on it will have a tangential velocity v_t and a translational velocity v_{cm} as shown in Figure C3.2c. The ball moves forward with a translational velocity of v_{cm} .

For any circular motion,

$$v = \omega r \quad (\text{Appendix C3.2})$$

where v is the tangential velocity of the object as it performs circular motion;
 ω is the angular velocity; and
 r is the radius of the circle.

For rolling, since the translational velocity v_{cm} of the ball is equal to the tangential velocity, v , of a point on the circumference of the ball therefore

$$v_{cm} = \omega r$$

APPENDIX C4

DERIVATION OF $a = \frac{v^2}{r}$ FOR CIRCULAR MOTION.

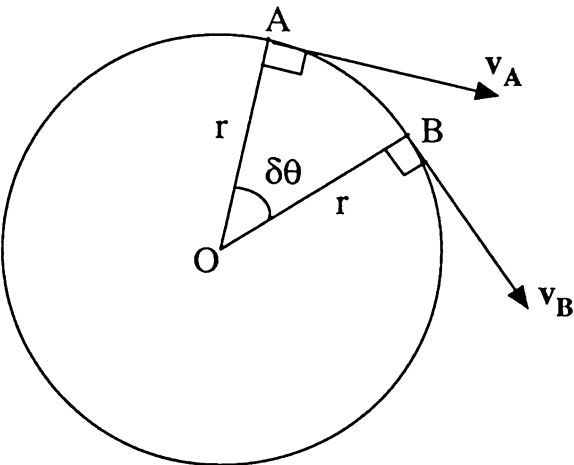


Figure C1.4: An object performing circular motion.

Consider an object traversing a circular path. As the object moves from A to B during a small time interval δt , the radius vector OA turns through a small angle $\delta\theta$. The tangential velocity vector v_A will also turn through an angle $\delta\theta$ during that small time interval, δt . The direction of the velocity will change. Over a very small time interval, the magnitude of the velocity may be assumed to remain constant even if there is a net resultant force in the tangential direction. The change in velocity δv due to the change in direction is shown below.

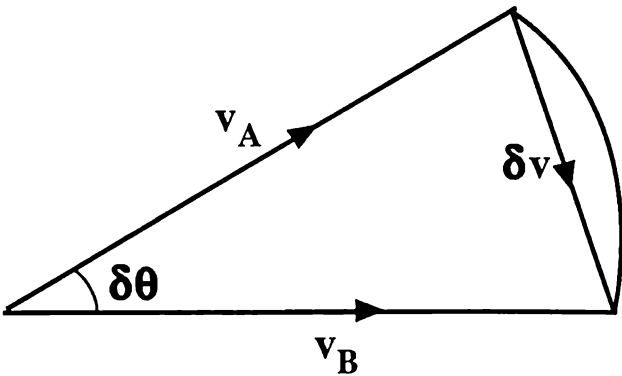


Figure C4.2: Vector diagram showing the change in velocity

If the angle turned, $\delta\theta$ is taken over a sufficiently small interval of time δt then

$$v_A \approx v_B \approx v \text{ (say)}$$

Then δv may be approximated to an arc of a circle with radius of magnitude v .

Using

$$s = r\theta \quad (\text{Appendix C3.1})$$

where θ is in radians

Then

$$\delta v = v \delta \theta$$

$$\frac{\delta v}{\delta t} = v \frac{\delta \theta}{\delta t}$$

For a sufficiently small time interval (Appendix C1.5) for an explanation on the derivative)

$$\delta t \rightarrow 0 \text{ and } \delta \theta \rightarrow 0.$$

$$\lim_{\delta t \rightarrow 0} \frac{\delta v}{\delta t} = v \lim_{\delta t \rightarrow 0} \frac{\delta \theta}{\delta t}$$

$$\lim_{\delta t \rightarrow 0} \frac{\delta v}{\delta t} = \frac{dv}{dt} = a \quad \text{and} \quad \lim_{\delta t \rightarrow 0} \frac{\delta \theta}{\delta t} = \frac{d\theta}{dt} = \omega$$

where a and ω are the linear acceleration and angular velocity respectively.

Thus

$$a = v\omega$$

Substituting

$$v = r\omega \quad (\text{Appendix C3.2})$$

$$a = \frac{v^2}{r}$$

APPENDIX C5

KINETIC ENERGY IS GIVEN BY $\frac{1}{2} mv^2$

A force acting on an object can do work on it and give energy to it. Here it is assumed that all the energy given through work done by the force goes towards increasing the kinetic energy of the object. This will occur if there are no frictional losses and the object travels along a horizontal level.

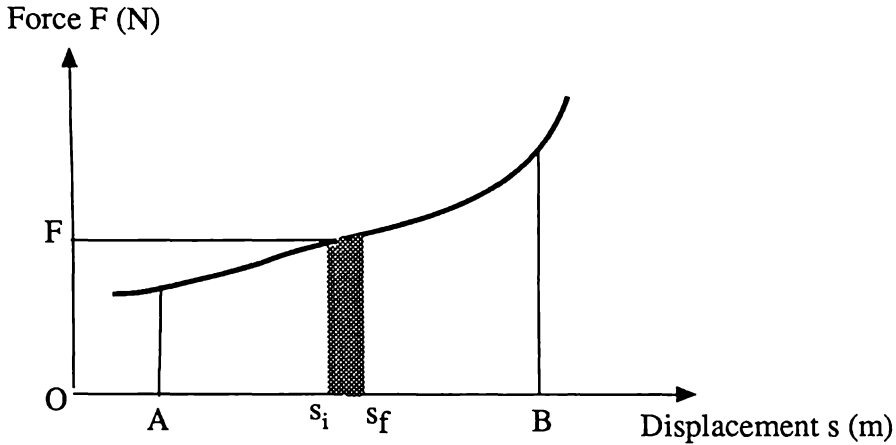


Figure C5.1: Force-displacement graph.

By definition work done is given by

$$\text{W.D.} = \int_{s_i}^{s_f} \mathbf{F} \cdot d\mathbf{s}$$

where $\mathbf{F}$ is the force vector and

$d\mathbf{s}$ is an infinitesimal displacement vector.

(see Appendix C1.4 for explanation on dot product and C1.6 for an explanation of the integral)

Assuming $\mathbf{F}$ acts in the direction of the displacement then

$$\text{W.D.} = \int_{s_i}^{s_f} F ds$$

From Newton's second law, for a constant mass

$$F = ma$$

(Appendix C2)

where m is the mass of the object on which the force is acting and

a is its acceleration if it moves under the action of the force.

$$W.D. = \int_{s_i}^{s_f} ma \, ds$$

By definition, acceleration is the rate of change of velocity.

So

$$a = \frac{dv}{dt} \quad (\text{Appendix C1.5})$$

$$\begin{aligned} W.D. &= m \int_{s_i}^{s_f} \frac{dv}{dt} \, ds \\ &= m \int_{v_i}^{v_f} v \, dv \\ &= m \left[\frac{1}{2} v^2 \right]_{v_i}^{v_f} \\ &= \frac{1}{2} mv_f^2 - \frac{1}{2} mv_i^2 \end{aligned}$$

The terms $\frac{1}{2} mv_f^2$ and $\frac{1}{2} mv_i^2$ are defined as the final and initial kinetic energies of the object. The equation tells us that the work done by a force causes the kinetic energy of the object to be changed provided no other kind of energy conversion takes place.

APPENDIX C6

POTENTIAL ENERGY IS GIVEN BY mgh

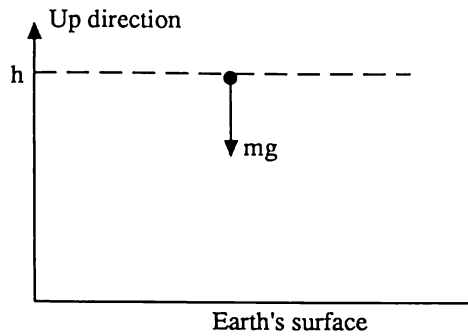


Figure C6.1: An object at a height h above the surface of the earth.

For situations whereby an object is near the surface of the earth the magnitude of g , the free fall acceleration due to gravity, may be considered constant. Thus a constant gravitational force mg will act on the object. As shown in Figure C6.1, the earth's surface is taken as the reference level for measurement of vertical displacement. At a vertical position (i.e. $y = h$) the gravitational potential energy of an object is defined as the work the gravitational force would do on an object in moving it from the position $y = h$ to the position $y = 0$ (the reference level).

Potential energy

= work done by the gravitational force in moving object from $y = h$ to $y = 0$

$$= \int_0^h \mathbf{F} \cdot d\mathbf{s} \qquad \text{(Appendix C6)}$$

Since the force and displacement vectors are parallel

$$\text{P.E.} = \int_0^h (-mg) ds$$

The gravitational force mg is negative because it acts downwards and the direction of the y -axis has been taken as positive.

$$\begin{aligned} \text{P.E.} &= [-mgs]_h^0 \\ &= mgh \end{aligned}$$

APPENDIX C7

DEFINITION OF THE MOMENT OF INERTIA, I.

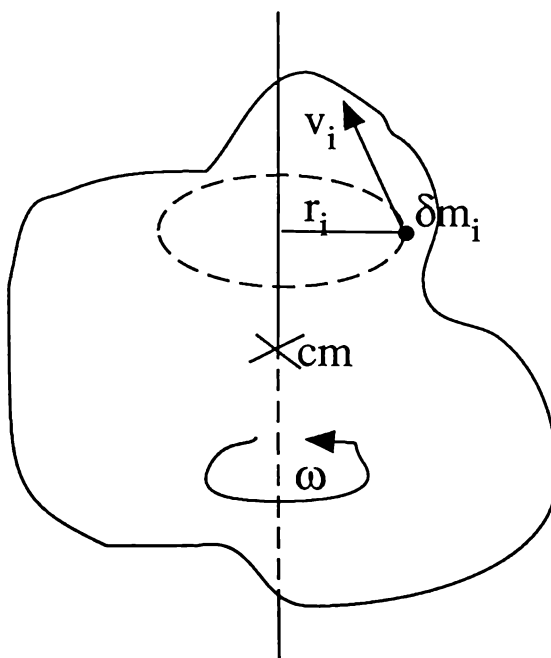


Figure C7.1: A rigid body rotating about an axis

A rigid body is composed of a continuous distribution of infinitesimally small differential masses δm_i . I is the rotational inertia or moment of inertia of a body about a particular axis. For a particle of mass δm_i at a distance r_i from the axis of rotation, the moment of inertia is defined as:

$$I_i = \delta m_i r_i^2$$

The total moment of inertia of the body about the axis of rotation is the sum of the moments of inertia of all the particles constituting the body. This is given by

$$\begin{aligned} I &= I_1 + I_2 + I_3 + \dots \\ &= \delta m_1 r_1^2 + \delta m_2 r_2^2 + \delta m_3 r_3^2 + \dots \\ &= \sum_{i=1}^{\infty} \delta m_i r_i^2 \end{aligned}$$

As $\delta m_i \rightarrow 0$

Then

$$\begin{aligned} I &= \lim_{\delta m \rightarrow 0} \sum_{i=0}^{\infty} \delta m_i r_i^2 \\ &= \int_0^{\infty} r^2 \, dm \end{aligned}$$

This is normally written without the limits as

$$I = \int r^2 \, dm$$

where r is the distance of the infinitesimal mass from the axis of rotation. The integral sums the contributions of all the infinitesimal masses to give the total moment of inertia of the body. The limits can be arbitrarily chosen dependent on the application.

The rotational inertia of a body depends on:

- (a) the axis about which it is rotating;
- (b) the shape of the body; and
- (c) the mass distribution of the body.

APPENDIX C8

ROTATIONAL KINETIC ENERGY IS $\frac{1}{2}I\omega^2$

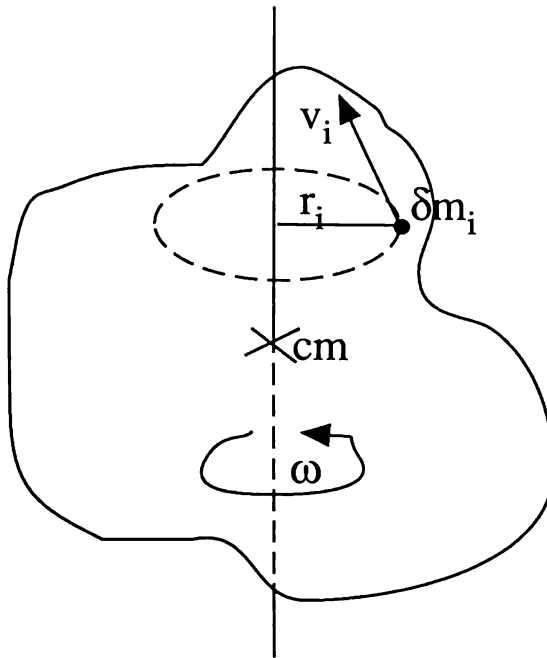


Figure C8.1: A rigid body rotating about an axis.

Consider a body rotating about an axis with an angular velocity of ω . The body can be considered as composed of infinitesimally small particles each of mass δm_i . The rotational kinetic energy of each particle will be given by

$$K_i = \frac{1}{2} \delta m_i v_i^2 \quad (\text{Appendix C5})$$

where v_i is the tangential velocity of each particle and δm_i is its mass.

For circular motion

$$v_i = \omega r_i$$

So

$$K_i = \frac{1}{2} \delta m_i r_i^2 \omega^2$$

For a rigid body, each particle constituting the body will rotate round the axis with the same angular velocity ω .

[Rigid body: A rigid body is one in which the distance between each particle of the body remains constant with respect to all other particles of the body.]

The total rotational kinetic energy of the body is the sum of the rotational kinetic energies of all the particles constituting the body.

Therefore,

$$\begin{aligned} K &= K_1 + K_2 + K_3 \dots\dots\dots \\ &= \frac{1}{2} \delta m_1 r_1^2 \omega^2 + \frac{1}{2} \delta m_2 r_2^2 \omega^2 + \frac{1}{2} \delta m_3 r_3^2 \omega^2 + \dots\dots\dots \\ &= \sum \frac{1}{2} \delta m_i r_i^2 \omega^2 \\ &= \frac{1}{2} (\sum \delta m_i r_i^2) \omega^2 \end{aligned}$$

As $\delta m_i \rightarrow 0$

$$\sum \delta m_i r_i^2 = \int r^2 \, dm \qquad \qquad \qquad \text{(Appendix C1.6)}$$

Therefore,

$$K = \frac{1}{2} (\int r^2 \, dm) \omega^2$$

Since

$$I = \int r^2 \, dm \qquad \qquad \qquad \text{(Appendix C7)}$$

$$K = \frac{1}{2} I \omega^2$$

Using work done:

In rotational dynamics the rotational kinetic energy may be calculated using the work done by the torque in causing the turning effect. This is given by the area under the graph for a torque-angular displacement graph.

By definition, the work done is given by

$$W.D. = \int \Gamma \, d\theta \qquad \qquad \qquad \text{(Appendix C9)}$$

where Γ is the torque acting on the body.

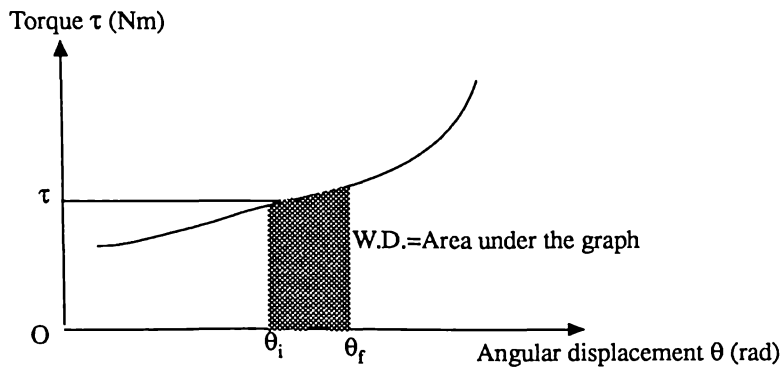


Figure C8.2: Torque-angular displacement graph.

Since

$$\Gamma = \frac{dL}{dt} \quad (\text{Appendix C10 and C11})$$

where $\frac{dL}{dt}$ is the rate of change of angular momentum.

$$\text{W.D.} = \int \frac{dL}{dt} d\theta$$

$$= \int \omega dL$$

Since

$$L = I\omega \quad (\text{Appendix C12})$$

$$dL = I d\omega$$

$$\text{W.D.} = \int \omega I d\omega$$

$$= \frac{1}{2} I \omega^2$$

APPENDIX C9

DERIVATION OF $W.D. = \int \Gamma \, d\theta$

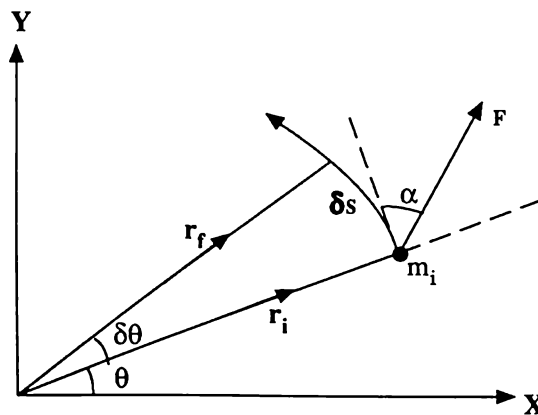


Figure C9.1: Displacement of a particle due to the action of a force.

By definition work done is given by

$$W.D. = \int \mathbf{F} \cdot d\mathbf{s}$$

where $\mathbf{F}$ is the force vector and $d\mathbf{s}$ is the infinitesimal displacement vector.

If the force acts at an angle α to the displacement direction as shown in Figure C9.1 then from the definition of the dot (scalar) product (Appendix C1.4)

$$W.D. = \int F \cos \alpha \, ds$$

For a small displacement δs ,

$$r_f \approx r_i \approx r$$

Since

$$s = r\theta \tag{Appendix C3.1}$$

$$\delta s = r_i \, \delta \theta$$

where δs is an infinitesimal linear displacement and $\delta \theta$ is the associated angular displacement of the point mass m_i at a distance r from the axis of rotation.

$$\text{If } \delta \theta \rightarrow 0 \tag{Appendix C1.6}$$

This gives

$$\text{W.D.} = \int (F \cos \alpha) r_i d\theta$$

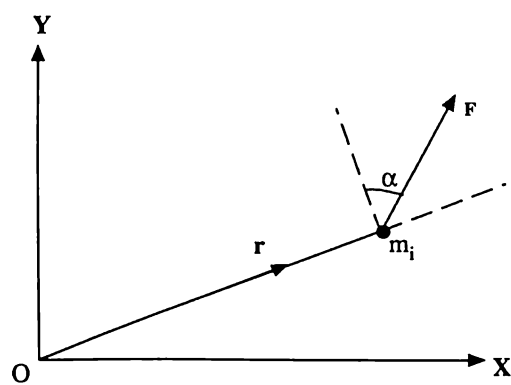


Figure C9.2: A force acting on a particle to produce a torque.

By definition

$$\Gamma = \mathbf{r} \times \mathbf{F}$$

A force $\mathbf{F}$ acting on a point mass m_i at a displacement $\mathbf{r}_i$ with respect to an inertial reference frame gives a torque

$$\Gamma = \mathbf{r}_i \times \mathbf{F} \qquad \qquad \qquad (\text{Appendix C1.4})$$

$$\Gamma = r_i F \sin(90 - \alpha)$$

$$= r_i F \cos \alpha$$

Therefore,

$$\text{W.D.} = \int \Gamma d\theta$$

APPENDIX C10

DERIVATION OF $\Gamma = \frac{dL}{dt}$

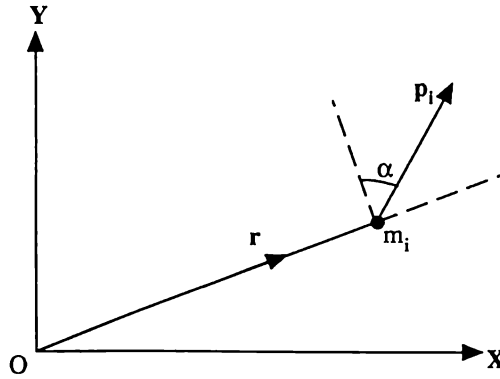


Figure C10.1: A particle moving with respect to a fixed inertial frame.

A particle of mass m_i having a linear velocity v_i will have a linear momentum p_i given by

$$\mathbf{p}_i = m_i \mathbf{v}_i$$

The angular momentum l_i , of a particle is defined as

$$\mathbf{l}_i = \mathbf{r}_i \times \mathbf{p}_i$$

where $\mathbf{r}_i$ is the displacement vector of a particle from an inertial reference frame and $\mathbf{p}_i$ is its linear momentum.

The magnitude of the angular momentum is

$$l_i = r_i p_i \sin\alpha \quad (\text{Appendix C1.4})$$

$$\frac{d\mathbf{l}_i}{dt} = \frac{d}{dt} (\mathbf{r}_i \times \mathbf{p}_i)$$

$$\frac{d\mathbf{l}_i}{dt} = \frac{d\mathbf{r}_i}{dt} \times \mathbf{p} + \mathbf{r}_i \times \frac{d\mathbf{p}_i}{dt}$$

From Newton's second law of motion,

$$\mathbf{F} = \frac{d\mathbf{p}}{dt} \quad (\text{Appendix C2})$$

$$\mathbf{v}_i = \frac{d\mathbf{r}_i}{dt}$$

$$\frac{d\mathbf{l}_i}{dt} = \mathbf{v}_i \times (m_i \mathbf{v}_i) + \mathbf{r}_i \times \mathbf{F}_i$$

$$\mathbf{v}_i \times (m_i \mathbf{v}_i) = 0$$

Giving

$$\frac{d\mathbf{l}_i}{dt} = \mathbf{r}_i \times \mathbf{F}_i$$

Therefore , $\frac{d\mathbf{l}_i}{dt} = \boldsymbol{\Gamma}_i$

The total angular momentum, $\mathbf{L}$, of the body is given by the sum of all the angular momenta of the individual particles constituting the body.

$$\mathbf{L} = \mathbf{l}_1 + \mathbf{l}_2 + \mathbf{l}_3 + \mathbf{l}_4 + \dots\dots\dots$$

$$\frac{d\mathbf{L}_i}{dt} = \frac{d\mathbf{l}_1}{dt} + \frac{d\mathbf{l}_2}{dt} + \frac{d\mathbf{l}_3}{dt} + \frac{d\mathbf{l}_4}{dt} + \dots\dots\dots$$

$$\begin{aligned} \frac{d\mathbf{L}_i}{dt} &= \sum \frac{d\mathbf{l}_i}{dt} \\ &= \sum \boldsymbol{\Gamma}_i \end{aligned}$$

The sum of the torques on a body includes external as well as internal torques.

$$\sum \boldsymbol{\Gamma}_i = \sum \boldsymbol{\Gamma}_{\text{int}} + \sum \boldsymbol{\Gamma}_{\text{ext}}$$

where $\sum \boldsymbol{\Gamma}_{\text{int}}$ and $\sum \boldsymbol{\Gamma}_{\text{ext}}$ is the sum of the internal and external torques respectively.

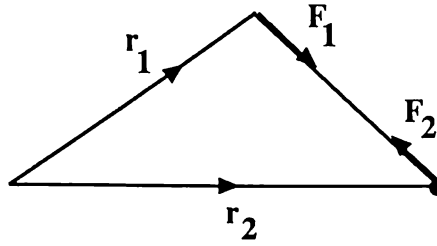


Figure C10.2: The internal force between two particles of a body.

Consider two particles of a body as shown in Figure C10.2. Let the final internal forces the particles exert on each other be $\mathbf{F}_1$ and $\mathbf{F}_2$. By Newton's third law

$$\mathbf{F}_1 = -\mathbf{F}_2$$

The sum of the torques exerted by the two forces about an axis through O is

$$\begin{aligned}\sum \Gamma_1 + \sum \Gamma_2 &= \mathbf{r}_1 \times \mathbf{F}_1 + \mathbf{r}_2 \times \mathbf{F}_2 \\ &= \mathbf{r}_1 \times \mathbf{F}_1 + \mathbf{r}_2 \times (-\mathbf{F}_1) \\ &= (\mathbf{r}_1 - \mathbf{r}_2) \times \mathbf{F}_1\end{aligned}$$

The vector $(\mathbf{r}_1 - \mathbf{r}_2)$ lies along the line joining the two particles and is parallel or anti-parallel to $\mathbf{F}_1$. Therefore,

$$(\mathbf{r}_1 - \mathbf{r}_2) \times \mathbf{F}_1 = 0$$

If this holds true for all the internal forces between the particles of the body, then

$$\sum \Gamma_{\text{int}} = 0$$

Giving

$$\frac{d\mathbf{L}_1}{dt} = \Gamma_{\text{ext}}$$

where Γ_i is the torque acting on each particle; and

Γ_{ext} is the resultant external torque acting on the body.

This relation has been derived for torque about a stationary axis. This relation also holds for a moving axis through the centre of mass of the body (Appendix C11).

APPENDIX C11

$$\frac{dL_i}{dt} = \sum \Gamma_{ext}$$

FOR A MOVING AXIS THROUGH THE CENTRE OF MASS OF A BODY.

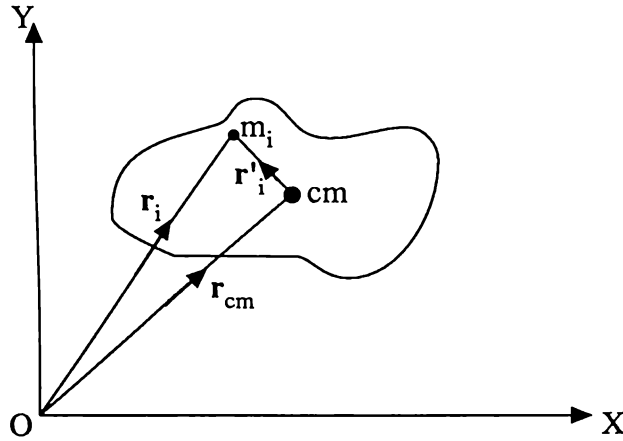


Figure C11.1: A body moving with respect to a fixed axis.

Let $\mathbf{r}_{cm}$ be the position vector of the centre of mass of the body with respect to the origin of a stationary inertial reference frame. Let $\mathbf{r}'_i$ be the position vector of a particle of mass m_i , with respect to the centre of mass and $\mathbf{r}_i$ be its position vector with respect to the stationary inertial reference frame.

$$\mathbf{r}_i = \mathbf{r}_{cm} + \mathbf{r}'_i$$

The linear momentum, $\mathbf{p}'_i$, of m_i with respect to the centre of mass is

$$\mathbf{p}'_i = m_i \mathbf{v}'_i$$

where $\mathbf{v}'_i$ is the tangential velocity of rotation of m_i with respect to the centre of mass.

$$\mathbf{v}'_i = \frac{d\mathbf{r}'_i}{dt}$$

$$\mathbf{p}'_i = m_i \frac{d\mathbf{r}'_i}{dt}$$

$$\begin{aligned}
&= m_i \frac{d}{dt} (\mathbf{r}_i - \mathbf{r}_{cm}) \\
&= m_i \frac{d\mathbf{r}_i}{dt} - m_i \frac{d\mathbf{r}_{cm}}{dt} \\
&= \mathbf{p}_i - m_i \mathbf{v}_{cm}
\end{aligned}$$

The total angular momentum of the body with respect to its centre of mass is

$$\begin{aligned}
\mathbf{L}' &= \sum \mathbf{r}'_i \times \mathbf{p}'_i \\
\frac{d\mathbf{L}'_i}{dt} &= \sum \left(\frac{d\mathbf{r}'_i}{dt} \times \mathbf{p}'_i + \mathbf{r}'_i \times \frac{d\mathbf{p}'_i}{dt} \right) \\
\frac{d\mathbf{r}'_i}{dt} \times \mathbf{p}'_i &= \mathbf{v}'_i \times m_i \mathbf{v}'_i \\
&= m_i (\mathbf{v}'_i \times \mathbf{v}'_i) \\
&= 0
\end{aligned}$$

since the cross product of parallel vectors is zero (Appendix C1.4)

$$\begin{aligned}
\frac{d\mathbf{L}'_i}{dt} &= \sum \mathbf{r}'_i \times \frac{d\mathbf{p}'_i}{dt} \\
\frac{d\mathbf{r}_i}{dt} &= \frac{d\mathbf{r}_{cm}}{dt} + \frac{d\mathbf{r}'_i}{dt} \\
\mathbf{v}_i &= \mathbf{v}_{cm} + \mathbf{v}'_i \\
m_i \mathbf{v}_i &= m_i \mathbf{v}_{cm} + m_i \mathbf{v}'_i \\
\mathbf{p}_i &= \mathbf{p}_{cm} + \mathbf{p}'_i \\
\frac{d\mathbf{p}_i}{dt} &= \frac{d\mathbf{p}_{cm}}{dt} + \frac{d\mathbf{p}'_i}{dt} \\
\frac{d\mathbf{p}'_i}{dt} &= \frac{d\mathbf{p}_i}{dt} - \frac{d\mathbf{p}_{cm}}{dt}
\end{aligned}$$

$$\begin{aligned}
\frac{d\mathbf{L}'_i}{dt} &= \sum \mathbf{r}'_i \times \left(\frac{d\mathbf{p}_i}{dt} - \frac{d\mathbf{p}_{cm}}{dt} \right) \\
&= \sum \left(\mathbf{r}'_i \times \frac{d\mathbf{p}_i}{dt} - \mathbf{r}'_i \times \frac{d\mathbf{p}_{cm}}{dt} \right) \\
\mathbf{r}'_i \times \frac{d\mathbf{p}_{cm}}{dt} &= \mathbf{r}'_i \times \frac{d}{dt} (m_i \mathbf{v}_{cm}) \\
&= \mathbf{r}'_i \times \left(\frac{dm_i}{dt} \times \mathbf{v}_{cm} + m_i \frac{d\mathbf{v}_{cm}}{dt} \right)
\end{aligned}$$

For a constant mass,

$$\begin{aligned}
\frac{dm_i}{dt} &= 0 \\
\mathbf{r}'_i \times \frac{d\mathbf{p}_{cm}}{dt} &= \mathbf{r}'_i \times m_i \frac{d\mathbf{v}_{cm}}{dt}
\end{aligned}$$

Corresponding to the centre of mass

$$\sum m_i \mathbf{r}'_i = 0$$

Therefore,

$$\begin{aligned}
\frac{d\mathbf{L}'_i}{dt} &= \mathbf{r}'_i \times \frac{d\mathbf{p}_i}{dt} \\
\mathbf{F}_i &= \frac{d\mathbf{p}_i}{dt} \quad (\text{Appendix C2}) \\
\frac{d\mathbf{L}'_i}{dt} &= \mathbf{r}'_i \times \mathbf{F}_i \\
&= \sum \Gamma'_{ext}
\end{aligned}$$

assuming that all the internal torques cancel (Appendix C10).

Therefore, the rate of change of angular momentum is equal to the external torque calculated with respect to an axis through the centre of mass.

APPENDIX C12

DERIVATION OF $L = I\omega$

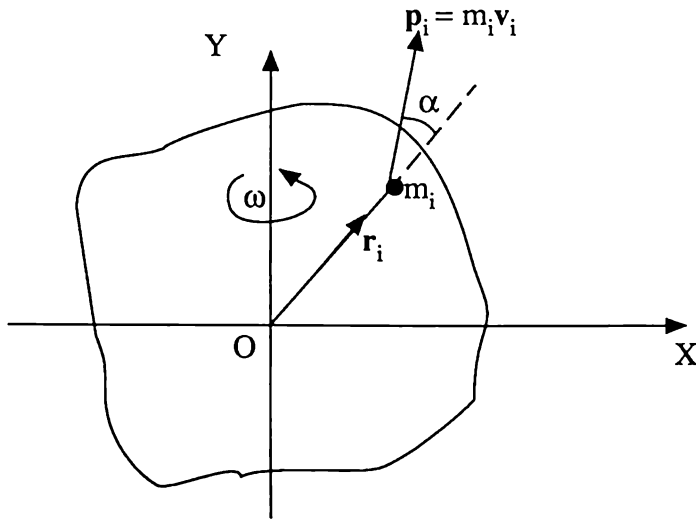


Figure C12.1: A rigid body rotating about a fixed axis.

Consider a rigid body rotating about an axis with an angular velocity ω . All the particles constituting the body will rotate with the same angular velocity ω .

The angular momentum of a particle of mass m_i , will be given by

$$\mathbf{l}_i = \mathbf{r}_i \times \mathbf{p}_i$$

where $\mathbf{p}_i$ is its linear momentum.

Taking $\mathbf{v}_i$ to be the tangential velocity of the particle

$$\mathbf{l}_i = \mathbf{r}_i \times m_i \mathbf{v}_i$$

The magnitude of the angular momentum of the particle is

$$l_i = r_i m_i v_i \sin\alpha \quad (\text{Appendix C1.4})$$

Since $\mathbf{v}_i$ is the tangential velocity, $\alpha = 90^\circ$

$$l_i = r_i m_i v_i$$

$$v_i = r_i \omega$$

$$l_i = r_i m_i r_i \omega$$

$$= m_i r_i^2 \omega$$

The total angular momentum of the body is

$$L = l_1 + l_2 + l_3 + \dots$$

The magnitude of the total angular momentum is

$$L = l_1 + l_2 + l_3 + \dots$$

$$= m_1 r_1^2 \omega + m_2 r_2^2 \omega + m_3 r_3^2 \omega + \dots$$

$$= \sum m_i r_i^2 \omega$$

$$= \left(\sum m_i r_i^2 \right) \omega$$

Therefore,

$$L = I\omega$$

where I is the moment of inertia of the body about the axis of rotation.

APPENDIX C13

MOMENT OF INERTIA OF A HOOP ABOUT AN AXIS THROUGH ITS CENTRE AND PERPENDICULAR TO ITS PLANE IS mR^2

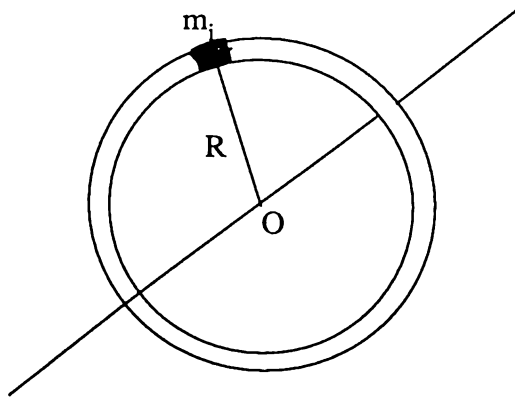


Figure C13.1: An axis of rotation through the centre of the hoop and perpendicular to its plane.

Let the mass of the hoop be m and its radius be R .

Take a point element of mass m_i .

The moment of inertia of the hoop is the sum of all the point elements which make up the hoop.

$$\begin{aligned} I &= m_1 R^2 + m_2 R^2 + m_3 R^2 + \dots \\ &= (m_1 + m_2 + m_3 + \dots) R^2 \\ &= (\sum m_i) R^2 \end{aligned}$$

Therefore,

$$I = m R^2$$

APPENDIX C14

THE MOMENT OF INERTIA OF A DISC ABOUT AN AXIS THROUGH THE CENTRE OF MASS AND PERPENDICULAR TO ITS PLANE IS $\frac{1}{2} mR^2$

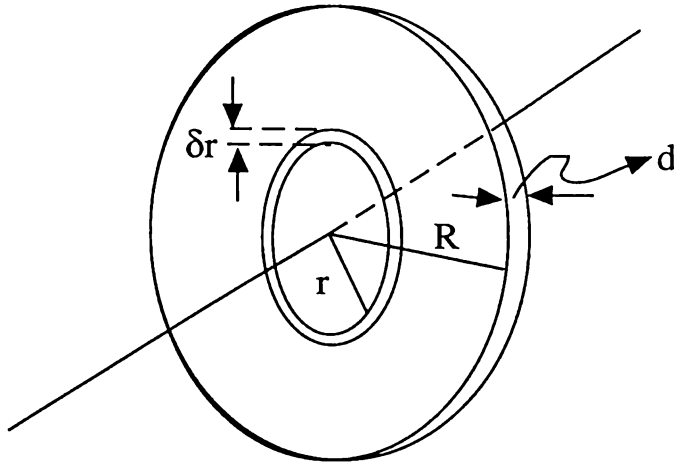


Figure C14.1: A ring element to determine the moment of inertia of a disc.

Mass of disc = m

Radius of disc = R

Thickness of the disc = d

Take a hoop as a mass element of mass δm , radius r and width δr .

$$\text{Density of disc } \rho = \frac{m}{Ad}$$

where A is the area of cross section of the disc.

$$\delta m = \rho \delta V$$

where δV is the volume of the element.

$$\begin{aligned}\delta m &= \frac{m}{Ad} 2\pi r \cdot \delta r \cdot d \\ &= \frac{m}{\pi R^2 d} 2\pi r \cdot \delta r \cdot d \\ &= \frac{m}{R^2} 2r \cdot \delta r\end{aligned}$$

Moment of Inertia of a hoop about an axis through the centre of mass and perpendicular to its plane is mr^2 (Appendix C13)

For the hoop element

$$\begin{aligned}\delta I &= \delta m \cdot r^2 \\ &= \frac{m}{R^2} 2\pi r^2 \delta r\end{aligned}$$

The moment of inertia of the disc is

$$\begin{aligned}I &= \int_0^R \frac{m}{R^2} 2\pi r^3 \cdot dr \\ &= \frac{2m}{R^2} \left[\frac{1}{4} r^4 \right]_0^R \\ &= \frac{1}{2} m R^2\end{aligned}$$

APPENDIX C15

MOMENT OF INERTIA OF A SOLID SPHERE ABOUT AN AXIS THROUGH ITS CENTRE OF MASS IS $I = \frac{2}{5} MR^2$

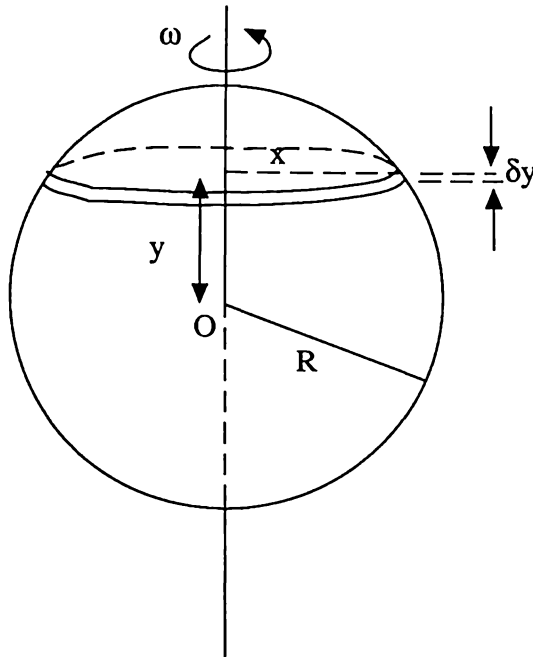


Figure C15.1: A solid sphere with an axis through its centre.

The sphere can be treated as being made up of a set of discs of different radii x stacked one above the other.

Let the mass of the sphere be M , its radius be R , and its density be ρ .

$$\text{Volume of the sphere} = \frac{4}{3} \pi R^3$$

$$\begin{aligned} \rho &= \frac{M}{\frac{4}{3} \pi R^3} \\ &= \frac{3M}{4\pi R^3} \end{aligned}$$

Take a disc element of thickness δy and centred at a distance y above the centre of mass of the sphere.

Moment of inertia of the disc is given by

$$\frac{1}{2} m r^2 \quad (\text{Appendix C14})$$

where m is the mass of the disc and
 r is its radius.

$$\begin{aligned}
 \delta I &= \frac{1}{2} \frac{3M}{4\pi R^3} \pi x^2 \cdot \delta y \cdot x^2 \\
 &= \frac{3M}{8R^3} x^4 \cdot \delta y \\
 I &= \int_{-R}^R \frac{3M}{8R^3} (R^2 - y^2)^2 dy \\
 &= 2 \int_0^R \frac{3M}{8R^3} (R^2 - y^2)^2 dy \\
 &= \frac{3M}{4R^3} \int_0^R (R^2 - y^2)^2 dy \\
 &= \frac{3M}{4R^3} \int_0^R (R^4 - 2y^2 R^2 + y^4) dy \\
 &= \frac{3M}{4R^3} \left[R^4 y - \frac{2}{3} R^2 y^3 + \frac{1}{5} y^5 \right]_0^R \\
 &= \frac{3M}{4R^3} \left(R^5 - \frac{2}{3} R^5 + \frac{1}{5} R^5 \right) \\
 &= \frac{3M}{4R^3} \frac{15R^5 - 10R^5 + 3R^5}{15} \\
 &= \frac{3M}{4R^3} \frac{8R^5}{15}
 \end{aligned}$$

Therefore,

$$I = \frac{2}{5} MR^2$$

APPENDIX C16

THE TOTAL KINETIC ENERGY OF A ROTATING BODY MAY BE CONSIDERED TO BE THE SUM OF THE TRANSLATIONAL KINETIC ENERGY OF ITS CENTRE OF MASS AND ITS ROTATIONAL KINETIC ENERGY RELATIVE TO THE CENTRE OF MASS

$$K = \frac{1}{2} m v^2 + \frac{1}{2} I \omega^2$$

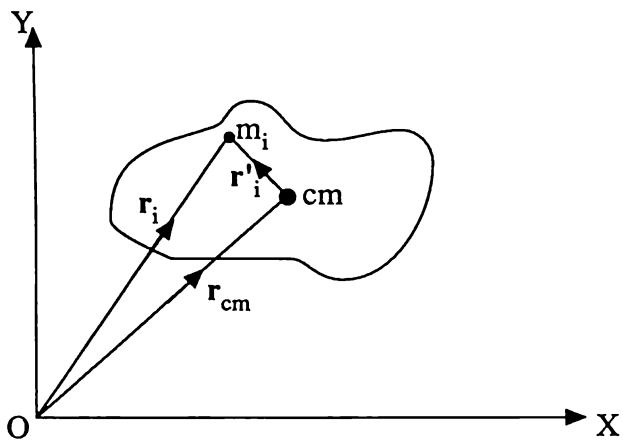


Figure C16.1: A rigid body rotating about a fixed axis.

Let $\mathbf{r}_{cm}$ be the position vector of the centre of mass of the body from O, the origin of a stationary inertial reference frame. Let $\mathbf{r}'_i$ and $\mathbf{r}_i$ be the position vectors of an infinitesimal particle of mass m_i from the centre of mass and the origin O respectively.

The velocity of the particle:

relative to the stationary inertial frame is

$$\mathbf{v}_i = \frac{d\mathbf{r}_i}{dt}$$

and relative to the centre of mass is

$$\mathbf{v}'_i = \frac{d\mathbf{r}'_i}{dt}.$$

The velocity of the centre of mass relative to the inertial reference frame is

$$\mathbf{v}_{cm} = \frac{d\mathbf{r}_{cm}}{dt}$$

$$\mathbf{r}_i = \mathbf{r}_{cm} + \mathbf{r}'_i$$

$$\frac{d\mathbf{r}_i}{dt} = \frac{d\mathbf{r}_{cm}}{dt} + \frac{d\mathbf{r}'_i}{dt}$$

$$\mathbf{v}_i = \mathbf{v}_{cm} + \mathbf{v}'_i$$

The three velocity vectors lie in a plane perpendicular to an axis of rotation through the centre of mass.

The kinetic energy of the particle relative to the stationary inertial reference frame is given by

$$\begin{aligned} K_i &= \frac{1}{2} m_i v_i^2 \\ &= \frac{1}{2} m_i (\mathbf{v}_i \cdot \mathbf{v}_i) \\ &= \frac{1}{2} m_i (\mathbf{v}_{cm} + \mathbf{v}'_i) \cdot (\mathbf{v}_{cm} + \mathbf{v}'_i) \\ &= \frac{1}{2} m_i (v_{cm}^2 + v_i'^2 + 2 \mathbf{v}_{cm} \cdot \mathbf{v}'_i) \end{aligned}$$

The total kinetic energy of all the particles constituting the body is

$$\begin{aligned} K &= \sum \frac{1}{2} m_i v_i^2 \\ &= \sum \frac{1}{2} m_i (v_{cm}^2 + v_i'^2 + 2 \mathbf{v}_{cm} \cdot \mathbf{v}'_i) \\ &= \frac{1}{2} \sum m_i v_{cm}^2 + \frac{1}{2} \sum m_i v_i'^2 + \sum m_i (\mathbf{v}_{cm} \cdot \mathbf{v}'_i) \\ &= \frac{1}{2} v_{cm}^2 \sum m_i + \frac{1}{2} \sum m_i v_i'^2 + \sum m_i (\mathbf{v}_{cm} \cdot \mathbf{v}'_i) \\ M &= \sum m_i \end{aligned}$$

where M is the total mass of the object.

$$\begin{aligned} \sum m_i (\mathbf{v}_{cm} \cdot \mathbf{v}'_i) &= \mathbf{v}_{cm} \cdot \sum m_i \mathbf{v}'_i \\ &= \mathbf{v}_{cm} \cdot \sum \frac{d}{dt} (m_i \mathbf{r}'_i) \end{aligned}$$

$$= \mathbf{v}_{\text{cm}} \cdot \frac{d}{dt} \sum m_i \mathbf{r}'_i$$

$\sum m_i \mathbf{r}'_i = 0$ as it is the position of the centre of mass with respect to itself.

Therefore,

$$K = \sum \frac{1}{2} m_i v_i'^2 + \frac{1}{2} M v_{\text{cm}}^2$$

The first term gives the kinetic energy of all the particles relative to the centre of mass of the object. The second term represents the kinetic energy of the body due to a translational motion of the centre of mass.

For rolling,

$$\mathbf{v}'_i = \mathbf{r}'_i \omega \quad (\text{Appendix C3.3})$$

If the body is rigid then all the particles will have the same angular velocity relative to the centre of mass.

$$\begin{aligned} K &= \frac{1}{2} \sum m_i (r_i'^2 \omega^2) + \frac{1}{2} M v_{\text{cm}}^2 \\ &= \frac{1}{2} (\sum m_i r_i'^2) \omega^2 + \frac{1}{2} M v_{\text{cm}}^2 \\ &= \frac{1}{2} I \omega^2 + \frac{1}{2} M v_{\text{cm}}^2 \end{aligned}$$

where $I = \sum m_i r_i'^2$ is the moment of inertia of the body about an axis through the centre of mass.

APPENDIX D1

User manual for the rotational dynamics problem-solving program RotaDyn. It provides brief instructional steps on how to use the program features.

USER MANUAL

CONTENTS

D1.1 INTRODUCTION	315
D1.2 UNTITLED WINDOW	316
D1.3 START PANEL	318
D1.3:1 QUESTION WINDOW	319
D1.3:2 START	321
D1.4 WRITE FORCE EQUATIONS	321
D1.5 WRITE ENERGY EXPRESSIONS	326
D1.6 ENVISION THE MOTION	329
D1.7 SOLVE THE QUESTION	330
D1.8 PERFORM SIMULATION	334
D1.9 ANSWER CHECK	336

D1.1 INTRODUCTION



Figure D1.1: The program icon.

Welcome to the RotaDyn problem-solving program. It is included on a floppy disk and runs on Macintosh computers using System 7 or later versions. The diagram above shows the icon which represents the program. To use the program load it onto the hard disk of the computer. To activate the program, place the mouse pointer on the icon and double click on it.

The program works on selections made by clicking the mouse pointer on buttons, menus and in windows as described below.

Buttons: A button is activated by clicking and releasing the mouse pointer on it.

Menus: An item in a menu is selected by clicking the mouse on the menu or on the triangle for the case of a pop-up menu and keeping the mouse pressed down. Releasing the mouse over the item in the menu selects the item.

Windows: A window can be activated or selected by clicking and releasing the mouse pointer anywhere in the observable region of the window or by selecting it from the menu.

When the application is loaded (activated), a window titled *Untitled Worksheet 1* first appears on the screen (see figure D1.2) followed by the window titled *Start Panel* shown in figure D1.6. A window is selected if its title bar is not greyed.

The functions of the *Untitled Worksheet* and the associated menus are first explained. Then follows the explanation on each window of the program commencing with the *Start Panel*.

D1.2: UNTITLED WORKSHEET

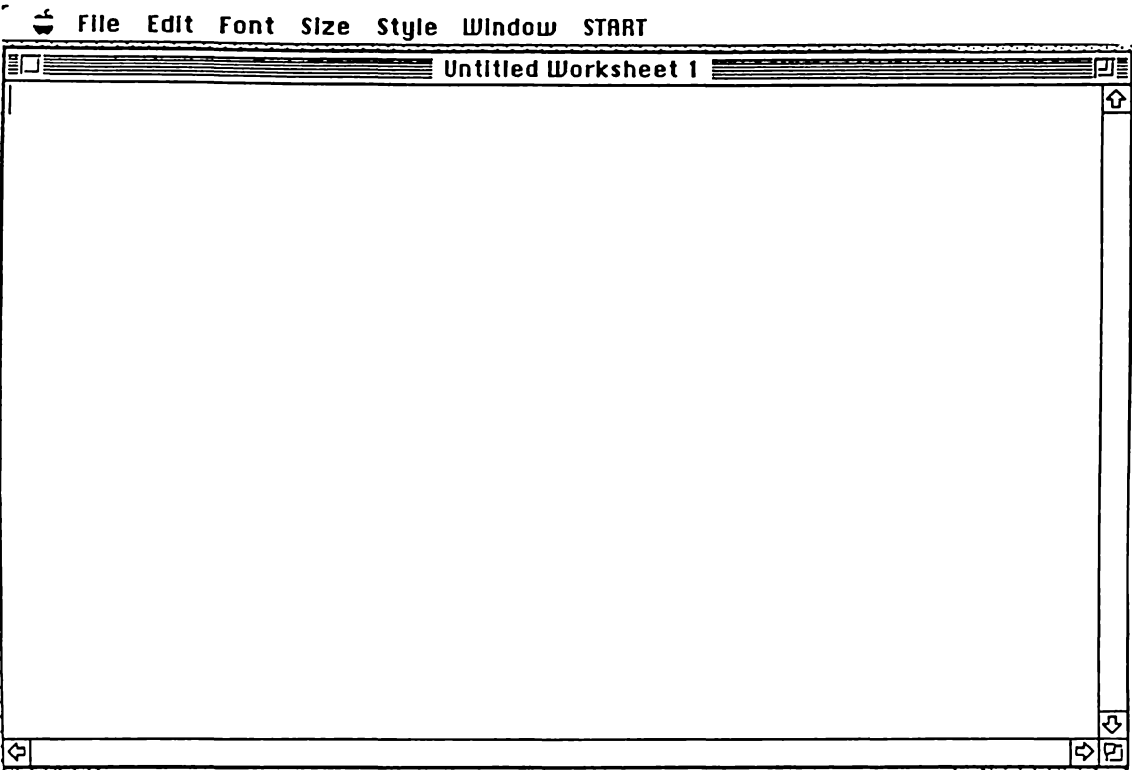


Figure D1.2: Untitled worksheet 1.

The untitled worksheet can be used to produce a printable copy of the solution to the problem. The editing functions in the *Edit* menu can be used to transfer working from other parts of the program.

Apple icon

About RotaDyn

Alarm Clock

Calculator

Chooser

Control Panels

Key Caps

Note Pad

Puzzle

Scrapbook

File

New ⌘N

Open... ⌘O

Close ⌘W

Save ⌘S

Save As...

Page Setup...

Print... ⌘P

Quit ⌘Q

Edit

Undo ⌘Z

Cut ⌘H

Copy ⌘C

Paste ⌘V

Clear

Show Clipboard

Figure D1.3: The items in the Apple, File and Edit menu.

When the *Untitled Worksheet* is the active window the *Apple*, *File*, *Edit*, *Font*, *Size*, *Style*, *Window* and *Start* menus are available. The **Apple** menu (see figure D1.3) allows access to other applications. The *About RotaDyn* item in the *Apple* menu opens a window and shows some preliminary simulations to depict the motion of the ball for the loop-the-loop track. The other items in the *Apple* menu represent other applications which may be available.

The functions performed by the **File** menu (see figure D1.3) items are given below.

New: Opens a new untitled worksheet.

Open: Opens any worksheet which has been previously saved.

Close: Closes an active worksheet.

Save: Saves the contents of the active worksheet. If the worksheet has not been labelled the user is prompted to supply a name by which it should be saved.

Save As: Saves the active worksheet by the name supplied by the user.

Page Set-up: Allows the user to make choices for the printing format.

Print: Prints the active worksheet.

Quit: Closes all the windows and ends the interaction.

The functions performed by the **Edit** menu (see figure D1.3) items are given below.

Undo: It reverses editing performed.

Cut: Removes the selected text from the worksheet and saves a copy of it in the clipboard.

Copy: Makes a copy of the text selected in the worksheet.

Paste: Copies the previously cut or copied text into the worksheet at the current cursor position.

Clear: Removes the selected text in the worksheet.

Show Clipboard: Opens the clipboard for viewing the items in it.

The **Font**, **Style** and **Size** menus allow the user to choose different font types and sizes for the text in the worksheet. The **Style** menu allows the appearance of the text to be edited (see the Macintosh manuals for more information on these).

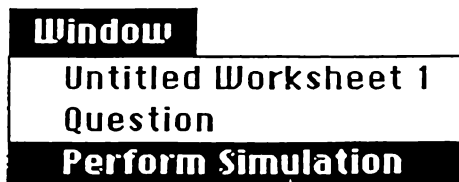


Figure D1.4: The items in the Window menu.

The **Window** menu includes names of the currently open windows. Figure D1.4 indicates the *Untitled Worksheet 1*, *Question* and *Perform Simulation* as the open windows.



Figure D1.5: The Start menu when the start panel is closed, open and currently active and open but in the background.

The *Start* menu allows the *Start Panel* to be opened or selected. If the *Start Panel* window is closed during interaction using *End Session* (see later section on Start Panel) then the *Start* menu item reads *Start Session* (see figure D1.5). If the *Start Panel* is open then it reads *Start Panel*. The item appears greyed if the *Start Panel* is active.

Next it is shown how the *Start Panel* can be used to start the interaction and use the program to commence solving the problem.

D1.3: THE START PANEL

The *Start Panel* window is shown in figure D1.6 below. This appears on the screen when the program is started. The menu functions have been explained in section D1.2 on the untitled worksheet. The window has 3 buttons labelled *Question*, *Start* and *End Session*. The buttons can be activated to perform specific functions described below by clicking in them once with the mouse pointer. The *Start* button is in bold. This indicates that it is the default button and can be activated by pressing the Return key on the keyboard.

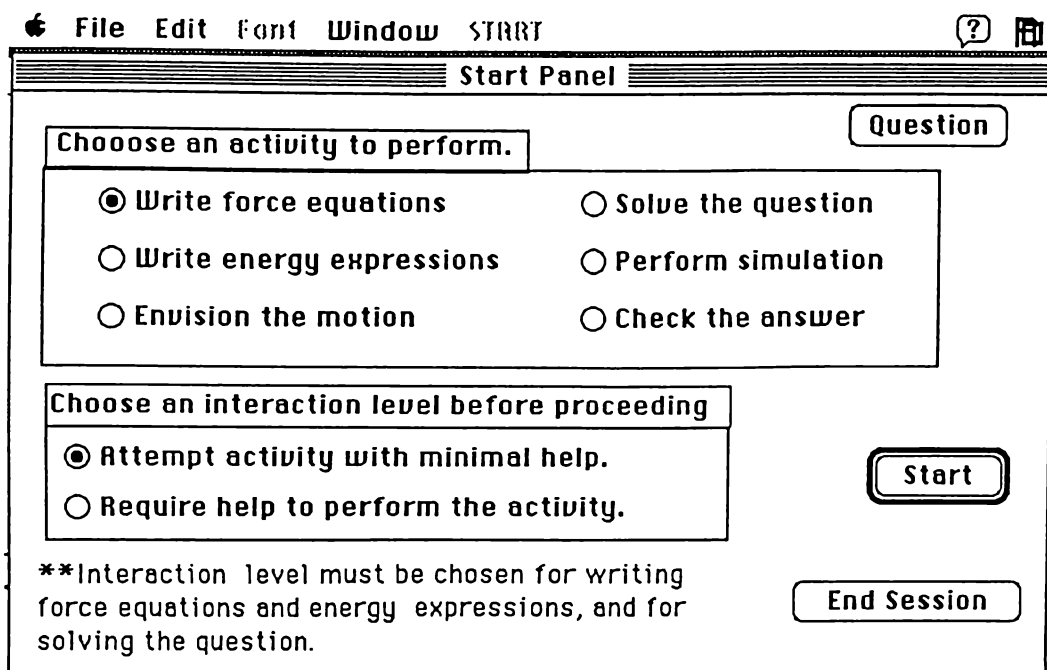


Figure D1.6: The Start Panel.

Question: Opens a window showing the question which needs to be solved. The title of the window is included in the *Window* menu when it is opened.

End Session: Closes all the windows except the worksheet. It ends the problem-solving interaction. Interaction can be restarted by selecting *Start Session* from the *Start* menu.

Start: This commences interaction by opening a window dependent on the radio buttons selected. Figure D1.6 shows the *Write Force Equations* and *Attempt activity with minimal help* as the selected radio buttons.

The *Question* and the *Start* functions are further explained.

D1.3:2 Question window

The *Question* window (see figure D1.7) can only be opened from the *Start Panel*. If it is open it can be brought to the front by selecting it from the window menu or by clicking in its visible region.

Question

A solid uniform ball of radius r , slides down a ramp into a loop-the-loop track. The ball starts from rest at a height h , above the lowest position of its centre of mass (as shown in the figure).

- (a). If the ball is not to leave the track at the top of the loop, what is the least value h can have in terms of R ?
- (b). If the ball were to roll without sliding, what is the minimum value for h ?

Note: R is the radius of the path of the centre of mass of the ball.

Explanation of: ☐

Close

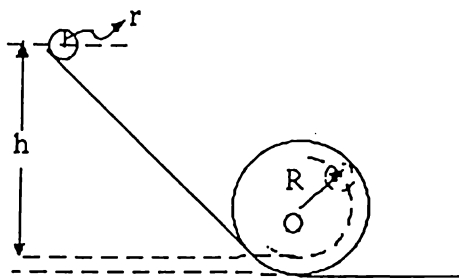
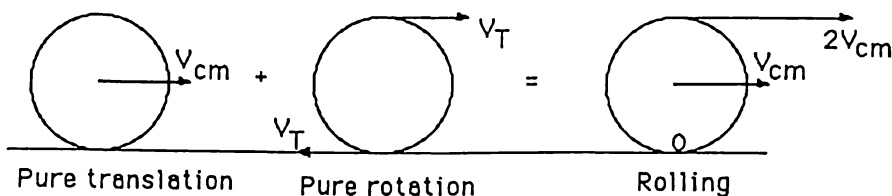


Figure D1.7: The Question window.

Explanation of:

✓ centre of mass
rolling vs sliding
slipping
uniform ball



When the ball moves it can rotate as well as translate. Except for the centre, each point on the ball has a velocity due to rotation V_T and that due to translation V_{cm} .

For sliding: $V_T = 0$

For rolling: $V_{cm} = V_T$

Friction must be present for the ball to roll.

OK

Figure D1.8: The pop-up menu for explanation on items and a dialogue box providing the explanation.

The *Question* window includes the question and a diagram. It can be closed by clicking the mouse pointer on its *Close* button. Explanations of terms can be obtained by clicking the mouse pointer on the triangle beside the words *Explanation of* and selecting from the pop-up menu (see figure D1.8). Moving the mouse pointer and releasing it over a menu item selects it. Information on a selected item is presented in a dialogue box (see figure D1.8).

D1.3:2 Start

Clicking the mouse pointer on the *Start* button commences the interaction. For the *Write force equations*, *Write energy expressions* and *Solve the question* activities one of the interaction levels radio buttons have to be chosen as well. The two choices for interaction levels are:

Attempt activity with minimal help; and

Require help to perform the activity.

For the other 3 activities the interaction level selection does not apply.

It should be noted that the program intentionally does only partial checking of information to make the users responsible for their learning. Therefore, during a check information in dialogue boxes has to be carefully compared with that being checked.

Each of the 6 activities together with the interaction levels if applicable are explained in the rest of the manual.

D1.4: WRITE FORCE EQUATIONS

The force equations windows can be used to obtain information on the forces acting on the ball and to practice writing force equations for various positions of the ball.

The windows for writing force equations (see figure D1.9) can be opened by selecting *Write force equations* and one of the interaction levels. *Attempt with minimal help* opens the *Force Equations* window and *Require help to perform the activity* opens the *Check Force Equations* window. To determine how to commence the interaction in each of the windows select *commencing session* from the *Need help with* pop-up menu.


 File Edit Font Window START Window Information Force Diagrams

Force Equations

☒ Sliding ☐ Rolling

Need help with: 

At a point on the incline 

Worksheet

a F g M N r R ø β sin cos f	+ - * ÷ = √ Square Return	Velocity  Angular velocity  Number 	CHECK DONE
--	--	---	---


 File Edit Window START Window Information

Check Force Equations

☒ Sliding
☐ Rolling

Position of ball 

Need help with: 

Direction
☒ Perpendicular to track
☐ Along or tangent to track

Check window

CHECK

Worksheet

a F g M N r R β ø sin cos f	+ - * ÷ = Square Return	Velocity  Angular velocity  Number 	Force Diagram DONE
---	---	---	---

Figure D1.9: The windows for writing force equations.

COMMON FEATURES OF THE TWO WINDOWS:

Sliding or Rolling: To check an equation one of these need to be selected.

Position of the ball: The position of the ball at which the equation of the ball is being written can be selected using the pop-up menu. In the *Force Equations* window the default selection is *At a point on the incline*. In the *Check Force Equations* window the menu shows *Position of the ball* to indicating that a selection has to be made.

Need help with: Clicking the mouse on the triangle beside it pops-up a menu which allows the user to select:

commencing session: This provides information on how to start and write the force equations.

relations: This provides information on relations between physical quantities for writing the force equations.

Worksheet: This slot can be used to write the force equations and to perform general working.

Keyboard: Each window has a keyboard of required symbols to help write the force equations. The symbols which are common to those supplied by the external keyboard of the computer can also be used to write the equation (e.g. F, g, I, etc). The velocity and angular velocity symbols and numbers can be accessed by using the pop-up menus.

Square: The square of a symbol can be obtained by selecting the symbol and clicking the mouse pointer on the *Square* button. Clicking on the *Square* button without selecting any symbol squares the symbol just before the cursor position.

DONE: Clicking the mouse pointer on this button closes the window and ends the respective interaction. The window can be reopened through the *Start Panel*.

Window Information menu: Information on an item in the window can be obtained by selecting it from the menu (Figure D1.10). This opens a dialogue box which provides information on the item.

Force Diagrams menu: The force diagrams for 9 positions of the ball can be accessed.

CHECK: This button provides a check of the force equation. Each window provides a different check. See the respective explanations below.

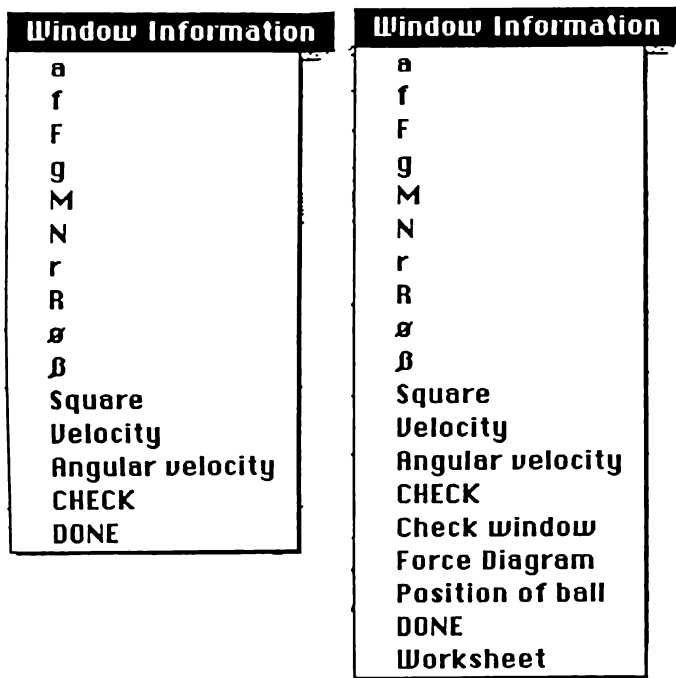


Figure D1.10: The Window Information menus for the Force equations and Check Force Equations windows.

Next the workings of each of the force windows are explained.

Force Equations:

In this window clicking the mouse pointer on the CHECK button provides the final force expression for the selected position of the ball. The steps needed to perform the check are:

- 1. Select sliding or rolling.
- 2. Select a position of the ball.
- 3. Click and release the mouse pointer on the CHECK button.

The Force diagrams for 9 different positions of the ball may be observed by selecting the relevant item from the *Force Diagrams* menu.

Check Force Equations

In this window clicking the mouse pointer on the CHECK button commences the interactive force equation check for the selected direction and position of the ball. The steps needed to perform the check are:

- 1. Select sliding or rolling.

2. Select a position of the ball.
3. Select a direction for which the force equation is being written.
4. Type the equation which needs to be checked into the *Check window*.
5. Click and release the mouse pointer on the CHECK button. This provides a dialogue box providing further information to help write the equation.
6. To carry on the interaction, correct the equation and click the CHECK button.
7. Keep repeating step 6, until the final form of the equation is obtained.
8. To write the equation for a different setting of 1, 2, or 3 repeat the steps.

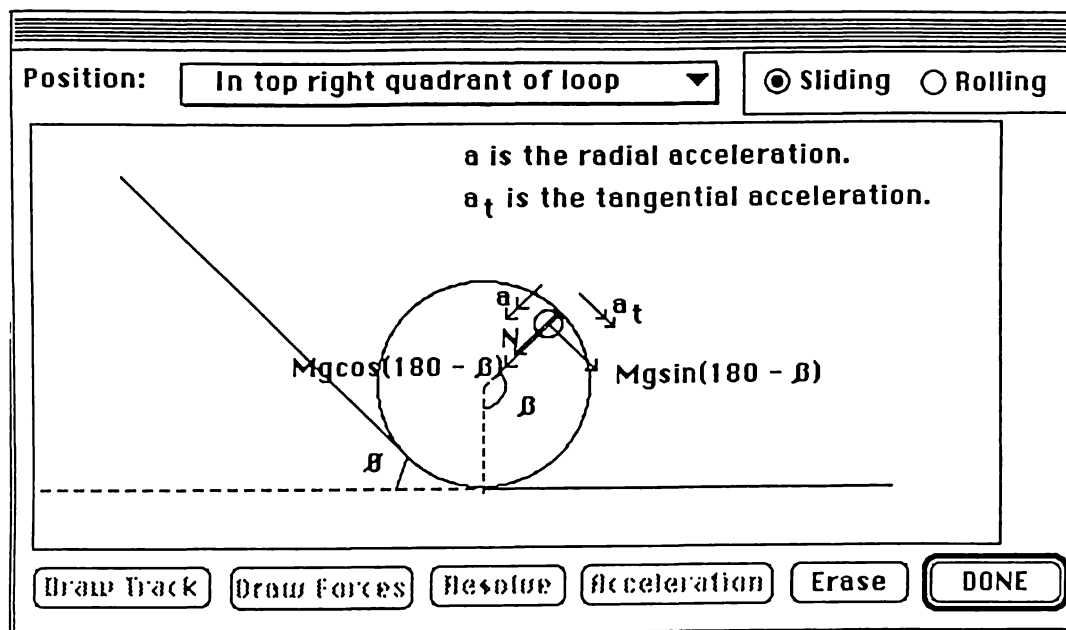


Figure D1.11: The Force Diagram window.

The force diagrams for 9 different positions of the ball may be observed by clicking and releasing the mouse pointer on the *Force Diagrams* button (see figure D1.9). This opens the window given in figure D1.11. Diagrams of resolved forces and accelerations can also be obtained as shown in the figure. The functions of the items in the window are explained below.

Position: This enables the choice of position of the ball at which the forces need to be observed. The pop-up menu allows 9 choices.

Draw Track: Draws the loop-the-loop track on which the ball moves.

Draw Forces: Draws the forces acting on the ball at the selected position of its motion.

Resolve: Draws the resolved forces acting on the ball at the selected position of its motion.

Acceleration: Draws the accelerations in the normal and the radial directions.

Erase: This clears the window for further drawing and activates the relevant buttons.

DONE: Closes the window and ends the force drawing session.

D1.5: WRITE ENERGY EXPRESSIONS

The energy expressions windows can be used to practice writing energy expressions at a single position and for equating the expressions for two points to obtain an energy equation.

The windows for writing energy expressions (see figure D1.12) can be opened by selecting *Write energy expressions* and one of the interaction levels. *Attempt with minimal help* opens the *Energy Expressions* window and *Require help to perform the activity* opens the *Check Energy Expressions* window. To determine how to commence the interaction in each of the windows select *commencing session* from the *Need help with* pop-up menu.

COMMON FEATURES OF THE TWO WINDOWS:

Sliding or Rolling: To check an expression one of these need to be selected.

Position of the ball: The position of the ball at which the energy expression is being written can be selected using the pop-up menu. In the *Energy Expressions* window there is only one position selection indicated by the *Start* default selection. In the *Check Energy Expressions* window there are two possible positions which can be selected. These are indicated by Position 1 and Position 2. See the explanation on each window later in the section for more information on these.

Need help with: Clicking the mouse on the triangle beside it pops-up a menu which allows the user to select:

commencing session: This provides information on how to start and write the force equations.

energy expressions: This provides information on the different energy expression.

Keyboard: Each window has a keyboard of required symbols to help write the energy expressions. See the explanation in section D1.4 for further information.

⚙ File Edit Font Window START Window Information

Energy Expressions

☒ Sliding ☐ Rolling

Need help with: ▼

Energy of the ball at the: start ▼

CHECK

Worksheet

g h l

M r R

y

= * ÷ +

- Square

Return

Velocity ▼

Angular velocity ▼

Number ▼

DONE

⚙ File Edit Font Window START Window Information

Check Energy Expressions

☒ Sliding ☐ Rolling

Need help with: ▼

Position 1 ▼

Position 2 ▼

Translational K.E.

Rotational K.E.

Gravitational P.E.

Total energy

Equate Totals

CHECK

g h l

M r R

y

= * ÷

+ - Square

Velocity ▼

Angular velocity ▼

Number ▼

DONE

Figure D1.12: The windows for writing energy expressions.

Square: The square of a symbol can be obtained by selecting the symbol and clicking the mouse pointer on the *Square* button. Clicking on the *Square* button without a selection squares the symbol just before the cursor.

DONE: This button closes the window and ends the respective interaction. The window can be reopened through the *Start Panel*.

CHECK: This button provides a check of the force equation. Each window provides a different check. See the respective explanations below.

Window Information menu: Information on an item in the window can be obtained by selecting it from the menu (Figure D1.13). This opens a dialogue box which provides information on the item.

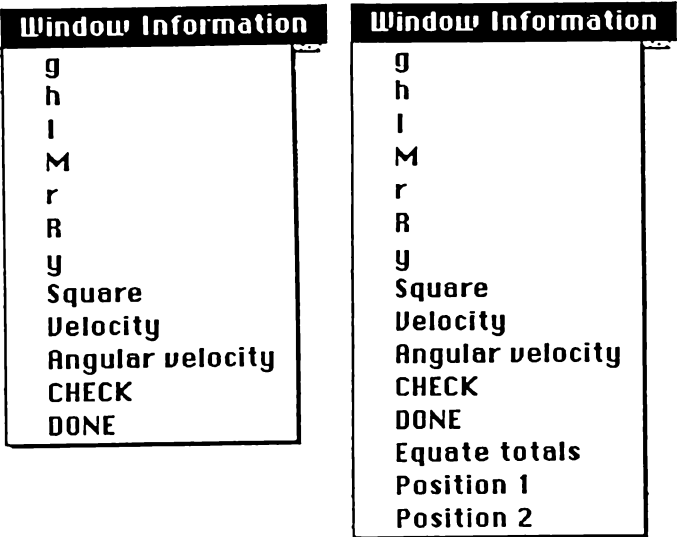


Figure D1.13: The Window Information menus for the Energy Expressions and Check Energy Expressions windows.

Next the workings of each window are explained.

Energy Expressions

In this window the worksheet can be used to perform general working and to write the energy expressions. Clicking the mouse pointer on the CHECK button provides the final energy expression. The steps needed to perform the check are:

1. Select sliding or rolling.
2. Select a position of the ball from the pop-up menu. A selection can be made for a single position or for equating the energy at two positions.
3. Click and release the mouse pointer on the CHECK button.

Check Energy Expressions

In this window 6 slots are provided to write the different energy expressions. Clicking the mouse pointer on the CHECK button commences the interactive energy expressions check. The steps needed to perform the check are:

1. Select sliding or rolling.
2. Select a position of the ball from Position 1 or Position 2 to check the energy expression at a single position of the ball. To equate the energy expressions at two positions, select a different position for both Position 1 and Position 2.
3. Type the energy expression for each of the different energies into the relevant slots.
4. Click the CHECK button. This provides a dialogue box providing further information to help write the expressions.
5. To carry on the interaction, correct the relevant expression and click the CHECK button.
6. Keep repeating step 5, until the final expression or equation is obtained.
7. To write the equation for a different setting of 1, 2, or 3 repeat all the steps.

If the expressions supplied for all the 6 slots are correct, the program sums the energy for each position and places the resultant expression in the *Total energy slot*. This provides an indication that the energy expressions are correct. If the expressions for both the positions are checked correct then clicking the *Equate Totals* button equates the energies for the two positions and places the simplified expression in the slot beside it.

D1.6: ENVISION THE MOTION

Selecting *Envision the motion* from in the *Start Panel* and clicking the *Start* button opens the **Envision** window. Figure D1.14 below shows an example. There is only one interaction level for envisioning the motion of the ball. Therefore, it does not matter which interaction level is selected in the Start Panel. The envisioning interaction can be performed as follows.

1. Select sliding or rolling. Sliding is the default selection.
2. Select one from the 6 possible choices namely overview (this is the default selection), down the incline, moving up in the loop, moving down in the loop, along the horizontal and insufficient starting height.
3. Click the *Display* button. An explanation of the type shown in figure D1.14 is presented in the slot below the Display button.
4. To envision for different choices repeat the steps. The previous text is erased and the new text appears each time the *Display* button is clicked.
5. Click the *DONE* button to end the interaction and close the window.

Note that envisioning can be enhanced with the aid of the simulation window. This is discussed in a later section.

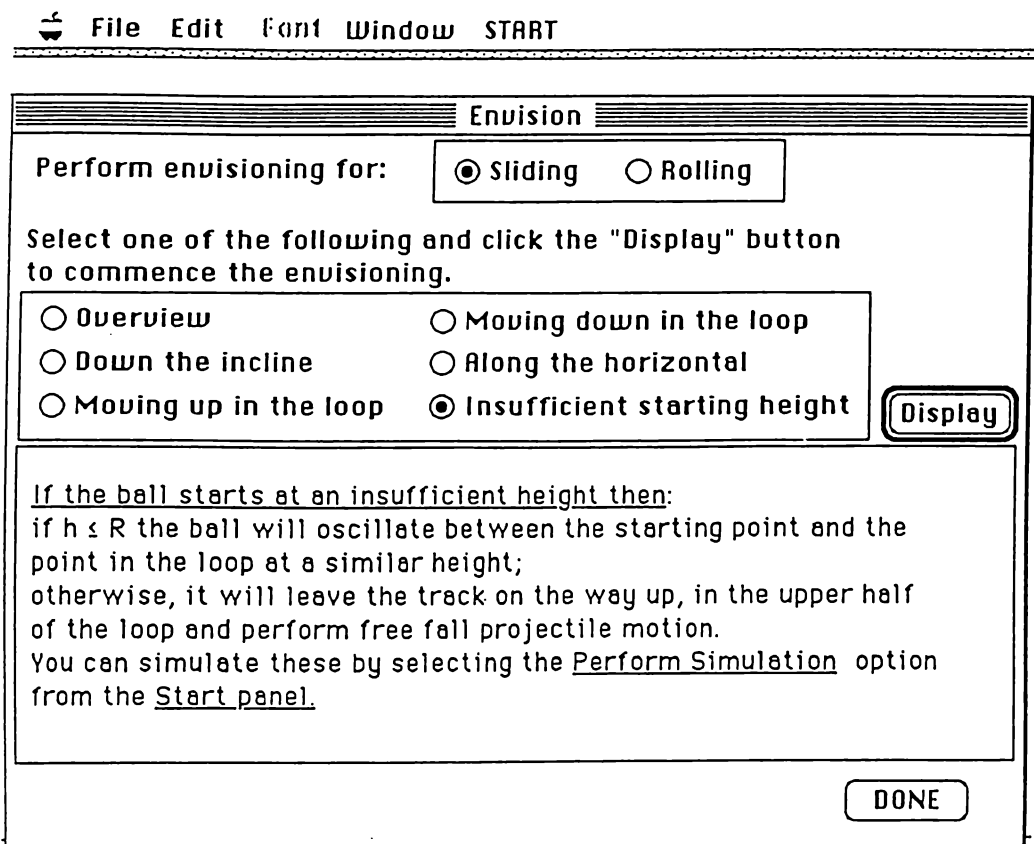


Figure D1.14: The Envision window.

D1.7 SOLVE THE QUESTION

Unlike the *Write force equations* and *Write energy expressions* windows which can be used to practice writing equations and expressions, the *Solving the question* windows are strictly for information directly related to the solution of the problem. If unsure on how to write the force equations or the energy expressions then practice writing these in the relevant windows described earlier before attempting to solve the problem.

The windows for solving the question (see figure D1.15) can be opened by selecting *Solve the question* and one of the interaction levels. *Attempt with minimal help* opens the *Perform Solution* window and *Require help to perform the activity* opens the *Perform Solution Check* window. To determine how to commence the interaction in

each of the windows select *commencing session* from the *Need help with* pop-up menu.

🍏

File

Edit

Font

Window

START

Window Information

Force Diagrams

Perform Solution

☒ Sliding

☐ Rolling

Need help to: ▼

Worksheet

aFg

hIM

NrR

yøβ

f

cos sin = *

÷ + - √

SquareReturn

Velocity ▼

Angular velocity ▼

Number ▼

CHECK

DONE

🍏

File

Edit

Font

Window

START

Window Information

Force Diagrams

Perform Solution Check

Position choice:

Need help with: ▼

Position 1 ▼

Position 2 ▼

Force equation:

Force Check

Energy expression:

Energy Clues

aFg

MNr

Rβø

cos sin sin

+ - *

÷ =

Square

Return

Velocity ▼

Angular velocity ▼

Number ▼

DONE

Figure D1.15: The windows for solving the question.

COMMON FEATURES OF THE TWO WINDOWS:

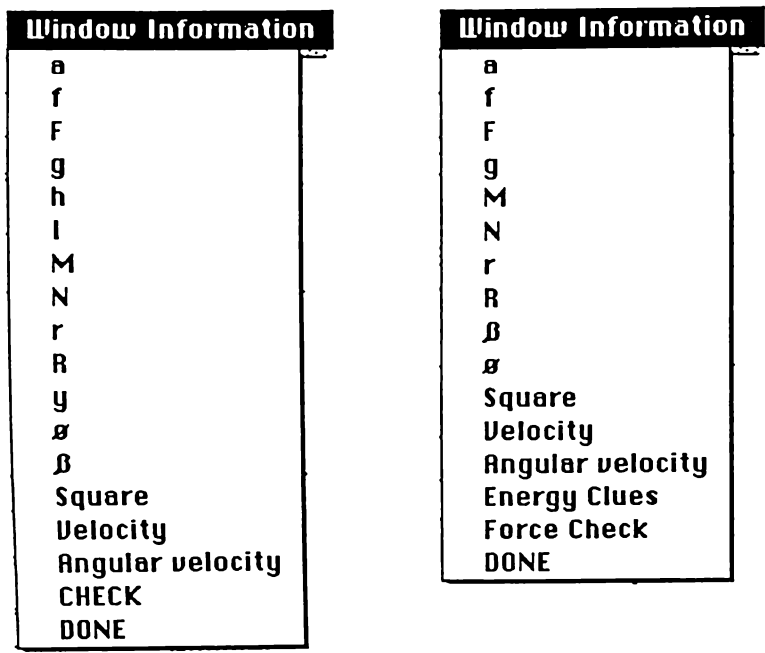


Figure D1.16: The Window Information menus for the Perform Solution and Perform Solution Check windows.

Window Information menu: Information on an item in the window can be obtained by selecting it from the menu (see figure D1.16). This opens a dialogue box which provides information on the item.

Need help with: Clicking the mouse on the triangle beside it pops-up a menu which allows the user to select:

- commencing session:** This provides information on how to start performing the solution.
- energy expressions:** This provides information on the different energy expressions.
- relations:** This provides information on relations between physical quantities for writing the force equations.

Keyboard: Each window has a keyboard of required symbols to help write the expressions. See the explanation in section D1.4 for further information. See also the explanation on Square in the same section.

DONE: This button closes the window and ends that interaction. The window can be reopened through the Start Panel.

Force Diagrams menu: The force diagrams for 9 positions of the ball can be obtained.

Next the workings of each window are explained.

Perform Solution

In this window the worksheet can be used to perform general working and to write the energy expressions and force equations. This window provides an environment similar to that of using pen and paper to perform the solution. The only help made available is the provision of the final expression. Clicking the CHECK button provides this final expression for the starting height of the ball. The steps needed to perform the check are:

1. Select sliding or rolling.
2. Click and release the mouse pointer on the CHECK button.

Perform solution check

In this window a check of the force equation for the top of the loop can be performed. However, the checking facility only performs a limited check. To perform a general check of the force equation use the *Check Force Equation* window. Clues on the positions at which to write the force equations and the energy expressions are also provided.

The steps needed to perform the force check are:

1. Select a position of the ball from Position 1.
2. Type the force equation into the slot labelled *Force equation*.
4. Click the CHECK button. This provides a dialogue box providing further information to help write and check the force equation.
5. To carry on the interaction, correct the equation and click the CHECK button.
6. Keep repeating step 5, until the final equation is obtained.

The steps needed to obtain the clues on writing the energy expressions are:

1. Click the Energy Clues button. This provides a dialogue box providing clues.
2. To obtain further clues perform the steps suggested and click the Energy Clues button again.
3. When ready to write the energy expression, go to one of the energy expression windows.

The difference between this window and the *Check Force Equations* or *Check Energy Expressions* is the provision of clues as to the position at which to write the equations or expressions.

D1.8: PERFORM SIMULATION

Opening this window enables the simulation of the ball to be performed. The motion of the ball can be observed for different starting heights of the ball, angles of incline of the track and values of the acceleration due to gravity as a percentage of the actual value. This window can also be used to obtain the minimum starting heights of the ball for sliding and rolling motion by trial and error using different starting values of h and R . The dependence of the motion of the ball on the incline of the track and the value of the gravitational intensity g can be studied as well.

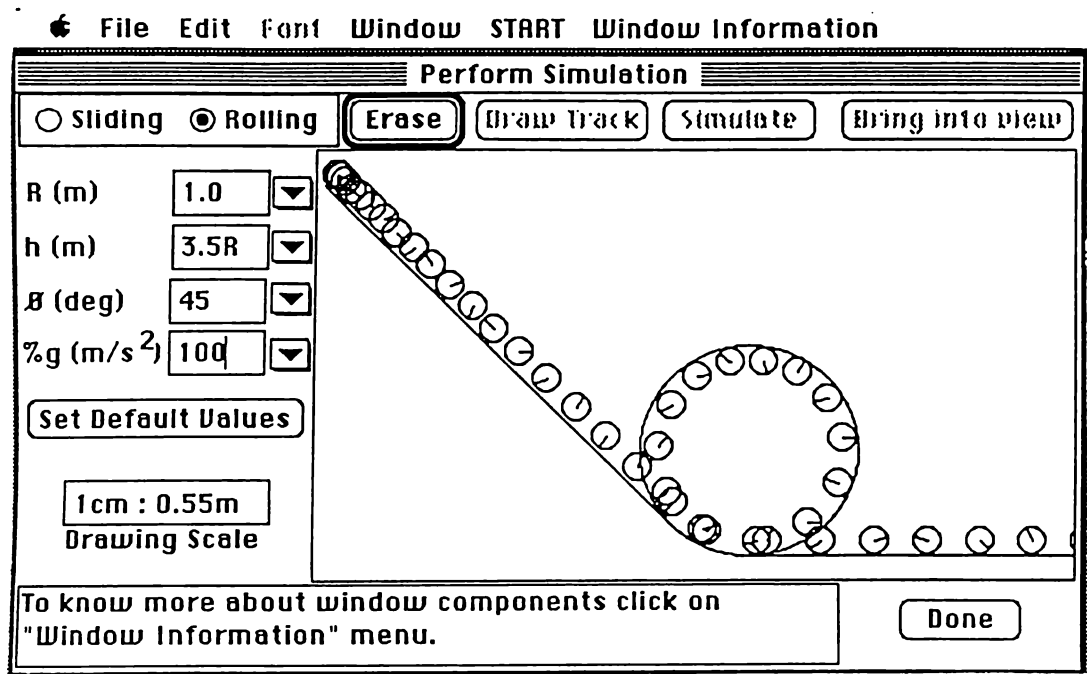


Figure D1.17: The simulation window.

The window can be opened by selecting *Perform simulation* from the *Start Panel* and clicking the Start button. The simulation does not have different interaction levels. Therefore, it does not matter which of the interaction levels are selected. Figure D1.17 shows an open simulation window with a simulation for rolling.

The different functions of the items in the window are first explained.

Values of R	Values of h	Incline angle	% g	Window Information
1.0	R	45	100	R
-----	-----	-----	-----	h
0.5	0.5R	10	50	Ø
1.5	1.5R	15	150	%g
2.0	2R	20	200	DONE
2.5	2.5R	25	250	Draw Track
3.0	3R	30	300	Drawing Scale
3.5	3.5R	35	350	Erase
4.0	4R	40	400	Set Default Values
4.5	4.5R	50	450	Simulate
5.0	5R	55	500	
-----	5.5R	60	-----	
Other	6R	65	Other	
	-----	70		
	Other	75		

		Other		

Figure D1.18: The pop-up menus for choosing values of R, h, Ø and g, and the Window Information menu.

Window Information menu: (Figure D1.18) Information on an item in the window can be obtained by selecting it from the menu. This opens a dialogue box which provides information on the item. The other menu bar items have been explained in the section on untitled worksheet.

Erase: Clears any drawing performed in the window.

Draw Track: Draws the loop-the-loop track using the values supplied for R, h and Ø.

Simulate: This performs the simulation using the values of R, h, Ø and g supplied and the choice of sliding or rolling.

Bring into view: This opens another window for drawing if the values of R, h and Ø are such that the track is drawn partially out of view.

Set Default Values: This resets the values of R, h, Ø and g to the respective default values of 1.0, 3.5R, 45 and 100.

Drawing Scale: This shows the scale used to depict the loop-the-loop track and the motion of the ball.

DONE: Closes the simulation window and ends this interaction. The window can be reopened from the *Start Panel*.

The values of R , h , θ and g can be set as follows:

1. Selecting from the pop-up menus by clicking on the triangle beside the relevant slot or typing into each slot using the keyboard. The pop-up menus are given in figure D1.18. The *Other* choice in the menus produces a dialogue box explaining that the values can also be typed into the slots.
2. The value for h can be supplied as a number or in terms of R .

D1.9: ANSWER CHECK

This window does not provide any interaction apart from producing a dialogue box showing the final expressions for the minimum starting heights of the ball to successfully perform the loop. This has been provided for users who wish to merely check the final expression.

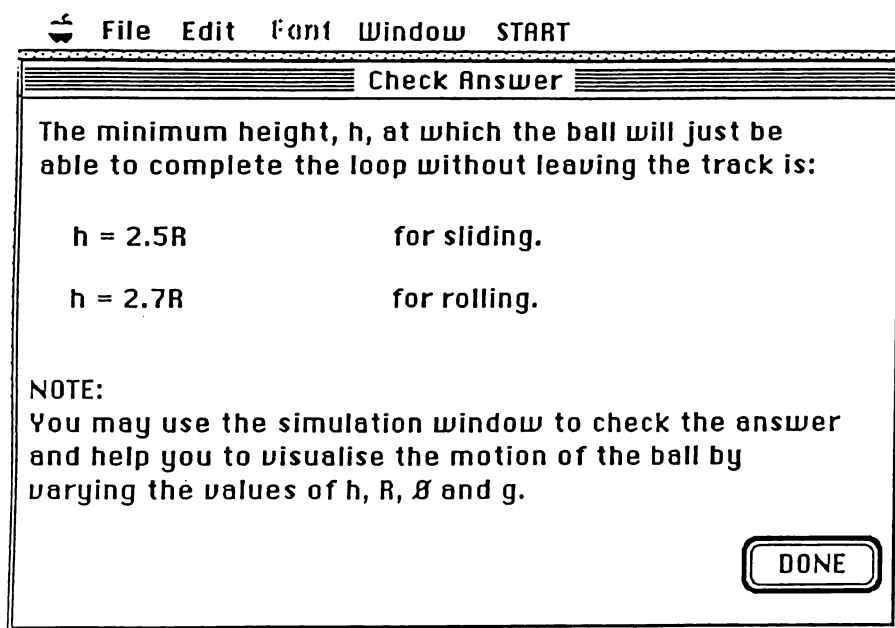


Figure D1.19: The answer check window.

The window can be opened by selecting *Answer Check* from the *Start Panel* and clicking the Start button. The answer check window does not have different interaction levels. Therefore, it does not matter which of the interaction levels are selected. Figure D1.19 shows an open answer check window.

APPENDIX D2

PROGRAM CODE

This appendix contains the code for the problem-solving program RotaDyn (included on disk in this thesis). The code has been compressed. Lines of code are separated by semi colons or by parenthesis. The code given in this appendix has to be used in conjunction with the other methods (functions) of the Think C Class Library.

Loop.c

(Main program for RotaDyn)

```
#include "CLoopApp.h"
extern CApplication *gApplication;
void main()
{
    CLoopApp      *loopApp;
    loopApp = new(CLoopApp);
    loopApp->ILoopApp();
    loopApp->Run();
    loopApp->Exit();
}
```

CLoopApp.c

(Application subclass for loop-the-loop).

```
#include: "Commands.h", "CBartender.h", "CLoopApp.h", "CLoopDoc.h", "Global.h",
"AppCommands.h", "CWindow.h", <CPane.h>, <SANE.h>, <CDialog.h>, <CDesktop.h>, <stdio.h>,
"CEquations.h", "CStartDirPanel.h", "CEnergyCheck.h", "CDLogDirSub.h", "CSolution.h",
"CForcesCheck.h", "CEnergyEquations.h", "CForces.h", "CForcesSimulate.h", "CSimPane.h",
"CForcesEqCheck.h", "CForcesPane.h", "CStdPopupPane.h", "CArrowPopupPane.h",
"CCheckBox.h", "CIconPane.h", "CIntegerText.h", "CEditText.h", <CPicture.h>

extern OSType gSignature; extern CBartender *gBartender; extern CDesktop *gDesktop; extern
CApplication *gApplication; extern CBureaucrat *gGopher; extern CStartDirPanel *theStartWind;
extern CWindow **otherWindDial; extern FILE *protFile;
```

```
#define      kExtraMasters      10
#define      kRainyDayFund      45000
#define      kCriticalBalance    40000
#define      kToolboxBalance     20000
```

```
void CLoopApp::ILoopApp(void)
{ CApplication::IApplication( kExtraMasters, kRainyDayFund, kCriticalBalance,
kToolboxBalance);}
```

```
void CLoopApp::SetUpFileParameters(void)
{      inherited::SetUpFileParameters();
    sfNumTypes = -1;
    gSignature = '????';      }
```

```
void CLoopApp::SetUpMenus(void)
{      inherited::SetUpMenus();
    AddResMenu(GetMHandle(MENUfont), 'FONT');
    gBartender->SetDimOption(MENUfont, dimNONE);
    gBartender->SetDimOption(MENUsize, dimNONE);
    gBartender->SetDimOption(MENUstyle, dimNONE);
    gBartender->SetDimOption(MENUwindow, dimNONE);
    gBartender->SetDimOption(MENUstart, dimNONE);
    gBartender->SetUnchecking(MENUfont, TRUE);
    gBartender->SetUnchecking(MENUsize, TRUE);
    gBartender->SetUnchecking(MENUstyle, TRUE);      }
```

```
void CLoopApp::CreateDocument()
{ CLoopDoc      *theDocument = NULL;
```

```
TRY
{      theDocument = new(CLoopDoc);
    theDocument->ILoopDoc(this, TRUE);
    theDocument->NewFile();      }      CATCH
{      ForgetObject( theDocument);      }      ENDTRY;      }
```

```
void CLoopApp::OpenDocument(SFReply *macSFReply)
{ CLoopDoc      *theDocument = NULL;
```

```

        TRY    {
            theDocument = new(CLoopDoc);
            theDocument->ILoopDoc(this, TRUE);
            theDocument->OpenFile(macSFReply);    }    CATCH
            {    ForgetObject( theDocument);    }    ENDTRY;    }

void CLoopApp::DoCommand(long theCommand)
{ switch (theCommand) {
    default:
        inherited::DoCommand(theCommand);
        break;
} }

void CLoopApp::ForceClassReferences( void)
{ Boolean alwaysFalse = FALSE;
  CObject *dummy = NULL;

if (alwaysFalse == TRUE)
    {member( dummy, CEquations); member( dummy, CStartDirPanel); member( dummy,
      CEnergyCheck); member( dummy, CDLogDirSub); member( dummy, CSolution); member(
      dummy, CForcesCheck); member( dummy, CEnergyEquations); member( dummy,
      CForces); member( dummy, CForcesSimulate); member( dummy, CSimPane); member(
      dummy, CForcesEqCheck); member( dummy, CForcesPane); member( dummy, CDialog);
      member( dummy, CRadioGroupPane); member( dummy, CForcesEquations); member(
      dummy, CView); member( dummy, CButton); member( dummy, CCheckBox); member(
      dummy, CRadioControl); member( dummy, CIconPane); member( dummy, CIntegerText);
      member( dummy, CStdPopupPane); member( dummy, CArrowPopupPane); member(
      dummy, CEditText); member( dummy, CPicture);    }
}

```

CLoopDoc.c

(Document method for loop-the-loop)

```

#include <Global.h>, <Commands.h>, <CAApplication.h>, <CBartender.h>, <CDataFile.h>,
<CDecorator.h>, <CDesktop.h>, <CView.h>, <CError.h>, <CPanorama.h>, <CScrollPane.h>,
<TBUtilities.h>, "CLoopDoc.h", "CDrawPane.h", "CWindow.h", "AppCommands.h",
"CStartDirPanel.h", <CDesktop.h>, <CDialog.h>, "CForcesSimulate.h", "CForces.h",
"CEnergyEquations.h", "CEnergyCheck.h", "CForcesCheck.h", <CRadioGroupPane.h>,
"CEquations.h", "CSolution.h", "CDLogDirSub.h", <stdio.h>

extern CAApplication *gApplication; extern CBartender *gBartender; extern CDecorator *gDecorator;
extern CDesktop *gDesktop; extern CBureaucrat *gGopher; extern OSType gSignature; extern
CError *gError; CStartDirPanel *theStartWind;

extern CStartDirPanel *theQuestion, *theEnvision, *theAnswerCheck; extern CForcesSimulate
*theDrawResolve; extern CForces *theForcesEq; extern CEnergyEquations *theEnergyEq; extern
CEnergyCheck *theEnergyCheck; extern CForcesCheck *theForcesCheck; extern CSolution
*theSolution, *theSolutionCheck; extern CDLogDirSub *theSimulation; CWindow
**otherWindDial; FILE *protFile; int workSheetNum, windowItemNum, numTextWindOpen;

static void NoteUsingMenu(long theCommand, Str255 itemTitle);
static void NoteUsingButtons(Str255 theTitle);

void CLoopDoc::ILoopDoc(CAApplication *aSupervisor, Boolean printable)
{ CDocument::IDocument(aSupervisor, printable); }

void CLoopDoc::NewFile(void)
{ Str255 theTitle;

BuildWindow(NULL);

numTextWindOpen += 1;
workSheetNum += 1;
switch (workSheetNum) {
case 1:
    *otherWindDial = new (CWindow); windowItemNum++; itsWindow->SetTitle("\pUntitled
Worksheet 1"); itsWindow->SetID(workSheetNum); break;

```

```

case 2: itsWindow->SetTitle("\pUntitled Worksheet 2"); itsWindow->SetID(workSheetNum); break;
case 3: itsWindow->SetTitle("\pUntitled Worksheet 3"); itsWindow->SetID(workSheetNum); break;
case 4: itsWindow->SetTitle("\pUntitled Worksheet 4"); itsWindow->SetID(workSheetNum); break;
case 5: itsWindow->SetTitle("\pUntitled Worksheet 5"); itsWindow->SetID(workSheetNum); break;
case 6: itsWindow->SetTitle("\pUntitled Worksheet 6"); itsWindow->SetID(workSheetNum); break;
case 7: itsWindow->SetTitle("\pUntitled Worksheet 7"); itsWindow->SetID(workSheetNum); break;
case 8: itsWindow->SetTitle("\pUntitled Worksheet 8"); itsWindow->SetID(workSheetNum); break;
case 9: itsWindow->SetTitle("\pUntitled Worksheet 9"); itsWindow->SetID(workSheetNum); break;
case 10: itsWindow->SetTitle("\pUntitled Worksheet 10"); itsWindow->SetID(workSheetNum);
break;
default:
itsWindow->SetTitle("\pUntitled Worksheet"); break;
}
itsWindow->GetTitle(theTitle);
*(otherWindDial + windowItemNum) = itsWindow;
InsMenuItem(gBartender->FindMacMenu(MENUWindow), theTitle, windowItemNum);
windowItemNum++;
itsWindow->Select();
if (theStartWind == NULL) {
    protFile = fopen("Protocol", "a+");
    fprintf(protFile, "Start paneN");
    fclose(protFile);
    theStartWind = new(CStartDirPanel);
    theStartWind->IStartDirPanel( StartPanel, this);
    theStartWind->BegDialDefBut(cmdStartPanel, cmdStartSession);
    (*(theStartWind).itsWindow).ID = cmdEndSession;
    TakingOutMenus();
    gBartender->RemoveMenuCmd(cmdStartPanel);
    gBartender->InsertMenuCmd(cmdStartPanel, "\pStart Panel", MENUStart, 0);
    gBartender->DisableMenu( MENUStart); }
}

void CLoopDoc::OpenFile(SFReply *macSFReply)
{
    CDataFile *theFile; Handle theData; Str63 theName; OSErr theError;
    theFile = new(CDataFile);
    itsFile = theFile;
    theFile->IDataFile();
    theFile->SFSpecify(macSFReply);
    theFile->Open(fsRdWrPerm);
    if (theFile->GetLength() > 10240L)
    { ParamText( "\pCan't open a file this big.", "\p", "\p", "\p"); PositionDialog('ALRT', 128);
    InitCursor(); Alert(128, NULL); Dispose(); return; }
    theData = theFile->ReadAll();
    BuildWindow(theData);
    DisposHandle(theData);
    itsFile->GetName(theName);
    InsMenuItem(gBartender->FindMacMenu(MENUWindow), theName, windowItemNum);
    windowItemNum++;
    itsWindow->Select(); }

void CLoopDoc::BuildWindow (Handle theData)
{
    CScrollPane *theScrollPane; CEditPane *theMainPane; Rect margin;
    itsWindow = new(CWindow);
    itsWindow->IWindow(WINDCulture, FALSE, gDesktop, this);
    theScrollPane = new(CScrollPane);
    theScrollPane->IScrollView(itsWindow, this, 10, 10, 0, 0, sizELASTIC, sizELASTIC, TRUE, TRUE,
    TRUE);
    theScrollPane->FitToEnclFrame(TRUE, TRUE);
    theMainPane = new(CEditPane);
    itsMainPane = theMainPane;
    itsGopher = theMainPane;
    theMainPane->IEditPane(theScrollPane, this);
    theMainPane->SetFontNumber(0);
    theScrollPane->InstallPanorama(theMainPane);
    if (theData)
        theMainPane->SetTextHandle(theData);
    gDecorator->PlaceNewWindow(itsWindow); }

Boolean CLoopDoc::DoSave(void)

```

```

{
    Handle theData;
    if (itsFile == NULL)
        return(DoSaveFileAs());
    else {
        theData = (*(CEditText *)itsMainPane)->macTE).hText;
        ((CDataFile *)itsFile)->WriteAll(theData);
        dirty = FALSE;
        gBartender->DisableCmd(cmdSave);
        return(TRUE);
    }
}

Boolean CLoopDoc::DoSaveAs(SFReply *macSFReply)
{
    Str255 theTitle;
    itsWindow->GetTitle(theTitle);
    if (itsFile != NULL)
        itsFile->Dispose();
    itsFile = new(CDataFile);
    ((CDataFile *)itsFile)->IDataFile();
    itsFile->SFSpecify(macSFReply);
    itsFile->CreateNew(gSignature, "TEXT");
    itsFile->Open(fsRdWrPerm);
    itsWindow->SetTitle(macSFReply->fName);
    DelMenuItem( gBartender->FindMacMenu(MENUWindow), gBartender-> FindItemText
(MENUWindow, theTitle));
    windowItemNum--;
    InsMenuItem(gBartender->FindMacMenu(MENUWindow), macSFReply->fName, windowItemNum);
    *(otherWindDial + windowItemNum) = itsWindow;
    windowItemNum++;
    return( DoSave() );
}

void CLoopDoc::CloseWind( CWindow *theWindow)
{
    Str255 theTitle; short z = 0;
    if (numTextWindOpen == 1) {
        SelWindQuit();
        gBartender->DisableMenu( MENUStart);
    }
    itsWindow->GetTitle(theTitle);
    theWindow = itsWindow;
    z = gBartender->FindItemText(MENUWindow, theTitle);
    DelMenuItem( gBartender->FindMacMenu(MENUWindow), z);
    windowItemNum--;
    for (z = z; z < windowItemNum; z++)
        *(otherWindDial + z) = *(otherWindDial + z + 1);
    *(otherWindDial + z) = NULL;
    ForgetPtr(*(otherWindDial + z));
    numTextWindOpen--;
    workSheetNum--;
    inherited::CloseWind( theWindow);
}

void CLoopDoc::DoCommand(long theCommand)
{
    CursHandle gWatchCursor; Str255 theTitle; CWindow *theWindow; CEquations
    *theEquations;
    if (theCommand < 0)
        if (HiShort(-theCommand) != MENUApple)
        {
            gWatchCursor = GetCursor(watchCursor);SetCursor(*gWatchCursor);theWindow =
            *(otherWindDial + LoShort(-theCommand));theWindow->Select();theWindow->GetTitle(theTitle);
            protFile = fopen("Protocol", "a+");fprintf(protFile, "%s\n", theTitle);fclose(protFile);
            InitCursor();
        }
    if (theCommand != cmdEndSession)
        NoteUsingMenu(theCommand, theTitle);
        switch (theCommand)
        {
            case cmdAbout: QuestionDialogClar(AboutWind); break;
            case cmdStartPanel: gWatchCursor = GetCursor(watchCursor); SetCursor(*gWatchCursor);
                if (theStartWind == NULL)
                {
                    theStartWind = new(CStartDirPanel);
                    theStartWind->IStartDirPanel( StartPanel, this);
                    theStartWind->BegDialDefBut(cmdStartPanel, cmdStartSession);
                    (*(theStartWind).itsWindow).ID = cmdEndSession;
                    TakingOutMenus();
                }

```

```

        gBartender->RemoveMenuCmd(cmdStartPanel);
        gBartender->InsertMenuCmd(cmdStartPanel, "\pStart Panel", MENUStart, 0);
        gBartender->DisableMenu( MENUStart);    }
else
{
    itsWindow = (*theStartWind).itsWindow;
    itsWindow->Select();
    gBartender->DisableMenu( MENUStart);    }
    InitCursor();
    break;
case cmdEndSession: protFile = fopen("Protocol", "a+"); fprintf(protFile, "End session\n");
fclose(protFile); SelWindQuit(); StartingMenus(); break;
case cmdQuit: gWatchCursor = GetCursor(watchCursor); SetCursor(*gWatchCursor);
SelWindQuit(); StartingMenus(); ForgetObject(theStartWind);
inherited::DoCommand(theCommand); InitCursor(); break;
case cmdClose:
    if (numTextWindOpen == 1)    {
        SelWindQuit();
        inherited::DoCommand(theCommand);
        return;    }
    numTextWindOpen--; inherited::DoCommand(theCommand); break;
case cmdEnEqCheckExp: QuestionDialogClar(EnEqCheckExp); break;

case cmdgSlidInfo: case cmdg: theEquations = new (CEquations); theEquations->
ModalDialPlace( "\pg is the gravitational intensity or the acceleration due to
gravity.", "\p", "\p", "\p", 2005); break;
case cmdFrictionBtn: theEquations = new (CEquations); theEquations->
ModalDialPlace("\pF is the frictional force between the track and the ball.", "\p", "\p",
"\p", 2005); break;
case cmdhSlidInfo: case cmdh: QuestionDialogClar(HeightInfo); break;
case cmdI: theEquations = new (CEquations); theEquations->ModalDialPlace( "\pI is the
moment of inertia of the ball about an axis through its centre.", "\p", "\p", "\p", 2005);
break;
case cmdM: theEquations = new (CEquations); theEquations->ModalDialPlace("\pM is the
mass of the ball.", "\p", "\p", "\p", 2005); break;
case cmdr: theEquations = new (CEquations); theEquations->ModalDialPlace("\pr is the
radius of the ball.", "\p", "\p", "\p", 2005); break;
case cmdRSlidInfo: case cmdR: QuestionDialogClar(RadiusInfo); break;
case cmdy: theEquations = new (CEquations); theEquations->ModalDialPlace( "\py is the
height of the ball above its lowest position. It is measured from the centre of the ball.",
"\p", "\p", "\p", 2005); break;
case cmdSquare: theEquations = new (CEquations); theEquations->ModalDialPlace(
"\pClicking this button squares the selected symbol, or the symbol before the cursor
position.", "\p", "\p", "\p", 2005); break;
case cmdVelocity: QuestionDialogClar(keyInfoVelocity); break;
case cmdAngVel: QuestionDialogClar(keyInfoAngVel); break;
case cmdCheck: theEquations = new (CEquations); theEquations->ModalDialPlace(
"\pClicking this button provides the final expression.", "\p", "\p", "\p", 2005); break;
case cmdQuitEnergy: theEquations = new (CEquations); theEquations->ModalDialPlace(
"\pClicking the DONE button closes the window.\nTo save working use the \"Edit Menu\"
to transfer the working to the main worksheet.", "\p", "\p", "\p", 2005); break;
case cmdEquateTotal: theEquations = new (CEquations); theEquations->ModalDialPlace(
"\pEquates the energy of the two points and gives a simplified form of the expression.",
"\p", "\p", "\p", 2005); break;
case cmdPos1: case cmdPos2: QuestionDialogClar(PositionOneTwo); break;
case HowStartCheckEnEq: QuestionDialogClar(StartCheckEnEqDial); break;
case cmdBetaForces: QuestionDialogClar(BetaForcesInfo); break;
case CheckForcesInfo: theEquations = new (CEquations); theEquations->ModalDialPlace(
"\pProvides a simplified force expression for the chosen position.", "\p", "\p", "\p", 2005);
break;
case cmdPhiSlidInfo: case PhiForcesInfo: QuestionDialogClar(AngleInfo); break;
case ReactionForcesInfo: theEquations = new (CEquations); theEquations->
ModalDialPlace( "\pN represents the normal reaction force of the track on the ball.",
"\p", "\p", "\p", 2005); break;
case ForceForcesInfo: theEquations = new (CEquations); theEquations->ModalDialPlace(
"\pF represents the resultant force acting on the ball in a chosen direction.", "\p", "\p", "\p",
2005); break;
case AccelForcesInfo: theEquations = new (CEquations); theEquations->ModalDialPlace(
"\pa represents the acceleration in the direction of the force F.", "\p", "\p", "\p", 2005);
break;

```



```

case cmdWorksheetInfo: theEquations = new (CEquations); theEquations->ModalDialPlace(
    "\pAny general working can be done here. You can use the \"Cut\" and \"Paste\" functions in
    the \"Edit\" menu to transfer force equations to the checking window for checking", "\p",
    "\p", "\p", 2005); break;
case cmdForceDiagChInfo: theEquations = new (CEquations); theEquations->
    ModalDialPlace( "\pOpens a dialog box for representing forces and acceleration for different
    positions of the ball. Forces may be resolved as well.", "\p", "\p", "\p", 2005); break;
case cmdFEqCheckInfo: theEquations = new (CEquations); theEquations->
    ModalDialPlace("\pChecks the force equation in the \"Check window\".", "\p", "\p", "\p",
    2005); break;
case cmdFEqWindCheck: theEquations = new (CEquations); theEquations->
    ModalDialPlace( "\pThe force equation needs to be placed here for checking.", "\p",
    "\p", "\p", 2005); break;
case cmdPosFEqInfo: theEquations = new (CEquations); theEquations->ModalDialPlace(
    "\pPops up a menu from which the position of the ball can be selected.", "\p", "\p", "\p",
    2005); break;
case cmdForceSolCheck: theEquations = new (CEquations); theEquations->
    ModalDialPlace( "\pEnables you to check the force equation for the ball, at the top
    of the loop.\r Information leading to the solution, is provided during the check.", "\p", "\p",
    "\p", 2005); break;
case cmdEnergySolCheck: theEquations = new (CEquations); theEquations->
    ModalDialPlace( "\pEnables you to check the energy expression for the ball, at the start and
    the top of the loop.\r It prompts the next step each time a step is successfully performed and
    provides information leading to the solution.", "\p", "\p", "\p", 2005); break;
case cmdHowStartFEq: QuestionDialogClar(HowStartFEquatCheck); break;
case cmdForceStart: QuestionDialogClar(ForcesStart); break;
case cmdForceIncline: QuestionDialogClar(ForcesIncline); break;
case cmdLowerLeftLoop: QuestionDialogClar(ForcesBottomLeft); break;
case cmdLowerRightLoop: QuestionDialogClar(ForcesBottomRight); break;
case cmdUpperRightLoop: QuestionDialogClar(ForcesTopRight); break;
case cmdUpperLeftLoop: QuestionDialogClar(ForcesTopLeft); break;
case cmdForceTopLoop: QuestionDialogClar(ForcesTop); break;
case cmdForceBottomLoop: QuestionDialogClar(ForcesBottom); break;
case cmdHorizontal: QuestionDialogClar(ForcesHorizontal); break;
case cmdBringViewSlidInfo: theEquations = new (CEquations); theEquations->
    ModalDialPlace( "\pOpens a larger window for drawing when the track drawn is partly out
    of view.", "\p", "\p", "\p", 2005); break;
case cmdCancelSlidInfo: theEquations = new (CEquations); theEquations->
    ModalDialPlace( "\pCloses the window and ends the simulation session.", "\p",
    "\p", "\p", 2005); break;
case cmdDrawSlidInfo: theEquations = new (CEquations); theEquations->ModalDialPlace(
    "\pDraws the track using the values shown for R, h, Ø and g.", "\p", "\p", "\p", 2005);
break;
case cmdScaleSlidInfo: theEquations = new (CEquations); theEquations->ModalDialPlace(
    "\pShows the scale used to draw the diagram.", "\p", "\p", "\p", 2005); break;
case cmdEraseSlidInfo: theEquations = new (CEquations); theEquations->
    ModalDialPlace("\pErases any drawing done.", "\p", "\p", "\p", 2005); break;
case cmdDefaultSlidInfo: theEquations = new (CEquations); theEquations->
    ModalDialPlace( "\pSets R, h, Ø and g to the respective default values of 1.0, 3.5R,
    45 and 100. ", "\p", "\p", "\p", 2005); break;
case cmdSimButSlidInfo: theEquations = new (CEquations); theEquations->
    ModalDialPlace( "\pSimulates the motion of the ball using the values supplied for
    R, h, Ø and g.", "\p", "\p", "\p", 2005); break;
default: inherited::DoCommand(theCommand); break;
}
}

```

```

void CLoopDoc::SelWindQuit(void)
{ CursHandle gWatchCursor; CWindow *theWindow; Str255 theTitle;

gWatchCursor = GetCursor(watchCursor);
SetCursor(*gWatchCursor);
if ( theQuestion != NULL ) {
    itsWindow = (*theQuestion).itsWindow; itsWindow->Select(); itsWindow->GetTitle(theTitle);
    NoteUsingButtons(theTitle); theQuestion->CloseDialogues(); theQuestion->
    ThrowAwayDialog(cmdCloseQuWind); theQuestion = NULL; ForgetObject(theQuestion);
}
if (theEnergyCheck != NULL) {
    itsWindow = (*theEnergyCheck).itsWindow; itsWindow->Select(); itsWindow->
    GetTitle(theTitle);
}

```

```

NoteUsingButtons(theTitle); theEnergyCheck->CloseDialEnergyCheck(); theEnergyCheck->
ThrowEnergyCheckDial(QuitCheckEnEq); gBartender->RemoveMenu(MENUEnergyEqCheck);
theEnergyCheck = NULL; ForgetObject(theEnergyCheck); }
if ( theEnergyEq != NULL) {
    itsWindow = (*theEnergyEq).itsWindow; itsWindow->Select(); itsWindow->GetTitle(theTitle);
    NoteUsingButtons(theTitle); gBartender->RemoveMenu(MENUEnergyEq); theEnergyEq->
    CloseDialEnergy(); theEnergyEq->ThrowEnergyDial(cmdQUITbtn); theEnergyEq = NULL;
    ForgetObject(theEnergyEq);
}
if ( theForcesEq != NULL) {
    itsWindow = (*theForcesEq).itsWindow; itsWindow->Select(); itsWindow->GetTitle(theTitle);
    NoteUsingButtons(theTitle); gBartender->RemoveMenu(MENUForcesEq); gBartender->
    RemoveMenu(MENUForceDiagrams); theForcesEq->CloseDialForces(); theForcesEq->
    ThrowForcesDial(cmdQuitForces); theForcesEq = NULL; ForgetObject(theForcesEq); }
if ( theDrawResolve != NULL) {
    itsWindow = (*theDrawResolve).itsWindow; itsWindow->Select(); itsWindow->GetTitle(theTitle);
    NoteUsingButtons(theTitle); theDrawResolve->ThrowAwayDial(cmdQUITForceDiag);
    theDrawResolve = NULL; ForgetObject(theDrawResolve); }
if ( theForcesCheck != NULL) {
    itsWindow = (*theForcesCheck).itsWindow; itsWindow->Select(); itsWindow->GetTitle(theTitle);
    NoteUsingButtons(theTitle); gBartender->RemoveMenu(MENUForcesCheck); theForcesCheck->
    CloseForceCheckDial(); theForcesCheck->ThrowForceCheckDial(cmdQuitForcesEq);
    theForcesCheck = NULL; ForgetObject(theForcesCheck); }
if ( theSolution != NULL) {
    itsWindow = (*theSolution).itsWindow; itsWindow->Select(); itsWindow->GetTitle(theTitle);
    NoteUsingButtons(theTitle); gBartender->RemoveMenu(MENUSolution); gBartender->
    RemoveMenu(MENUForceDiagrams); theSolution->CloseDialSolution(); theSolution->
    ThrowSolutionDial(cmdSolutionQuit); theSolution = NULL; ForgetObject(theSolution); }
if ( theSolutionCheck != NULL) {
    itsWindow = (*theSolutionCheck).itsWindow; itsWindow->Select(); itsWindow->
    GetTitle(theTitle);
    NoteUsingButtons(theTitle); gBartender->RemoveMenu(MENUSolCheck); gBartender->
    RemoveMenu(MENUForceDiagrams); theSolutionCheck->CloseDialSolution(); theSolutionCheck
    ->ThrowSolutionDial(cmdCheckSolQuit); theSolutionCheck = NULL;
    ForgetObject(theSolutionCheck);
}
if ( theSimulation != NULL) {
    itsWindow = (*theSimulation).itsWindow; itsWindow->Select(); itsWindow->GetTitle(theTitle);
    NoteUsingButtons(theTitle); gBartender->RemoveMenu(MENUSlidWindInfo); theSimulation->
    CloseSimulationDial(); theSimulation->ThrowSimulationDial(cmdDone); theSimulation = NULL;
    ForgetObject(theSimulation);
}
if ( theEnvision != NULL) {
    itsWindow = (*theEnvision).itsWindow; itsWindow->Select(); itsWindow->GetTitle(theTitle);
    NoteUsingButtons(theTitle); theEnvision->CloseDialogues(); theEnvision->
    ThrowAwayDialog(cmdEnvisionQuit); theEnvision = NULL; ForgetObject(theEnvision); }
if ( theAnswerCheck != NULL) {
    itsWindow = (*theAnswerCheck).itsWindow; itsWindow->Select(); itsWindow->
    GetTitle(theTitle);
    NoteUsingButtons(theTitle); theAnswerCheck->CloseDialogues(); theAnswerCheck->
    ThrowAwayDialog(cmdAnswerQuit); theAnswerCheck = NULL; ForgetObject(theAnswerCheck);
}
if ( theStartWind != NULL) {
    itsWindow = (*theStartWind).itsWindow; itsWindow->Select(); itsWindow->GetTitle(theTitle);
    NoteUsingButtons(theTitle); gBartender->RemoveMenuCmd(cmdStartPanel); gBartender->
    InsertMenuCmd(cmdStartPanel, "\pStart Session", MENUStart, 0); theStartWind->
    ThrowAwayDialog(cmdEndSession); theStartWind = NULL; }
theWindow = gDesktop->GetTopWindow();
theWindow->Select();
itsWindow = theWindow;
InitCursor();
}

void CLoopDoc::ChangingMenus(void)
{
    CWindow *theWindow; Str255 theTitle;
    theWindow = gDesktop->GetTopWindow();
    switch ((*theWindow).ID)
    {
        case 1: case 2: case 3: case 4: case 5: case 6: case 7: case 8: case 9: case 10: case 11: case 12:
            TakingOutMenus(); StartingMenus(); ((CEditText *)itsMainPane)-> BecomeGopher(TRUE);
            break;
        case cmdQuestion: TakingOutMenus(); gBartender->EnableMenu(MENUStart); break;
        case cmdEndSession: TakingOutMenus(); gBartender->DisableMenu( MENUStart); break;
    }
}

```

```

case EnergyEquation: TakingOutMenus(); gBartender->EnableMenu(MENUStart); gBartender->
AddMenu(MENUEnergyEq, TRUE, 0); gBartender->
SetDimOption(MENUEnergyEq, dimNONE); break;
case SlidEnEqCheck: TakingOutMenus(); gBartender->EnableMenu(MENUStart); gBartender->
AddMenu(MENUEnEqCheck, TRUE, 0); gBartender->
SetDimOption(MENUEnEqCheck, dimNONE); break;
case ForcesEquations: TakingOutMenus(); gBartender->EnableMenu(MENUStart); gBartender->
AddMenu(MENUForcesEq, TRUE, 0); gBartender->
SetDimOption(MENUForcesEq, dimNONE); gBartender->AddMenu(MENUForceDiagrams, TRUE,
0); gBartender->SetDimOption(MENUForceDiagrams, dimNONE); break;
case ForcesEquatCheck: TakingOutMenus(); gBartender->EnableMenu(MENUStart); gBartender->
AddMenu(MENUForcesCheck, TRUE, 0); gBartender->
SetDimOption(MENUForcesCheck, dimNONE); break;
case Solution: TakingOutMenus(); gBartender->EnableMenu(MENUStart); gBartender->
AddMenu(MENUSolution, TRUE, 0); gBartender->SetDimOption(MENUSolution, dimNONE);
gBartender->AddMenu(MENUForceDiagrams, TRUE, 0); gBartender->
SetDimOption(MENUForceDiagrams, dimNONE); break;
case SolutionCheck: TakingOutMenus(); gBartender->EnableMenu(MENUStart); gBartender
-> AddMenu(MENUSolCheck, TRUE, 0); gBartender->SetDimOption(MENUSolCheck,
dimNONE); gBartender->AddMenu(MENUForceDiagrams, TRUE, 0); gBartender->
SetDimOption(MENUForceDiagrams, dimNONE); break;
case DialogWindow: TakingOutMenus(); gBartender->EnableMenu(MENUStart); gBartender
->AddMenu(MENUSlidWindInfo, TRUE, 0); gBartender-> SetDimOption
(MENUSlidWindInfo, dimNONE); break;
case EnvisionDialog: TakingOutMenus(); gBartender->EnableMenu(MENUStart); break;
case CheckAnswer: TakingOutMenus(); gBartender->EnableMenu(MENUStart); break; }
}

void CLoopDoc::TakingOutMenus(void)
{
gBartender->DisableMenu(MENUfont); gBartender->RemoveMenu( MENUsize); gBartender->
RemoveMenu( MENUstyle); gBartender->RemoveMenu( MENUEnergyEq); gBartender->
RemoveMenu( MENUEnEqCheck); gBartender->RemoveMenu( MENUForcesEq); gBartender->
RemoveMenu( MENUForceDiagrams); gBartender->RemoveMenu( MENUForcesCheck); gBartender
->RemoveMenu( MENUSolution); gBartender->RemoveMenu( MENUSolCheck); gBartender->
RemoveMenu( MENUSlidWindInfo); }

void CLoopDoc::StartingMenus(void)
{
gBartender->EnableMenu(MENUfont); gBartender->EnableMenu(MENUStart); gBartender->
AddMenu(MENUsize, TRUE, MENUWindow); gBartender->AddMenu(MENUstyle, TRUE,
MENUWindow); gBartender->SetDimOption(MENUfont, dimNONE); gBartender->
SetDimOption(MENUsize, dimNONE); gBartender->SetDimOption(MENUstyle, dimNONE);
gBartender->SetUnchecking(MENUfont, TRUE); gBartender->SetUnchecking(MENUsize, TRUE);
gBartender->SetUnchecking(MENUstyle, TRUE); }

void CLoopDoc::Activate(void)
{
inherited::Activate(); itsWindow->BecomeGopher(TRUE); ChangingMenus(); protFile =
fopen("Protocol", "a+"); fprintf(protFile, "Untitled window accessed through clicking.\r");
fclose(protFile); }

void CLoopDoc::QuestionDialogClar(short DialogueId)
{
CDLOGDirector *theInfo; CDialog *aDialog = (CDialog*) itsWindow;
theInfo = new (CDLOGDirector); theInfo->IDLOGDirector(DialogueId, this); theInfo->
DoModalDialog(cmdOK); theInfo->EndDialog(cmdOK, TRUE); theInfo->Close(TRUE); aDialog->
Update(); }

static void NoteUsingMenu(long theCommand, Str255 itemTitle)
{
gBartender->GetCmdText(theCommand, itemTitle); rotFile = fopen("Protocol", "a+");
fprintf(protFile, "%s\r", itemTitle); fclose(protFile); }

static void NoteUsingButtons(Str255 theTitle)
{
protFile = fopen("Protocol", "a+"); fprintf(protFile, "%s\r", theTitle); fclose(protFile); }

```

CStartDirPanel.c

(the director of the dialogue windows)

```
#include the following: "Commands.h", "CStartDirPanel.h", "Global.h", "AppCommands.h",
<CBartender.h>, <CDialog.h>, <CRadioGroupPane.h>, <CDLOGDirector.h>, <CDialog.h>,
"CLoopDoc.h", "CLoopApp.h", <CDesktop.h>, <CAbstractText.h>, <CEditText.h>, "CDrawPane.h",
<TBUtilities.h>, "CEquations.h", "CForces.h", "CEnergyEquations.h", "CEnergyCheck.h",
"CForcesCheck.h", "CSolution.h", "CDLogDirSub.h", "CSimPane.h", <CPicture.h>, <stdio.h>
```

```
CStartDirPanel *theQuestion, *theEnvision, *theAnswerCheck; CSimPane *theCharPane;
CDLogDirSub *theSimulation; CPicture *thePicture;
extern CStartDirPanel *theStartWind; extern CBartender *gBartender; extern CBureaucrat
*gGopher; extern CWindow *itsWindow, **otherWindDial; extern CForces *theForcesEq; extern
CDesktop *gDesktop; extern CEnergyEquations *theEnergyEq; extern CEnergyCheck
*theEnergyCheck; extern CForcesCheck *theForcesCheck; extern CSolution *theSolution,
*theSolutionCheck; extern int windowItemNum, workSheetNum;
```

```
enum
{ kRadioItemsID1 = 4, kSolveQuestion, kPerformSimulation, kCheckAnswer, kForceEquation,
kEnergyEquations, kEnvision };
```

```
enum
{ kRadioChoiceHelpID = 13, kAttempt = 16, kCheck};
```

```
enum
{ kRadioEnvisionSlidRoll = 12, kEnvisionSliding, kEnvisionRolling };
```

```
enum
{ kRadioEnvsionWhereID = 1, kOverview, kIncline, kUpLoop, kDownLoop, kHorizontal,
kInsufficient};
```

```
void CStartDirPanel::IStartDirPanel(short DLOGid, CDirectorOwner *aSupervisor)
{ CDLOGDirector::IDLOGDirector( DLOGid, aSupervisor); }
```

```
void CStartDirPanel::DoCommand(long theCommand)
{ CursHandle gWatchCursor; CLoopDoc *theSessionQuit; CDialog *theDialogWind = (CDialog*)
itsWindow;
switch (theCommand) {
case cmdQuestion: gWatchCursor = GetCursor(watchCursor); SetCursor(*gWatchCursor);
if (theQuestion == NULL) {
theQuestion = new(CStartDirPanel); theQuestion->IStartDirPanel(Question, this); theQuestion->
BegDialDefBut(0, cmdCloseQuWind); InsMenuItem(gBartender->FindMacMenu(MENUWindow),
"pQuestion", windowItemNum); ((*theQuestion).itsWindow).ID = cmdQuestion; *(otherWindDial
+ windowItemNum) = (*theQuestion).itsWindow; windowItemNum++; }
else {
(*theQuestion).itsWindow->Select(); } InitCursor(); break;
case cmdStartSession: WhichRadBut(); break;
case cmdCloseQuWind: theQuestion->CloseDialogues(); theQuestion->
ThrowAwayDialog(cmdCloseQuWind); theQuestion = NULL; ForgetObject(theQuestion); break;
case cmdCentreMass: QuestionDialogClar(CentreMass); break;
case cmdRoll: QuestionDialogClar(RollClarif); break;
case cmdSlipping: QuestionDialogClar(Slipping); break;
case cmdUniform: QuestionDialogClar(Uniform); break;
case cmdClose:
if ((*itsWindow).ID == cmdEndSession)
ClosingSessionDialog();
else
theSessionQuit->SelWindQuit(); break;
case cmdEnvisionQuit: theEnvision->CloseDialogues(); theEnvision->
ThrowAwayDialog(cmdEnvisionQuit); theEnvision = NULL; ForgetObject(theEnvision); break;
case cmdAnswerQuit: theAnswerCheck->CloseDialogues(); theAnswerCheck->
ThrowAwayDialog(cmdEnvisionQuit); theAnswerCheck = NULL; ForgetObject(theAnswerCheck);
break;
case cmdStartEnvision: EnvisionSlidRoll(); break;
default: inherited::DoCommand(theCommand); break; }
}
```

```
void CStartDirPanel::BegDialDefBut(long theCommand, long theDefaultCmd)
```

```

{
    CDialog *theDialog = (CDialog*) itsWindow; CRadioGroupPane *radioGroupSet;

theDialog->Select(); itsWindow = theDialog;
if (theCommand == cmdStartPanel) {
    ((*theStartWind).itsWindow).ID = cmdStartPanel; (*theStartWind).itsWindow = itsWindow; }
else
    *(otherWindDial + windowItemNum) = itsWindow;
theDialog->SetDefaultCmd(theDefaultCmd); inherited::BeginDialog();
if (theDefaultCmd == cmdStartEnvision) {
    radioGroupSet = (CRadioGroupPane *)itsWindow->FindViewByID(kRadioEnvisionSlidRoll);
    radioGroupSet -> SetStationID(kEnvisionSliding); }
if (theDefaultCmd == cmdStartSession) {
    radioGroupSet = (CRadioGroupPane *)itsWindow->FindViewByID(kRadioItemsID1);
    radioGroupSet -> SetStationID(kForceEquation); radioGroupSet = (CRadioGroupPane *)itsWindow
->FindViewByID(kRadioChoiceHelpID); radioGroupSet -> SetStationID(kAttempt); }
}

void    CStartDirPanel::QuestionDialogClar(short DialogueId)
{
    CDLOGDirector *theInfo; CDialog *aDialog = (CDialog*) itsWindow;

theInfo = new (CDLOGDirector); theInfo->IDLOGDirector(DialogueId, this); theInfo->
DoModalDialog(cmdOK); theInfo->EndDialog(cmdOK, TRUE); theInfo->Close(TRUE); aDialog->
Update();}

void    CStartDirPanel::WhichRadBut(void)
{
    CursHandle gWatchCursor; CRadioGroupPane *radioGroupSet;
gWatchCursor = GetCursor(watchCursor); SetCursor(*gWatchCursor);
radioGroupSet(CRadioGroupPane *)itsWindow->FindViewByID(kRadioItemsID1);
switch (radioGroupSet -> GetStationID()) {
    case kSolveQuestion: case kEnergyEquations: case kForceEquation: InitCursor();
    WhichLevelHelpRB(); break;
    case kPerformSimulation: OpenSimulationDial(); break; case kCheckAnswer:
    OpenCheckAnswerDial(); break;
    case kEnvision: OpenEnvisionDial(); break;
    default: break; } InitCursor();
}

void    CStartDirPanel::WhichRadCheck(void)
{
    CursHandle gWatchCursor; CRadioGroupPane *radioGroupSet;

gWatchCursor = GetCursor(watchCursor); SetCursor(*gWatchCursor);
radioGroupSet = (CRadioGroupPane *)itsWindow->FindViewByID(kRadioItemsID1);
switch (radioGroupSet -> GetStationID()) {
    case kSolveQuestion: OpenSolCheckDial(); break;
    case kForceEquation: OpenForcesCheckDial(); break;
    case kEnergyEquations: OpenEnEqCheckDial(); break;}
InitCursor();
}

void    CStartDirPanel::WhichRadIntLevel(void)
{
    CursHandle gWatchCursor; CRadioGroupPane *radioGroupSet;

gWatchCursor = GetCursor(watchCursor); SetCursor(*gWatchCursor);
radioGroupSet = (CRadioGroupPane *)itsWindow->FindViewByID(kRadioItemsID1);
switch (radioGroupSet -> GetStationID()) {
    case kSolveQuestion: OpenSolutionDial(); break;
    case kForceEquation: OpenForcesDialog(); break;
    case kEnergyEquations: OpenDialEnergyEquations(); break; }
InitCursor();}

void    CStartDirPanel::WhichLevelHelpRB(void)
{
    CRadioGroupPane *radioGroupSet; radioGroupSet = (CRadioGroupPane *)itsWindow
->FindViewByID(kRadioChoiceHelpID);
switch (radioGroupSet -> GetStationID()) {
    case kAttempt: WhichRadIntLevel(); break;
    case kCheck: WhichRadCheck(); break;
    default: break; }
}

```

```

void    CStartDirPanel::ThrowAwayDialog(long closingButton)
{
    CDialog *theDialogStart = (CDialog*) itsWindow;

inherited::EndDialog(closingButton, FALSE); theDialogStart->Select(); theDialogStart->Close();
if (closingButton != cmdEndSession)
windowItemNum--; }

void    CStartDirPanel::CloseDialogues( void)
{
    Str255 theTitle; short z = 0; CWindow *theWindow;

itsWindow->GetTitle(theTitle); z = gBartender->FindItemText( MENUWindow, theTitle);
gBartender->RemoveMenuCmd(gBartender->FindCmdNumber( MENUWindow, z));
for (z = z; z < windowItemNum; z++)
*(otherWindDial + z) = *(otherWindDial + z + 1); *(otherWindDial + z) = NULL;
ForgetPtr(*(otherWindDial + z));
}

void    CStartDirPanel::ClosingSessionDialog(void)
{
    CWindow *theWindow; CLoopDoc *inOuttMenus;

windowItemNum--; gBartender->RemoveMenuCmd(cmdStartPanel); gBartender->
InsertMenuCmd(cmdStartPanel, "pStart Session", MENUStart, 0); inOuttMenus = new
(CLoopDoc); inOuttMenus->TakingOutMenus(); theStartWind-
>ThrowAwayDialog(cmdEndSession);
theStartWind = NULL; gBartender->EnableMenu(MENUfont); gBartender->AddMenu(MENUsize,
TRUE, MENUWindow); gBartender->AddMenu(MENUstyle, TRUE, MENUWindow); gBartender->
SetDimOption(MENUfont, dimNONE); gBartender->SetDimOption(MENUsize, dimNONE);
gBartender->SetDimOption(MENUstyle, dimNONE); gBartender->SetUnchecking(MENUfont,
TRUE); gBartender->SetUnchecking(MENUsize, TRUE); gBartender->SetUnchecking(MENUstyle,
TRUE);}

void    CStartDirPanel::Activate(void)
{
    CLoopDoc *theWindowChanging = NULL;

inherited::Activate(); theWindowChanging = new (CLoopDoc); theWindowChanging->
ChangingMenus(); }

void    CStartDirPanel::PlacetheChar(long theCommand)
{
    short theModifiers;

switch (theCommand) {
case cmdFriction: ((CEditText*) gGopher)->TypeChar('f', theModifiers); break;
case cmdgbtn: ((CEditText*) gGopher)->TypeChar('g', theModifiers); break;
case cmdhbtn: ((CEditText*) gGopher)->TypeChar('h', theModifiers); break;
case cmdlbtn: ((CEditText*) gGopher)->TypeChar('l', theModifiers); break;
case cmdMbtn: ((CEditText*) gGopher)->TypeChar('M', theModifiers); break;
case cmdrbtn: ((CEditText*) gGopher)->TypeChar('r', theModifiers); break;
case cmdRbtn: ((CEditText*) gGopher)->TypeChar('R', theModifiers); break;
case cmdybtn: ((CEditText*) gGopher)->TypeChar('y', theModifiers); break;
case cmdOpenParenbtn: ((CEditText*) gGopher)->TypeChar('(', theModifiers); break;
case cmdCloseParenbtn: ((CEditText*) gGopher)->TypeChar(')', theModifiers); break;
case cmdEqualbtn: ((CEditText*) gGopher)->TypeChar('=', theModifiers); break;
case cmdMultbtn: ((CEditText*) gGopher)->TypeChar('*', theModifiers); break;
case cmdDividebtn: ((CEditText*) gGopher)->TypeChar('/', theModifiers); break;
case cmdAddbtn: ((CEditText*) gGopher)->TypeChar('+', theModifiers); break;
case cmdSubtbtn: ((CEditText*) gGopher)->TypeChar('-', theModifiers); break;
case cmdSqRtbtn: ((CEditText*) gGopher)->TypeChar('√', theModifiers); break;
case cmdReturnbtn: ((CEditText*) gGopher)->TypeChar('r', theModifiers); break;
case cmdAnyVel: ((CEditText*) gGopher)->TypeChar('v', theModifiers); break;
case cmdTopVel: ((CEditText*) gGopher)->TypeChar(128, theModifiers); break;
case cmdBottomVel: ((CEditText*) gGopher)->TypeChar(130, theModifiers);break;
case cmdStartVel: ((CEditText*) gGopher)->TypeChar(147, theModifiers); break;
case cmdAnyAngVel: ((CEditText*) gGopher)->TypeChar(133, theModifiers);
case cmdTopAngVel: ((CEditText*) gGopher)->TypeChar(134, theModifiers); break;
case cmdBottomAngVel: ((CEditText*) gGopher)->TypeChar(137, theModifiers); break;
case cmdAccel: ((CEditText*) gGopher)->TypeChar('a', theModifiers); break;
case cmdForces: ((CEditText*) gGopher)->TypeChar('F', theModifiers); break;
case cmdReaction: ((CEditText*) gGopher)->TypeChar('N', theModifiers); break;

```

```

case cmdPhi: ((CEditText*) gGopher)->TypeChar('Ø', theModifiers); break;
case cmdBetabtn: ((CEditText*) gGopher)->TypeChar(167, theModifiers); break; }
}

```

```

void CStartDirPanel::Squaring(void)
{ tCharBuf charBuf; long selPresent, selFinal; short theModifiers;

```

```

((CEditText*) gGopher)->GetSelection(&selPresent, &selFinal);
if (selPresent == selFinal) ((CEditText*) gGopher)->SetSelection(selPresent - 1, selFinal, TRUE);
((CEditText*) gGopher)->GetCharBefore(&selPresent, charBuf);
switch (charBuf[1]) {
case 'g': ((CEditText*) gGopher)->TypeChar(143, theModifiers); break;
case 'h': ((CEditText*) gGopher)->TypeChar(144, theModifiers); break;
case 'l': ((CEditText*) gGopher)->TypeChar(141, theModifiers); break;
case 'y': ((CEditText*) gGopher)->TypeChar(145, theModifiers); break;
case 'r': ((CEditText*) gGopher)->TypeChar(139, theModifiers); break;
case 'R': ((CEditText*) gGopher)->TypeChar(140, theModifiers); break;
case 'M': ((CEditText*) gGopher)->TypeChar(142, theModifiers); break;
case 'v': ((CEditText*) gGopher)->TypeChar(132, theModifiers); break;
case 128: ((CEditText*) gGopher)->TypeChar(129, theModifiers); break;
case 130: ((CEditText*) gGopher)->TypeChar(131, theModifiers); break;
case 133: ((CEditText*) gGopher)->TypeChar(138, theModifiers); break;
case 134: ((CEditText*) gGopher)->TypeChar(135, theModifiers); break;
case 137: ((CEditText*) gGopher)->TypeChar(136, theModifiers); break;
case 147: ((CEditText*) gGopher)->TypeChar(146, theModifiers); break;
case 'a': ((CEditText*) gGopher)->TypeChar(148, theModifiers); break;
case 'F': ((CEditText*) gGopher)->TypeChar(149, theModifiers); break;
case 'N': ((CEditText*) gGopher)->TypeChar(150, theModifiers); break;
case 175: ((CEditText*) gGopher)->TypeChar(151, theModifiers); break;
case 167: ((CEditText*) gGopher)->TypeChar(152, theModifiers); break;
default: ((CEditText*) gGopher)->SetSelection(selFinal, selFinal, TRUE); break; }
}

```

```

void CStartDirPanel::OpenDialEnergyEquations(void)
{ CLoopDoc *theChangingMenus;

```

```

if ( theEnergyEq == NULL) {
theChangingMenus = new(CLoopDoc); theChangingMenus->TakingOutMenus(); gBartender->
AddMenu(MENUEnergyEq, TRUE, 0); gBartender->SetDimOption(MENUEnergyEq, dimNONE);
theEnergyEq = new(CEnergyEquations); theEnergyEq->IEnergyEquations(EnergyEquation, this);
theEnergyEq->BegDialEnergy(0); gBartender->EnableMenu(MENUSStart);
InsMenuItem(gBartender->FindMacMenu(MENUWindow), "pEnergy Expressions",
windowItemNum); (*theEnergyEq).itsWindow.ID = EnergyEquation; *(otherWindDial +
windowItemNum) = (*theEnergyEq).itsWindow; (*theEnergyEq).itsWindow->SetTitle("pEnergy
Expressions"); windowItemNum++; }
else {
theChangingMenus = new(CLoopDoc); theChangingMenus->ChangingMenus();
(*theEnergyEq).itsWindow->Select(); }
}

```

```

void CStartDirPanel::OpenForcesDialog(void)
{ CLoopDoc *theSessionQuit;

```

```

if ( theForcesEq == NULL) {
theSessionQuit = new(CLoopDoc); theSessionQuit->TakingOutMenus();
gBartender->AddMenu(MENUForcesEq, TRUE, 0); gBartender->AddMenu(MENUForceDiagrams,
TRUE, 0); gBartender->SetDimOption(MENUForceDiagrams, dimNONE); gBartender->
SetDimOption(MENUForcesEq, dimNONE); theForcesEq = new (CForces); theForcesEq->
IForces(ForcesEquations, this); theForcesEq->BegDialForces(0); gBartender->
EnableMenu(MENUSStart); InsMenuItem(gBartender->FindMacMenu(MENUWindow),
"pForce Equations", windowItemNum); (*theForcesEq).itsWindow.ID = ForcesEquations;
*(otherWindDial + windowItemNum) = (*theForcesEq).itsWindow; (*theForcesEq).itsWindow->
SetTitle("pForce Equations"); windowItemNum++; }
else {
theSessionQuit = new(CLoopDoc); theSessionQuit->ChangingMenus(); (*theForcesEq).itsWindow->
Select(); }
}

```

```

void CStartDirPanel::OpenEnEqCheckDial(void)
{
    CLoopDoc *theSessionQuit;

    if ( theEnergyCheck == NULL ) {
        theSessionQuit = new(CLoopDoc); theSessionQuit->TakingOutMenus(); gBartender->
        AddMenu(MENUEnEqCheck, TRUE, 0); gBartender->SetDimOption(MENUEnEqCheck,
        dimNONE); theEnergyCheck = new(CEnergyCheck); theEnergyCheck->
        IEnergyCheck(SlidEnEqCheck, this); theEnergyCheck->BegDialEnergyCheck(0); gBartender->
        EnableMenu(MENUSstart); InsMenuItem(gBartender->FindMacMenu(MENUWindow), "pCheck
        Energy Expressions", windowItemNum); (*theEnergyCheck).itsWindow.ID = SlidEnEqCheck;
        *(otherWindDial + windowItemNum) = (*theEnergyCheck).itsWindow;
        (*theEnergyCheck).itsWindow->SetTitle("pCheck Energy Expressions"); windowItemNum++;}
    else {
        theSessionQuit = new(CLoopDoc); theSessionQuit->ChangingMenus();
        (*theEnergyCheck).itsWindow->Select(); }
    }

void CStartDirPanel::OpenForcesCheckDial(void)
{ CLoopDoc *theSessionQuit;

    if ( theForcesCheck == NULL ) {
        theSessionQuit = new(CLoopDoc); theSessionQuit->TakingOutMenus(); gBartender->
        AddMenu(MENUForcesCheck, TRUE, 0); gBartender->SetDimOption(MENUForcesCheck,
        dimNONE); theForcesCheck = new(CForcesCheck); theForcesCheck->
        IForcesCheck(ForcesEquatCheck, this); theForcesCheck->BegDialForceCheck(cmdQuitForcesEq, 0);
        gBartender->EnableMenu(MENUSstart); InsMenuItem(gBartender->FindMacMenu(MENUWindow),
        "pCheck Force Equations", windowItemNum); (*theForcesCheck).itsWindow.ID =
        ForcesEquatCheck; *(otherWindDial + windowItemNum) = (*theForcesCheck).itsWindow;
        (*theForcesCheck).itsWindow->SetTitle("pCheck Force Equations"); windowItemNum++;}
    else {
        theSessionQuit = new(CLoopDoc); theSessionQuit->ChangingMenus();
        (*theForcesCheck).itsWindow->Select(); }
    }

void CStartDirPanel::OpenSolutionDial(void)
{ CLoopDoc *theSessionQuit;

    if ( theSolution == NULL ) {
        theSessionQuit = new(CLoopDoc); theSessionQuit->TakingOutMenus(); gBartender->
        AddMenu(MENUSolution, TRUE, 0); gBartender->SetDimOption(MENUSolution, dimNONE);
        gBartender->AddMenu(MENUForceDiagrams, TRUE, 0); gBartender->
        SetDimOption(MENUForceDiagrams, dimNONE); theSolution = new(CSolution); theSolution->
        ISolution(Solution, this); theSolution->BegDialSolution(cmdSolutionCheck); gBartender->
        EnableMenu(MENUSstart); InsMenuItem(gBartender->FindMacMenu(MENUWindow), "pPerform
        Solution", windowItemNum); (*theSolution).itsWindow.ID = Solution; *(otherWindDial +
        windowItemNum) = (*theSolution).itsWindow; (*theSolution).itsWindow->SetTitle("pPerform
        Solution"); windowItemNum++; }
    else {
        theSessionQuit = new(CLoopDoc); theSessionQuit->ChangingMenus(); (*theSolution).itsWindow->
        Select(); }
    }

void CStartDirPanel::OpenSolCheckDial(void)
{ CLoopDoc *theSessionQuit;

    if ( theSolutionCheck == NULL ) {
        theSessionQuit = new(CLoopDoc); theSessionQuit->TakingOutMenus(); gBartender->
        AddMenu(MENUSolCheck, TRUE, 0); gBartender->SetDimOption(MENUSolCheck, dimNONE);
        gBartender->AddMenu(MENUForceDiagrams, TRUE, 0); gBartender->
        SetDimOption(MENUForceDiagrams, dimNONE); theSolutionCheck = new(CSolution);
        theSolutionCheck->ISolution(SolutionCheck, this); theSolutionCheck->
        BegDialSolution(cmdCheckSolQuit); gBartender->EnableMenu(MENUSstart);
        InsMenuItem(gBartender->FindMacMenu(MENUWindow), "pPerform Solution Check",
        windowItemNum); (*theSolutionCheck).itsWindow.ID = SolutionCheck; *(otherWindDial +
        windowItemNum) = (*theSolutionCheck).itsWindow; (*theSolutionCheck).itsWindow->
        SetTitle("pPerform Solution Check"); windowItemNum++; }
    else {

```



```

theSessionQuit = new(CLoopDoc); theSessionQuit->ChangingMenus();
(*theSolutionCheck).itsWindow->Select(); }
}

void CStartDirPanel::OpenSimulationDial(void)
{
    CLoopDoc *theSessionQuit;

    if ( theSimulation == NULL) {
        theSessionQuit = new(CLoopDoc); theSessionQuit->TakingOutMenus();
        gBartender->AddMenu(MENUSlidWindInfo, TRUE, 0); gBartender->
        SetDimOption(MENUSlidWindInfo, dimNONE); theSimulation = new(CDLogDirSub);
        theSimulation->IDLogDirSub(DialogWindow, this); theSimulation->
        BeginDialogDefault(cmdDrawTrack, cmdStartSimulation, cmdErase, cmdBringIntoView);
        theSimulation->SetDefaultValues(3, cmdROne); theSimulation->SetDefaultValues(6,
        cmdhThreeHalfR); theSimulation->SetDefaultValues(9, cmdAlpha45); theSimulation->
        SetDefaultValues(12, cmdgOne); gBartender->EnableMenu(MENUStart);
        InsMenuItem(gBartender->FindMacMenu(MENUWindow), "\pPerform Simulation",
        windowItemNum); (*theSimulation).itsWindow.ID = DialogWindow; *(otherWindDial +
        windowItemNum) = (*theSimulation).itsWindow; (*theSimulation).itsWindow->SetTitle("\pPerform
        Simulation"); windowItemNum++; }
    else {
        theSessionQuit = new(CLoopDoc); theSessionQuit->ChangingMenus();
        (*theSimulation).itsWindow->Select(); }
    }

    void CStartDirPanel::OpenEnvisionDial(void)
    {
        CLoopDoc *theSessionQuit;

        if ( theEnvision == NULL) {
            theSessionQuit = new(CLoopDoc); theSessionQuit->TakingOutMenus(); theEnvision =
            new(CStartDirPanel); theEnvision->IStartDirPanel(EnvisionDialog, this); theEnvision->
            BegDialDefBut(0, cmdStartEnvision); gBartender->EnableMenu(MENUStart);
            InsMenuItem(gBartender->FindMacMenu(MENUWindow), "\pEnvision", windowItemNum);
            (*theEnvision).itsWindow.ID = EnvisionDialog; *(otherWindDial + windowItemNum) =
            (*theEnvision).itsWindow; (*theEnvision).itsWindow->SetTitle("\pEnvision"); windowItemNum++;
        }
        else {
            theSessionQuit = new(CLoopDoc); theSessionQuit->ChangingMenus(); (*theEnvision).itsWindow-
            > Select(); }
        }

        void CStartDirPanel::EnvisionSlidRoll(void)
        {
            CRadioGroupPane *radioGroupSet;

            radioGroupSet = (CRadioGroupPane *)itsWindow->FindViewByID(kRadioEnvisionSlidRoll);
            switch (radioGroupSet -> GetStationID()) {
            case kEnvisionSliding: EnvisionSliding(); break;
            case kEnvisionRolling: EnvisionRolling(); break;
            default: break; }
        }

        void CStartDirPanel::EnvisionSliding(void)
        {
            CRadioGroupPane *radioGroupSet;

            PrepareEnvision(); radioGroupSet = (CRadioGroupPane *)itsWindow->
            FindViewByID(kRadioEnvisionWhereID);
            switch (radioGroupSet -> GetStationID()) {
            case kOverview: thePicture->UsePICT(319); break;
            case kInline: thePicture->UsePICT(321); break;
            case kUpLoop: thePicture->UsePICT(323); break;
            case kDownLoop: thePicture->UsePICT(325); break;
            case kHorizontal: thePicture->UsePICT(327); break;
            case kInsufficient: thePicture->UsePICT(329); break;
            default: ParamText("\pWhat would you like to envision?", "\pMake a choice by clicking one of the
            radio buttons.", "\pQUALITATIVE ENVISIONING", "\pMAKE A CHOICE");
            PositionDialog('ALRT', 128); InitCursor; NoteAlert(128, NULL); return; break; }
        }

        void CStartDirPanel::EnvisionRolling(void)

```

```

{
    CRadioGroupPane *radioGroupSet;

PrepareEnvnsion(); radioGroupSet = (CRadioGroupPane *)itsWindow->
FindViewById(kRadioEnvnsionWhereID);
switch (radioGroupSet -> GetStationID()) {
case kOverview: thePicture->UsePICT(320); break;
case kIncline: thePicture->UsePICT(322); break;
case kUpLoop: thePicture->UsePICT(324); break;
case kDownLoop: thePicture->UsePICT(326); break;
case kHorizontal: thePicture->UsePICT(328); break;
case kInsufficient: thePicture->UsePICT(329); break;
default: ParamText("\pWhat would you like to envision?", "\pMake a choice by clicking one of the
radio buttons.", "\pQUALITATIVE ENVISIONING", "\pMAKE A CHOICE");
PositionDialog("ALRT", 128); InitCursor; NoteAlert(128, NULL); return; break; }
}

void    CStartDirPanel::OpenCheckAnswerDial(void)
{
    CLoopDoc      *theSessionQuit;

if ( theAnswerCheck == NULL)    {
theSessionQuit = new(CLoopDoc); theSessionQuit->TakingOutMenus(); theAnswerCheck =
new(CStartDirPanel); theAnswerCheck->IStartDirPanel(CheckAnswer, this); theAnswerCheck->
BegDialDefBut(0, cmdAnswerQuit); gBartender->EnableMenu(MENUStart);
InsMenuItem(gBartender-> FindMacMenu(MENUWindow), "\pCheck Answer", windowItemNum);
(*(*theAnswerCheck).itsWindow).ID = CheckAnswer; *(otherWindDial + windowItemNum) =
(*theAnswerCheck).itsWindow; (*theAnswerCheck).itsWindow->SetTitle("\pCheck Answer");
windowItemNum++; }
else
{
theSessionQuit = new(CLoopDoc); theSessionQuit->ChangingMenus();
(*theAnswerCheck).itsWindow->Select(); }
}

void    CStartDirPanel::PrepareEnvnsion(void)
{
theCharPane = new (CSimPane); theCharPane->ISimPane(EnvisionPane, itsWindow, this);
theCharPane->Prepare();
if (thePicture != NULL)
thePicture->Dispose(); thePicture = new(CPicture); thePicture->IPicture(itsWindow, this, 464, 150,
6, 160, sizELASTIC, sizELASTIC); theCharPane->Refresh();
}

```

CDLogDirSub.c

(The director for the simulations. It contains the methods which govern the simulation.))

```

#include the following: "Commands.h", "CDLogDirSub.h", <CPane.h>, <CWindow.h>,
<CDesktop.h>, "Global.h", "AppCommands.h", <CBartender.h>, <CDialog.h>, <CDialogText.h>,
<CAbstractText.h>, <CDialogDirector.h>, <CScrollPane.h>, "CSimPane.h", <SANE.h>,
<CEditText.h>, <ctype.h>, <CRadioGroupPane.h>, <TBUtilities.h>, "CLoopDoc.h", <CPicture.h>,
"CEquations.h"

enum
{
    kRadioSimulationID = 25, kSlidingSimulation, kRollingSimulation    };

extern CDesktop *gDesktop; extern CBartender *gBartender; extern CBureaucrat *gGopher; extern
CWindow **otherWindDial; extern CDLogDirSub *theSimulation; CSimPane *theSimPane,
*theScalePane, *theLargeScalePane, *theInfoPane; double Pi = 3.14159265358979323846; double
ht, R, Alpha, g, H; double readR, readht, readAlpha, readg; int Track, SIZFACTOR,
SIZFACTORExt, PaneWidth, side = 16; Str255 scaleNum; int Sliding, Rolling, picture,
Oscillating; extern int windowItemNum; extern double xPos[80], yPos[80], angDisp[80], noPos;
PicHandle theSimPic; Rect picRect;

void    CDLogDirSub::IDLogDirSub(short DLOGid, CDirectorOwner *aSupervisor)
{
    CDLOGDirector::IDLOGDirector( DLOGid, aSupervisor);
}

void    CDLogDirSub::PrepareSim(short PaneId, short startX, short startY, int penWidth, int
penHeight)
{
    PenSize(penWidth, penHeight); theSimPane = new (CSimPane); theSimPane->ISimPane(PaneId,
itsWindow, this); theSimPane->Prepare();
}

```

```

void    CDLogDirSub::SimTrack(short PanelId, short startX, short startY, int penWidth, int
penHeight)
{
    CDialog *theDialogWindow = (CDialog*) itsWindow;
SetDrawValues(3.6);
if (R == 0 || g == 0 || Alpha == 0) {
    ParamText("\p", "\pR, h, g and Ø must have positive non zero values.", "\pUNACCEPTABLE
VALUE(S)", "\p"); PositionDialog('ALRT', 128); InitCursor; NoteAlert(128, NULL); return; }
if (readAlpha < 10 || readAlpha > 85) {
    ParamText("\p", "\pRestrict your choice to values between 10 and 85 degrees.", "\pØ VALUE
OUTSIDE RANGE", "\p"); PositionDialog('ALRT', 128); InitCursor; NoteAlert(128, NULL);
return; }
if (readg < 1 || readg > 1000) {
    ParamText("\p", "\pRestrict your choice to values between 1 and 1000%.", "\pg VALUE
OUTSIDE RANGE", "\p"); PositionDialog('ALRT', 128); InitCursor; NoteAlert(128, NULL);
return; }
if (ht == 180 && R < 16) {
    ParamText("\p", "\pTo perform a simulation, choose a larger value for R relative to h.", "\pLOOP
RADIUS TOO SMALL", "\p"); PositionDialog('ALRT', 128); InitCursor; NoteAlert(128, NULL);
return; }
theDialogWindow->SetCmdEnable(cmdDrawTrack, FALSE);
PrepareSim(PanelId, startX, startY, penWidth, penHeight);
theSimPane->DrawTrack(PanelId, 2, 15, 1, 1);
theDialogWindow->SetCmdEnable(cmdErase, TRUE);
theDialogWindow->SetCmdEnable(cmdStartSimulation, TRUE);
theDialogWindow->SetDefaultCmd(cmdStartSimulation);
if (readAlpha <= 32 || (readAlpha <= 37 && R == 90)) {
    theDialogWindow->SetCmdEnable(cmdStartSimulation, FALSE); theDialogWindow->
SetCmdEnable(cmdBringIntoView, TRUE); theDialogWindow-> SetDefaultCmd
(cmdBringIntoView); }
}

void    CDLogDirSub::DoCommand(long theCommand)
{
    CDialog *theDialogWindow = (CDialog*) itsWindow; CDLogDirSub *theLargerWindow;
CursHandle gWatchCursor; CSimPane *theDiag; double Critical; int changed = 0;
switch(theCommand) {
    case cmdStartSimulation: SlidingRolling(); changed = CheckDraw(2, 15, 1, 1, 0.5, Critical);
if (changed)
    return;
theDialogWindow->SetCmdEnable(cmdStartSimulation, FALSE);
if (ht == 0)
    theSimPane->StatOsc(2, 15, 1);
if (ht > R) {
    gWatchCursor = GetCursor(watchCursor); SetCursor(*gWatchCursor); MakeSimPicture(2,
15, 1, 1, 0.5, Critical); InitCursor(); }
if (ht != 0 && ht <= R) {
    OscillateBall(2, 15, 1, 1, 0.5); }
theDialogWindow->SetDefaultCmd(cmdErase); theDialogWindow->SetCmdEnable(cmdErase,
TRUE); break;
    case cmdDrawTrack: gWatchCursor = GetCursor(watchCursor); SetCursor(*gWatchCursor);
if (Track) {
    Track = 0; picture = 0; theSimPane->Refresh(); theScalePane->Refresh(); return; }
SimTrack(SimulationPane, 2, 15, 1, 1); InitCursor(); break;
    case cmdRHalf:
    case cmdROneHalf: case cmdRTwo: case cmdRTwoHalf: case cmdRThree: case cmdROne:
case cmdRThreeHalf: case cmdRFour: case cmdRFourHalf: case cmdRFive: case cmdROther:
    ChoosingValues(3, theCommand, cmdROther, OtherRWindow); break;
    case cmdhOneR: case cmdhHalfR: case cmdhOneHalfR: case cmdhTwoR: case
cmdhTwoHalfR: case cmdhThreeR: case cmdhThreeHalfR: case cmdhFourR: case
cmdhFourHalfR: case cmdhFiveR: case cmdhFiveHalfR: case cmdhSixR: case cmdhOther:
    ChoosingValues(6, theCommand, cmdhOther, OtherhWindow); break;
    case cmdAlpha10: case cmdAlpha15: case cmdAlpha20: case cmdAlpha25: case cmdAlpha30:
case cmdAlpha35: case cmdAlpha40: case cmdAlpha45: case cmdAlpha50: case cmdAlpha55:
case cmdAlpha60: case cmdAlpha65: case cmdAlpha70: case cmdAlpha75: case
cmdAlphaOther:
    ChoosingValues(9, theCommand, cmdAlphaOther, OtherAlphaWindow); break;
    case cmdgOne: case cmdgHalf: case cmdgOneAndHalf: case cmdgTwo: case
cmdgTwoAndHalf:
    case cmdgThree: case cmdgThreeAndHalf: case cmdgFour: case
cmdgFourAndHalf: case cmdgFive: case cmdgOther:

```

```

        ChoosingValues(12, theCommand, cmdgOther, OthergWindow); break;
case cmdSetDefaultValues: SetDefaultValues(3, cmdROne); SetDefaultValues(6,
cmdhThreeHalfR); SetDefaultValues(9, cmdAlpha45); SetDefaultValues(12, cmdgOne); break;
case cmdDone: gWatchCursor = GetCursor(watchCursor); SetCursor(*gWatchCursor);
theSimulation->CloseSimulationDial(); theSimulation->ThrowSimulationDial(cmdDone);
theSimulation = NULL; ForgetObject(theSimulation); Track = 0; picture = 0;
KillPicture(theSimPic); ForgetHandle(theSimPic); InitCursor(); break;
case cmdOKLarge: Track = 0; CDialogDirector::EndDialog(cmdOKLarge, TRUE);
theDialogWindow->Select(); theDialogWindow->Close(); gBartender->EnableMenu
(MENUSlidWindInfo); break;
case cmdDrawTrackLarge: gWatchCursor = GetCursor(watchCursor);
SetCursor(*gWatchCursor); theDialogWindow->SetCmdEnable(cmdDrawTrackLarge,
FALSE); ResetDrawValues(3.6); PrepareSim(DrawingPane, 2, 15, 1, 1); theSimPane->
DrawTrack(DrawingPane, 2, 15, 1, 1); theDialogWindow->SetCmdEnable(cmdSimulateLarge,
TRUE); InitCursor(); break;
case cmdBringIntoView: SlidingRolling(); Track = 0; KillPicture(theSimPic); gBartender->
DisableMenu(MENUSlidWindInfo); theLargerWindow = new (CDLogDirSub);
theLargerWindow->IDLogDirSub(DrawingWindow, this); theLargerWindow->
BeginDialogDefault(cmdOKLarge, cmdSimulateLarge, 0, 0); theDialogWindow->
SetCmdEnable(cmdErase, FALSE); theDialogWindow->SetCmdEnable(cmdBringIntoView,
FALSE); theDialogWindow->SetCmdEnable(cmdStartSimulation, FALSE);
theDialogWindow->SetCmdEnable(cmdDrawTrack, TRUE); theDialogWindow->
SetDefaultCmd(cmdDrawTrack); break;
case cmdSimulateLarge: theDialogWindow->SetCmdEnable(cmdSimulateLarge, FALSE);
gWatchCursor = GetCursor(watchCursor); SetCursor(*gWatchCursor);
if (ht > R)
    theSimPane->SimSlidRoll(2, 15, 1, 1, 0.5, Critical);
else
    OscillateBall(2, 15, 1, 1, 0.5);
InitCursor(); break;
case cmdErase: Track = 0; picture = 0; KillPicture(theSimPic); theSimPane->Refresh();
theScalePane->Refresh(); theDialogWindow->SetCmdEnable(cmdErase, FALSE);
theDialogWindow->SetCmdEnable(cmdStartSimulation, FALSE); theDialogWindow->
SetCmdEnable(cmdBringIntoView, FALSE); theDialogWindow->SetCmdEnable
(cmdDrawTrack, TRUE); theDialogWindow->SetDefaultCmd(cmdDrawTrack); break;
    default: inherited::DoCommand(theCommand); break;
}

void    CDLogDirSub::ChoosingValues(short dialogItemNo, long theCommand,
                                   long OtherCmd, short OtherWindow)
{
    CDialog *theDialog = (CDialog*) itsWindow; Str255 theText; tGopherInfo gopherInfo;
    CView *newGopher;
    gBartender->GetCmdText(theCommand, theText);
    do
    {
        gopherInfo.currentGopher = (CView*) gGopher; theDialog->FindGophers (&gopherInfo);
        newGopher = gopherInfo.nextGopher; newGopher->BecomeGopher(TRUE);
    }
    while(newGopher->ID != dialogItemNo);
    if (theCommand != OtherCmd)
        ((CAbstractText*) newGopher)->SetTextString(theText);
    else
        ProducingDialogBox(OtherWindow);
}

void    CDLogDirSub::SetDefaultValues(short dialogItemNo, long theCommand)
{
    tGopherInfo gopherInfo; CDialog *theDialog = (CDialog*) itsWindow; CView
    *newGopher; Str255 theText;
    gBartender->GetCmdText(theCommand, theText);
    do
    {
        gopherInfo.currentGopher = (CView*) gGopher; theDialog->FindGophers(&gopherInfo);
        newGopher = gopherInfo.nextGopher; newGopher->BecomeGopher(TRUE);
    }
    while(newGopher->ID != dialogItemNo);
    ((CAbstractText*) newGopher)->SetTextString(theText);
}

void    CDLogDirSub::BeginDialogDefault(long theCommand, long cmdId1, long cmdId2, long
                                   cmdId3)
{
    CDialog *theDialog = (CDialog*) itsWindow; CRadioGroupPane *radioGroupSet;
    theDialog->Select(); theDialog->SetDefaultCmd(theCommand); theDialog->SetCmdEnable(cmdId1,
FALSE); theDialog->SetCmdEnable(cmdId2, FALSE); theDialog->SetCmdEnable(cmdId3, FALSE);
inherited::BeginDialog();
if (theCommand == cmdDrawTrack)
{

```

```

radioGroupSet = (CRadioGroupPane *)itsWindow->FindViewById(kRadioSimulationID);
radioGroupSet -> SetStationID(kSlidingSimulation); Sliding = 1; Rolling = 0;
}

double  CDLogDirSub::ReadingValues(short dialogItemNo)
{
    CDialog *theDialog = (CDialog*) itsWindow; tGopherInfo gopherInfo;
    CView *newGopher; Handle tonumEntry; double num, decimal; int i = 0;
do
    {
        gopherInfo.currentGopher = (CView*) gGopher; theDialog->FindGophers(&gopherInfo); newGopher
        = gopherInfo.nextGopher; newGopher->BecomeGopher(TRUE); }
while(newGopher->ID != dialogItemNo);
tonumEntry = ((CEditText*) newGopher)->CopyTextRange(0, ((CEditText*) newGopher)->
GetLength());
if ( (*tonumEntry + i) == '-')    {
    R = 0; return 0;
}
for(num = 0.0; isdigit(*tonumEntry + i)); i++)
    num = 10.0 * num + (*tonumEntry + i) - '0';
if ( (*tonumEntry + i) == '.')
    i++;
for(decimal = 1.0; isdigit(*tonumEntry + i)); i++)    {
    num = 10.0 * num + (*tonumEntry + i) - '0'; decimal *= 10.0;    }
num = num / decimal;
if (*tonumEntry + i) == 'R')    {
    if (num == 0)
        num = 1.0 * readR;
    if (R == 0)
        num = 0;
    if (R != 0 )
        num = num * readR;
}
return num;
}

void  CDLogDirSub::SetDrawValues(double startRforH)
{
    CDialog *theDialogWindow = (CDialog*) itsWindow;

    SIZFACTOR = 50; R = ReadingValues(3); readR = R; readht = ReadingValues(6); readAlpha =
    ReadingValues(9); readg = ReadingValues(12); g = readg * 9.81/100; ht = readht * SIZFACTOR;
    Alpha = readAlpha * Pi/180; = R * SIZFACTOR; H = startRforH * SIZFACTOR;
    if (R != 0)
    SetScale(ScalePane);
}

void  CDLogDirSub::ResetDrawValues(double startRforH)
{
    double scaleFactor, readRad, Scale; int scalePaneHeight = 20, scalePaneWidth = 96;

    PaneWidth = LargeDrawPaneWidth; readRad = R; scaleFactor =
    LargeDrawPaneWidth/DrawingPaneWidth; SIZFACTOR *= scaleFactor; ht *= scaleFactor; R *=
    scaleFactor;
    while (ht/tan(Alpha) + R*(1.5 + sin(Alpha)) > PaneWidth) {
        Scale = R; R -= 5; ht = ht * R / Scale;
    }
    SIZFACTORExt = SIZFACTOR * R / readRad; H = startRforH * SIZFACTORExt;
    SetScale(LargeScalePane);
}

int  CDLogDirSub::CheckDraw(short startX, short startY, short TrackWidth,
    short TrackHeight, double deltaT, double Critical)
{
    CDialog *theDialog = (CDialog*) itsWindow; CSimPane *theDiag; CEquations
    *theEquations;

    if (Track && (readR != ReadingValues(3) || readht != ReadingValues(6) || readAlpha !=
    ReadingValues(9) || readg != ReadingValues(12) ))    {
        theDialog->SetCmdEnable(cmdErase, TRUE); theDialog->SetCmdEnable(cmdStartSimulation,
        FALSE); theDialog->SetDefaultCmd(cmdErase); theDialog->SetCmdEnable(cmdBringIntoView,
        FALSE); theEquations = new (CEquations); theEquations->ModalDialogPlace("\pThe values for
        simulation have changed. You need to redraw the track.\rClick the \"Erase\" button followed by the
        \"Draw Track\" button.", "\p", "\p", "\p", 2005); return(1);    } return(0);
}

void  CDLogDirSub::ProducingDialogBox(short DialogueId)
{
    CDLOGDirector *theInfo; CDialog *theDialog = (CDialog*) itsWindow;

```

```

theInfo = new (CDLOGDirector); theInfo->IDLOGDirector(DialogueId, this); theInfo->
DoModalDialog(cmdOK); theInfo->EndDialog(cmdOK, TRUE); theInfo->Close(TRUE); theDialog-
> Update();
}

void    CDLogDirSub::SetScale(short    PaneDialogID)
{
    double cmtoPts, Scale; int scalePaneHeight = 20, scalePaneWidth = 96, i = 1; Str255
    UnitsFrom = "\p1cm : ", UnitsTo = "\pmm", scaleNo;

    cmtoPts = 72 / 2.54;
    if (readht - 2*readR >= 0 )      {
        Scale = readht * cmtoPts/ H; R = H * R/ht; ht = H;      }
    else {
        Scale = 2 * readR * cmtoPts/ H; ht = H * ht/(R * 2); R = H/2;      }
    if (Scale <= 0.1) {
        Scale = Scale * 1000; NumToString(Scale, scaleNo); scaleNum[0] = 5; scaleNum[1] = '0';
        scaleNum[2] = '.'; scaleNum[3] = '0'; for(i = 4; i <= scaleNum[0]; i++) scaleNum[i] = scaleNo[i -
        2];
        Scale = Scale/1000;      }
    if (Scale > 0.1 && Scale < 1)      {
        Scale = Scale * 100; NumToString(Scale, scaleNo); scaleNum[0] = 4; scaleNum[1] = '0';
        scaleNum[2] = '.'; for(i = 3; i <= scaleNum[0]; i++) scaleNum[i] = scaleNo[i - 2]; Scale =
        Scale/100;      }
    if ( Scale >= 1 && Scale < 10)      {
        Scale = Scale * 100; NumToString(Scale, scaleNo); scaleNum[0] = scaleNo[0] + 1;
        for(i = 1; i <= 1; i++)
            scaleNum[i] = scaleNo[i]; scaleNum[i++] = '.';
        for(i = i; i <= scaleNo[0] + 1; i++)
            scaleNum[i] = scaleNo[i - 1]; Scale = Scale/100;      }
    if (Scale >= 10) {
        Scale = Scale * 100; NumToString(Scale, scaleNo); scaleNum[0] = scaleNo[0] + 1;
        for(i = 1; i <= 2; i++)
            scaleNum[i] = scaleNo[i]; scaleNum[i++] = '.';
        for(i = i; i <= scaleNo[0] + 1; i++)
            scaleNum[i] = scaleNo[i - 1];      }
    theScalePane = new (CSimPane); theScalePane->ISimPane(PaneDialogID, itsWindow, this);
    theScalePane->Prepare(); MoveTo (scalePaneWidth/20, 3*scalePaneHeight/4);
    DrawString(UnitsFrom); DrawString(scaleNum); DrawString(UnitsTo);      }

void    CDLogDirSub::ThrowSimulationDial(long closingButton)
{
    CDialog *theDialogStart = (CDialog*) itsWindow;

    inherited::EndDialog(closingButton, FALSE); theDialogStart->Select(); theDialogStart->Close();
    windowItemNum--;
}

void    CDLogDirSub::ThrowAboutDial(void)
{
    CDialog *theDialogStart = (CDialog*) itsWindow;

    theDialogStart->Select(); theDialogStart->Close();      }

void    CDLogDirSub::CloseSimulationDial( void)
{
    Str255 theTitle; short z = 0; CWindow *theWindow;

    itsWindow->GetTitle(theTitle); z = gBartender->FindItemText( MENUWindow, theTitle);
    gBartender
    ->RemoveMenuCmd(gBartender->FindCmdNumber( MENUWindow, z));
    for (z = z; z < windowItemNum; z++)
        *(otherWindDial + z) = *(otherWindDial + z + 1);
    *(otherWindDial + z) = NULL; ForgetPtr(*(otherWindDial + z));      }

void    CDLogDirSub::SlidingRolling(void)
{
    CRadioGroupPane *radioGroupSet;

    radioGroupSet = (CRadioGroupPane *)itsWindow->FindViewByID(kRadioSimulationID);
    switch (radioGroupSet -> GetStationID()) {
        case kSlidingSimulation: Sliding = 1; Rolling = 0; break;
        case kRollingSimulation: Rolling = 1; Sliding = 0; break;      }
}

```

```

void    CDLogDirSub::MakeSimPicture(short startX, short startY, short TrackWidth,
                                     short TrackHeight, double deltaT, double Critical)
{
    CDialog *theDialogWindow = (CDialog*) itsWindow;

Track = 0; SetRect(&picRect, 0, 0, 358, 207); ClipRect(&picRect); theSimPic =
OpenPicture(&picRect); ShowPen(); theSimPane->Prepare(); theSimPane->DrawTrack(
SimulationPane, startX, startY, TrackWidth, TrackHeight); theSimPane->SimSlidRoll(startX, startY,
TrackWidth, TrackHeight, deltaT, Critical); HidePen(); ClosePicture(); picture = 1;    }

void    CDLogDirSub::Activate(void)
{
    CLoopDoc *theWindowChanging = NULL;

inherited::Activate(); theWindowChanging = new (CLoopDoc); theWindowChanging->
ChangingMenus();    }

void    CDLogDirSub::BallIsOscillating(void)
{
    unsigned long *Ticks; int i = 0; double L2 = 0, T2 = 0, R2 = 0, B2 = 0, Left = 0, Top =
0, Right = 0, Bottom = 0; Rect Ball;

theSimPane->Prepare();
for (i = 1; i < noPos; i++)    {
    Left = xPos[i]; Top = yPos[i] - 1; Right = Left + side; Bottom = Top + side; SetRect(&Ball,
Left,
    Top, Right, Bottom); FrameOval(&Ball);
if (Rolling)    {
    MoveTo(Left + side/2, Top + side/2); LineTo( Left + side/2 - (side - 2)*sin(angDisp[i])/2, Top +
side/2 + (side - 2)*cos(angDisp[i])/2);    }
Delay(10, &Ticks); L2 = Left; T2 = Top; R2 = Right; B2 = Bottom; SetRect(&Ball, L2, T2, R2,
B2); FillOval(&Ball, white);    }
}

void    CDLogDirSub::BallIsReturning(void)
{
    unsigned long *Ticks; int i = noPos; double L2 = 0, T2 = 0, R2 = 0, B2 = 0, Left = 0,
Top = 0, Right = 0, Bottom = 0; Rect Ball;

theSimPane->Prepare();
for (i = i - 1; i > 1; i--)    {
    Left = xPos[i]; Top = yPos[i] - 1; Right = Left + side; Bottom = Top + side; SetRect(&Ball,
Left,
    Top, Right, Bottom); FrameOval(&Ball);
if (Rolling)    {
    MoveTo(Left + side/2, Top + side/2); LineTo( Left + side/2 - (side - 2)*sin(angDisp[i])/2, Top +
side/2 + (side - 2)*cos(angDisp[i])/2);    }
Delay(10, &Ticks); L2 = Left; T2 = Top; R2 = Right; B2 = Bottom; SetRect(&Ball, L2, T2, R2,
B2); FillOval(&Ball, white);    }
}

void    CDLogDirSub::OscillateBall(short startX, short startY, short TrackWidth, short
TrackHeight, double deltaT)
{
    Point hitPt; short modifierKeys; long when; int j = 0;

if (picture)    {
    KillPicture(theSimPic); picture = 0; ForgetHandle(theSimPic);    }
Track = 1;
if ( ht <= R && ht > R*(1 - cos(Alpha))) {
    theSimPane->FramingSlope(startX, startY, TrackWidth, TrackHeight, deltaT); theSimPane->
FramingOscill(deltaT);
for (j = 0; j < 2; j++)    {
    BallIsOscillating(); BallIsReturning();    }
}
if (ht <= R*(1 - cos(Alpha)) /*&& ht > side*/)    {
    theSimPane->FrameInLoopOsc(startX, startY, TrackWidth, deltaT);
for (j = 0; j < 2; j++)    {
    BallIsOscillating(); BallIsReturning();    }
}
theSimPane->Refresh();    }

```

CEditPane.c

(Methods for a text editing pane)

```
#include the following: "CDrawPane.h", <Commands.h>, <CDocument.h>, <CBartender.h>,
<Constants.h>, "CLoopDoc.h", "AppCommands.h", <CWindow.h>, <CDirector.h>
```

```
extern CBartender gBartender;
```

```
void CEditPane::IEditPane(CView *anEnclosure, CBureaucrat *aSupervisor)
{
    Rect margin;
```

```
CEditText::IEditText(anEnclosure, aSupervisor, 1, 1, 0, 0, sizELASTIC, sizELASTIC, 432);
FitToEnclosure(TRUE, TRUE); SetRect(&margin, 2, 2, -2, -2); ChangeSize(&margin, FALSE); }
```

```
void CEditPane::DoCommand(long theCommand)
{
    if (((theCommand == cmdPaste) || (theCommand == cmdCut)) && !((CDocument *)itsSupervisor)
->dirty) {
        ((CDocument *)itsSupervisor)->dirty = TRUE; gBartender->EnableCmd(cmdSave); gBartender->
        EnableCmd(cmdSaveAs);
    }
    inherited::DoCommand(theCommand);
}
```

```
void CEditPane::DoKeyDown(char theChar, Byte keyCode, EventRecord *macEvent)
{
    inherited::DoKeyDown(theChar, keyCode, macEvent);
    switch (keyCode) {
        case KeyHome: case KeyEnd: case KeyPageUp: case KeyPageDown: break;
        default:
            if (!((CDocument *)itsSupervisor)->dirty) {
                ((CDocument *)itsSupervisor)->dirty = TRUE; gBartender->EnableCmd(cmdSave);
                gBartender->EnableCmd(cmdSaveAs);
            }
            break;
    }
}
```

```
void CEditPane::DoAutoKey(char theChar, Byte keyCode, EventRecord *macEvent)
{
    inherited::DoAutoKey(theChar, keyCode, macEvent);
    switch (keyCode) {
        case KeyHome: case KeyEnd: case KeyPageUp: case KeyPageDown: break;
        default:
            if (!((CDocument *)itsSupervisor)->dirty) {
                ((CDocument *)itsSupervisor)->dirty = TRUE; gBartender->EnableCmd(cmdSave);
                gBartender->EnableCmd(cmdSaveAs);
            }
            break;
    }
}
```

CEnergyCheck.c

(This file performs the checks for the energy expressions)

```
#include the following: "Commands.h", "CEnergyCheck.h", "Global.h", "AppCommands.h",
<CBartender.h>, <CDialog.h>, "CLoopDoc.h", "CLoopApp.h", <CDesktop.h>, <TBUtilities.h>,
<CRadioGroupPane.h>, <CEditText.h>, <CRadioGroupPane.h>, "CStartDirPanel.h", "CEquations.h"
```

```
enum
{
    kRadioEnEqCheckID = 1, kSlidingEnEqCheck = 40, kRollingEnEqCheck    };

```

```
enum
{
    kPos1TKE = 28, kPos1RKE, kPos1GPE, kPos1Tot, kPos2TKE, kPos2RKE, kPos2GPE,
    kPos2Tot, kEqualTot    };

```

```
CEnergyCheck *theEnergyCheck; extern CWindow **otherWindDial; extern CBartender
*gBartender; extern CBureaucrat *gGopher; extern int windowItemNum; int whichPos12,
whichPos1, whichPos2, Motion, textBox; long Length; Handle toPrevEntry; short num, den;
Boolean Good, Totals = FALSE;
```

```
void    CEnergyCheck::IEnergyCheck(short DLOGid, CDirectorOwner *aSupervisor)
{
    CDLOGDirector::IDLOGDirector( DLOGid, aSupervisor);
}
```



```

void    CEnergyCheck::BegDialEnergyCheck(long theCommand)
{
    CDialog *theDialog = (CDialog*) itsWindow; CRadioGroupPane *radioGroupSet;

theDialog->Select(); inherited::BeginDialog(); radioGroupSet = (CRadioGroupPane *)itsWindow->
FindViewById(kRadioEnEqCheckID); radioGroupSet -> SetStationID(kSlidingEnEqCheck);
Motion = 1;    }

void    CEnergyCheck::ThrowEnergyCheckDial(long closingButton)
{
    CDialog *theDialogStart = (CDialog*) itsWindow;

inherited::EndDialog(closingButton, FALSE); theDialogStart->Select(); theDialogStart->Close();
windowItemNum--;    }

void    CEnergyCheck::EnergyCheckDial(short DialogueId)
{
    CDLOGDirector *theInfo; CDialog *aDialog = (CDialog*) itsWindow;

theInfo = new (CDLOGDirector); theInfo->IDLOGDirector(DialogueId, this); theInfo->
DoModalDialog(cmdOK); theInfo->EndDialog(cmdOK, TRUE); theInfo->Close(TRUE); aDialog->
Update();    }

void    CEnergyCheck::Activate(void)
{
    CLoopDoc *theWindowChanging = NULL;

inherited::Activate(); theWindowChanging = new (CLoopDoc); theWindowChanging->
ChangingMenus();    }

void    CEnergyCheck::DoCommand(long theCommand)
{
    CStartDirPanel *theStartDir; CEquations *theEquations; short theModifiers;
    Str255 theTitle;

switch (theCommand)
{
case QuitCheckEnEq: theEnergyCheck->CloseDialEnergyCheck(); theEnergyCheck->
ThrowEnergyCheckDial(QuitCheckEnEq); gBartender->RemoveMenu(MENUEnEqCheck);
theEnergyCheck = NULL; ForgetObject(theEnergyCheck); break;
case CheckPosOne: PerformCheckSlots(); break;
case cmdEquateTot:
if (Totals) { EquateTotalsFunction();    }
else
{
    ParamText("\pThe energy expressions have not been verified by the check.", "\pPerform a check to
verify the correctness, then click the equate button.", "\pCHECK NOT PERFORMED",
"\pVERIFY EXPRESSIONS"); PositionDialog('ALRT', 128); InitCursor; StopAlert(128, NULL);
return;    } break;
case cmdWriteEnEq: EnergyCheckDial(WriteEnergyEq); break;
case cmdSubstQuant: EnergyCheckDial(SubstQuant); break;
case cmdSolutionHelp: EnergyCheckDial(StartCheckEnEqDial); break;
case cmdPosition1: whichPos1 = 0; break;
case cmdPosition2: whichPos2 = 0; break;
case cmdPosStart1: whichPos1 = 1; break;
case cmdPosBot1: whichPos1 = 2; break;
case cmdPosTop1: whichPos1 = 3; break;
case cmdPosIncline1: whichPos1 = 4; break;
case cmdPosLoop1: whichPos1 = 5; break;
case cmdPosStart2: whichPos2 = 1; break;
case cmdPosBot2: hichPos2 = 2; break;
case cmdPosTop2: whichPos2 = 3; break;
case cmdgbtn: case cmdhbtn: case cmdlbtn: case cmdMbtn: case cmdrbtn: case cmdRbtn: case
cmdybtn: case cmdEqualbtn: case cmdMultbtn: case cmdDividebtn: case cmdAddbtn: case
cmdSubbtn: case cmdSqRtbtn: case cmdReturnbtn: case cmdAnyVel: case cmdTopVel: case
cmdBottomVel: case cmdStartVel: case cmdAnyAngVel: case cmdTopAngVel: case cmdBottomAngVel:
theStartDir = new (CStartDirPanel); theStartDir->PlacetheChar(theCommand); break;
case cmdSquarebtn: theStartDir = new (CStartDirPanel); theStartDir->Squaring(); break;
case cmdZero: case cmdOne: case cmdTwo: case cmdThree: case cmdFour: case cmdFive: case
cmdSix: case cmdSeven: case cmdEight: case cmdNine: gBartender->GetCmdText(theCommand,
theTitle); ((CEditText*) gGopher)->TypeChar(theTitle[1], theModifiers); break;
default: inherited::DoCommand(theCommand);    }
}

void    CEnergyCheck::CloseDialEnergyCheck( void)

```

```

{
    Str255 theTitle; short z = 0; CWindow *theWindow;

itsWindow->GetTitle(theTitle); z = gBartender->FindItemText( MENUWindow, theTitle);
gBartender->RemoveMenuCmd(gBartender->FindCmdNumber( MENUWindow, z));
for (z = z; z < windowItemNum; z++)
    *(otherWindDial + z) = *(otherWindDial + z + 1);
*(otherWindDial + z) = NULL; ForgetPtr(*(otherWindDial + z));    }

void    CEnergyCheck::EnergyCheckModal(Str255 theText0, Str255 theText1, Str255 theText2,
                                        Str255 theText3, short IdNum)
{
    Good = FALSE; ParamText(theText0, theText1, theText2, theText3); PositionDialog('ALRT',
    IdNum); InitCursor; NoteAlert(IdNum, NULL); return;    }

void    CEnergyCheck::PerformCheckSlots(void)
{
    int resWhichPos1; short theModifiers; CRadioGroupPane *radioGroupSet;

radioGroupSet = (CRadioGroupPane *)itsWindow->FindViewByID(kRadioEnEqCheckID);
switch (radioGroupSet -> GetStationID())
{
    case kSlidingEnEqCheck: Motion = 1; break;
    case kRollingEnEqCheck: Motion = 2; break;
    default: break;    }
    if (!whichPos1 && !whichPos2) {
        EnergyCheckDial(ChoosePos1Pos2); return;    }
    if (whichPos1 == whichPos2 && whichPos1) {
        EnergyCheckModal("\pThe two positions are equal.", "\pChange one of them and the corresponding
        energy expressions", "\pPOSITION 1 = POSITION 2", "\pCHANGE CHOICE", 128); return;    }
    MakeEditWindCurrent(kEqualTot); ((CEditText*) gGopher->SetSelection(0, ((CEditText*)
    gGopher) -> GetLength(), TRUE); ((CEditText*) gGopher->SetTextString("\p"); Good = TRUE;
    Totals = FALSE; textBox = 0;
    if (whichPos1 != 0) {
        MakeEditWindCurrent(kPos1Tot); ((CEditText*) gGopher->SetSelection(0, ((CEditText*)
        gGopher->GetLength(), TRUE); ((CEditText*) gGopher->SetTextString("\p");
        CheckSlidEnergiesSet();
    if (!Good) {
        GetEditWind(kPos1Tot); ((CEditText*) gGopher->SetSelection(0, ((CEditText*) gGopher->
        GetLength(), TRUE); ((CEditText*) gGopher->SetTextString("\p"); return;    }
    else {
        GetEditWind(kPos1Tot); Length = ((CEditText*) gGopher->GetLength()); ((CEditText*)
        gGopher->SetSelection(Length - 2, Length, TRUE); ((CEditText*) gGopher->TypeChar(' ',
        theModifiers);
        Totals = TRUE;    }
    }
    if (whichPos2 != 0) {
        resWhichPos1 = whichPos1; whichPos1 = 0; MakeEditWindCurrent(kPos2Tot); ((CEditText*)
        gGopher->SetSelection(0, ((CEditText*) gGopher->GetLength(), TRUE); ((CEditText*)
        gGopher)
        ->SetTextString("\p"); CheckSlidEnergiesSet();
    if (!Good) {
        GetEditWind(kPos2Tot); ((CEditText*) gGopher->SetSelection(0, ((CEditText*) gGopher->
        GetLength(), TRUE); ((CEditText*) gGopher->SetTextString("\p"); whichPos1 = resWhichPos1;
        Totals = FALSE;    }
    else {
        GetEditWind(kPos2Tot); Length = ((CEditText*) gGopher->GetLength()); ((CEditText*)
        gGopher)
        ->SetSelection(Length - 2, Length, TRUE); ((CEditText*) gGopher->TypeChar(' ', theModifiers);
        whichPos1 = resWhichPos1; Totals = TRUE;    }
    }
    if (Totals == TRUE && whichPos1 && whichPos2) {
        EnergyCheckModal("\pCHECK COMPLETED", "\pThe energies may be equated by clicking the
        \pEquate Totals" button.", "\pENERGY EXPRESSION", "\pAT TWO DIFFERENT POINTS",
        128); return;    }
    }

void    CEnergyCheck::CheckSlidEnergiesSet(void)
{
    CEquations *theEquations;

theEquations = new (CEquations);

```

```

switch (whichPos1)
{
case 1: GetEditWind(kPos1TKE); theEquations->GetStartSlidTKE();
if (!Good) return;
FillTotEnergyCheck(kPos1Tot); whichPos12 = 1;
if (Motion == 1) {
    GetEditWind(kPos1RKE); theEquations->GetSlidRKE();
    if (!Good) return; }
else {
    GetEditWind(kPos1RKE); theEquations->GetStartRKE();
    if (!Good) return; }
FillTotEnergyCheck(kPos1Tot); GetEditWind(kPos1GPE); theEquations->GetStartSlidGPE();
if (!Good) return;
else {
    FillTotEnergyCheck(kPos1Tot); EnergyCheckModal("\pCHECK COMPLETED", "\pThe
    translational kinetic energy, rotational kinetic energy and the gravitational potential energy are
    correct", "\pPOSITION 1", "\pAT THE START", 128); Good = TRUE; return; } break;
case 2: GetEditWind(kPos1TKE); theEquations->GetBotTKE();
if (!Good) return;
if (Motion == 1)
    FillTotEnergyCheck(kPos1Tot); whichPos12 = 2;
if (Motion == 1) {
    GetEditWind(kPos1RKE); theEquations->GetSlidRKE();
    if (!Good) return; }
else {
    GetEditWind(kPos1RKE); theEquations->GetBotRKE();
    if (!Good) return; AddRKEandTKE(-125); } FillTotEnergyCheck(kPos1Tot); GetEditWind(
    kPos1GPE); theEquations->GetBotGPE();
    if (!Good) return;
    else {
        FillTotEnergyCheck(kPos1Tot); EnergyCheckModal("\pCHECK COMPLETED", "\pThe
        translational kinetic energy, rotational kinetic energy and the gravitational potential energy are
        correct", "\pPOSITION 1", "\pAT THE BOTTOM OF THE LOOP", 128); Good = TRUE; return; }
break;
case 3: GetEditWind(kPos1TKE); theEquations->GetTopTKE();
if (!Good) return;
if (Motion == 1) FillTotEnergyCheck(kPos1Tot); whichPos12 = 3;
if (Motion == 1) {
    GetEditWind(kPos1RKE); theEquations->GetSlidRKE();
    if (!Good) return; }
else {
    GetEditWind(kPos1RKE); theEquations->GetTopRKE();
    if (!Good) return; AddRKEandTKE(-127); } FillTotEnergyCheck(kPos1Tot); GetEditWind(
    kPos1GPE); theEquations->GetTopGPE();
    if (!Good) return;
    else {
        FillTotEnergyCheck(kPos1Tot); EnergyCheckModal("\pCHECK COMPLETED", "\pThe
        translational kinetic energy, rotational kinetic energy and the gravitational potential energy are
        correct", "\pPOSITION 1", "\pAT THE TOP OF THE LOOP", 128); Good = TRUE; return; }
break;
case 4: GetEditWind(kPos1TKE); theEquations->GetInclineTKE();
if (!Good) return; if (Motion == 1) FillTotEnergyCheck(kPos1Tot); whichPos12 = 4;
if (Motion == 1) {
    GetEditWind(kPos1RKE); theEquations->GetSlidRKE();
    if (!Good) return; }
else {
    GetEditWind(kPos1RKE); theEquations->GetInclineRKE();
    if (!Good) return; AddRKEandTKE(-124); }
FillTotEnergyCheck(kPos1Tot); GetEditWind(kPos1GPE); theEquations->GetInclineGPE();
if (!Good) return;
else {
    FillTotEnergyCheck(kPos1Tot); EnergyCheckModal("\pCHECK COMPLETED", "\pThe
    translational kinetic energy, rotational kinetic energy and the gravitational potential energy are
    correct", "\pPOSITION 1", "\pALONG THE INCLINE", 128); Good = TRUE; return; }
break;
case 5: GetEditWind(kPos1TKE); theEquations->GetLoopTKE();
if (!Good) return; if (Motion == 1) FillTotEnergyCheck(kPos1Tot); whichPos12 = 5;
if (Motion == 1) {
    GetEditWind(kPos1RKE); theEquations->GetSlidRKE(); if (!Good) return; }

```

```

else {
    GetEditWind(kPos1RKE); theEquations->GetInclineRKE(); if (!Good) return; AddRKEandTKE(-
125); }
FillTotEnergyCheck(kPos1Tot); GetEditWind( kPos1GPE); theEquations->GetInclineGPE();
if (!Good) return;
else {
    FillTotEnergyCheck(kPos1Tot); EnergyCheckModal("\pCHECK COMPLETED", "\pThe
translational kinetic energy, rotational kinetic energy and the gravitational potential energy are
correct", "\pPOSITION 1", "\pAT A POINT IN THE LOOP", 128); Good = TRUE; return; }
break;
}
switch (whichPos2)
{
case 1: GetEditWind(kPos2TKE); theEquations->GetStartSlidTKE(); if (!Good) return;
FillTotEnergyCheck(kPos2Tot); whichPos12 = 1;
if (Motion == 1) {
    GetEditWind(kPos2RKE); theEquations->GetSlidRKE(); if (!Good) return; }
else {
    GetEditWind( kPos2RKE); theEquations->GetStartRKE(); if (!Good) return; }
FillTotEnergyCheck(kPos2Tot); GetEditWind( kPos2GPE); theEquations->GetStartSlidGPE();
if (!Good) return;
else {
    FillTotEnergyCheck(kPos2Tot); EnergyCheckModal("\pCHECK COMPLETED", "\pThe
translational kinetic energy, rotational kinetic energy and the gravitational potential energy are
correct", "\pPOSITION 2", "\pAT THE START", 128); Good = TRUE; return; } break;
case 2: GetEditWind( kPos2TKE); theEquations->GetBotTKE(); if (!Good) return; if (Motion ==
1)
FillTotEnergyCheck(kPos2Tot); whichPos12 = 2;
if (Motion == 1) {
    GetEditWind(kPos1RKE); theEquations->GetSlidRKE(); if (!Good) return; }
else {
    GetEditWind(kPos2RKE); theEquations->GetBotRKE(); if (!Good) return; AddRKEandTKE(-
125); }
FillTotEnergyCheck(kPos2Tot); GetEditWind( kPos2GPE); theEquations->GetBotGPE();
if (!Good) return;
else {
    FillTotEnergyCheck(kPos2Tot); EnergyCheckModal("\pCHECK COMPLETED", "\pThe
translational kinetic energy, rotational kinetic energy and the gravitational potential energy are
correct", "\pPOSITION 2", "\pAT THE BOTTOM OF THE LOOP", 128); Good = TRUE; return; }
break;
case 3: GetEditWind( kPos2TKE); theEquations->GetTopTKE(); if (!Good) return; if (Motion ==
1) FillTotEnergyCheck(kPos2Tot); whichPos12 = 2;
if (Motion == 1) {
    GetEditWind(kPos2RKE); theEquations->GetSlidRKE(); if (!Good) return; }
else {
    GetEditWind(kPos2RKE); theEquations->GetTopRKE(); if (!Good) return; AddRKEandTKE(-
127); }
FillTotEnergyCheck(kPos2Tot); GetEditWind( kPos2GPE); theEquations->GetTopGPE();
if (!Good) return;
if (Good) {
    FillTotEnergyCheck(kPos2Tot); EnergyCheckModal("\pCHECK COMPLETED", "\pThe
translational kinetic energy, rotational kinetic energy and the gravitational potential energy are
correct", "\pPOSITION 2", "\pAT THE TOP OF THE LOOP", 128); Good = TRUE; return; }
break; }
}

void    CEnergyCheck::MakeEditWindCurrent(short dialogItemNo)
{
    CDialog *theDialog = (CDialog*) itsWindow; tGopherInfo gopherInfo; CView
    *newGopher;
do {
    gopherInfo.currentGopher = (CView*) gGopher; theDialog->FindGophers(&gopherInfo); newGopher
    = gopherInfo.nextGopher; newGopher->BecomeGopher(TRUE); }
while(newGopher->ID != dialogItemNo);
}

void    CEnergyCheck::GetEditWind(short dialogItemNo)
{

```

```

MakeEditWindCurrent(dialogItemNo); Length = ((CEditText*) gGopher)->GetLength();
toPrevEntry = ((CEditText*) gGopher)->CopyTextRange(0, ((CEditText*) gGopher)->
GetLength());
}

void CEnergyCheck::AddRKEandTKE(char velAbbr)
{
    *(*toPrevEntry) = '7'; *(*toPrevEntry + 1) = '/'; *(*toPrevEntry + 2) = '1'; *(*toPrevEntry + 3) =
    '0'; *(*toPrevEntry + 4) = ' '; *(*toPrevEntry + 5) = 'M'; *(*toPrevEntry + 6) = velAbbr;
}

void CEnergyCheck::FillTotEnergyCheck(short dialogItemNo)
{
    short theModifiers;

    if ((*toPrevEntry) == 0 || *(*toPrevEntry) == '0') return; textBox++;
    MakeEditWindCurrent(dialogItemNo);
    if ( textBox == 1) {
        ((CAbstractText*) gGopher)->SetTextString("\p"); ((CEditText*) gGopher)->SetSelection(0, 0,
        TRUE); ((CEditText*) gGopher)->InsertTextPtr( *toPrevEntry, Length, TRUE); }
    else
        ((CEditText*) gGopher)->InsertTextPtr( *toPrevEntry, Length, TRUE);

    ((CEditText*) gGopher)->TypeChar(' ', theModifiers); ((CEditText*) gGopher)->TypeChar('+',
    theModifiers); ((CEditText*) gGopher)->TypeChar(' ', theModifiers); Length = ((CEditText*)
    gGopher)->GetLength(); ((CEditText*) gGopher)->SetSelection(Length, Length, TRUE);
}

void CEnergyCheck::EquateTotalsFunction(void)
{
    short theModifiers;

    MakeEditWindCurrent(kEqualTot); ((CEditText*) gGopher)->SetSelection(0, ((CEditText*)
    gGopher)->GetLength(), TRUE); ((CEditText*) gGopher)->SetTextString("\p");
    if ((whichPos1 == 1 && whichPos2 == 2) || (whichPos1 == 2 && whichPos2 == 1))
        if ( Motion == 1) {
            MakeEditWindCurrent(kEqualTot); ((CAbstractText*) gGopher)->SetTextString("\p = 2gh");
            ((CEditText*) gGopher)->SetSelection(0, 0, TRUE); ((CEditText*) gGopher)->TypeChar(-
            125, theModifiers); }
        else {
            MakeEditWindCurrent(kEqualTot); ((CAbstractText*) gGopher)->SetTextString("\p = 10/7
            gh"); ((CEditText*) gGopher)->SetSelection(0, 0, TRUE); ((CEditText*) gGopher)->
            TypeChar(-125, theModifiers); }
    if ((whichPos1 == 1 && whichPos2 == 3) || (whichPos1 == 3 && whichPos2 == 1)) {
        if ( Motion == 1) {
            MakeEditWindCurrent(kEqualTot); ((CAbstractText*) gGopher)->SetTextString("\p =
            2g(h - 2R)"); ((CEditText*) gGopher)->SetSelection(0, 0, TRUE); ((CEditText*)
            gGopher)-> TypeChar(-127, theModifiers); }
        else {
            MakeEditWindCurrent(kEqualTot); ((CAbstractText*) gGopher)->SetTextString("\p = 10/7
            g(h - 2R)"); ((CEditText*) gGopher)->SetSelection(0, 0, TRUE); ((CEditText*)
            gGopher)-> TypeChar(-127, theModifiers); }
        EnergyCheckModal("\pTo obtain the solution to the problem:", "\pObtain the force
        expression and equate the two expressions.", "\pENERGY EXPRESSION EQUATED",
        "\pAT START AND TOP OF THE LOOP", 128); return; }
    if ((whichPos1 == 2 && whichPos2 == 3) || (whichPos1 == 3 && whichPos2 == 2))
        if ( Motion == 1) {
            MakeEditWindCurrent(kEqualTot); ((CAbstractText*) gGopher)->SetTextString("\p =
            4gR"); ((CEditText*) gGopher)->SetSelection(0, 0, TRUE); ((CEditText*) gGopher)->
            TypeChar(-125, theModifiers); ((CEditText*) gGopher)->SetSelection(4, 4, TRUE);
            ((CEditText*) gGopher)->TypeChar(-127, theModifiers); }
        else {
            MakeEditWindCurrent(kEqualTot); ((CAbstractText*) gGopher)->SetTextString("\p =
            20/7
            gR"); ((CEditText*) gGopher)->SetSelection(0, 0, TRUE); ((CEditText*) gGopher)->
            TypeChar(-125, theModifiers); ((CEditText*) gGopher)->SetSelection(4, 4, TRUE);
            ((CEditText*) gGopher)->TypeChar(-127, theModifiers); }
    if ((whichPos1 == 4 || whichPos1 == 5) && whichPos2 == 1)
        if ( Motion == 1) {

```

```

        MakeEditWindCurrent(kEqualTot); ((CAbstractText*) gGopher)->SetTextString("\p =
        2g(h - y)"); ((CEditText*) gGopher)->SetSelection(0, 0, TRUE); ((CEditText*)
        gGopher)-> TypeChar(-124, theModifiers); }
    else
    {
        MakeEditWindCurrent(kEqualTot); ((CAbstractText*) gGopher)->SetTextString("\p = 10/7
        g(h - y)"); ((CEditText*) gGopher)->SetSelection(0, 0, TRUE); ((CEditText*) gGopher)->
        TypeChar(-124, theModifiers); }
    if ((whichPos1 == 4 || whichPos1 == 5) && whichPos2 == 3)
    if ( Motion == 1)
    {
        MakeEditWindCurrent(kEqualTot); ((CAbstractText*) gGopher)->SetTextString("\p - =
        2g(2R - y)"); ((CEditText*) gGopher)->SetSelection(0, 0, TRUE); ((CEditText*)
        gGopher) ->TypeChar(-124, theModifiers); ((CEditText*) gGopher)->SetSelection(4, 4,
        TRUE); ((CEditText*) gGopher)->TypeChar(-127, theModifiers); }
    else
    {
        MakeEditWindCurrent(kEqualTot); ((CAbstractText*) gGopher)->SetTextString("\p - =
        10/7 g(2R - y)"); ((CEditText*) gGopher)->SetSelection(0, 0, TRUE); ((CEditText*)
        gGopher)-> TypeChar(-124, theModifiers); ((CEditText*) gGopher)->SetSelection(4, 4,
        TRUE); ((CEditText*) gGopher)->TypeChar(-127, theModifiers); }
    if ((whichPos1 == 4 || whichPos1 == 5) && whichPos2 == 2)
    if ( Motion == 1)
    {
        MakeEditWindCurrent(kEqualTot); ((CAbstractText*) gGopher)->SetTextString("\p - =
        2gy"); ((CEditText*) gGopher)->SetSelection(0, 0, TRUE); ((CEditText*) gGopher)->
        TypeChar(-125, theModifiers); ((CEditText*) gGopher)->SetSelection(4, 4, TRUE);
        ((CEditText*) gGopher)->TypeChar(-124, theModifiers); }
    else
    {
        MakeEditWindCurrent(kEqualTot); ((CAbstractText*) gGopher)->SetTextString("\p - =
        10/7 gy"); ((CEditText*) gGopher)->SetSelection(0, 0, TRUE); ((CEditText*)
        gGopher)-> TypeChar(-125, theModifiers); ((CEditText*) gGopher)->SetSelection(4, 4,
        TRUE); ((CEditText*) gGopher)->TypeChar(-124, theModifiers); }
}

```

CEnergyEquations.c

(The director of the energy equations window)

```

#include the following: "Commands.h", "CEnergyEquations.h", "Global.h", "AppCommands.h",
<CBartender.h>, <CDialog.h>, "CLoopDoc.h", "CLoopApp.h", <CDesktop.h>, <TBUilities.h>,
"CStartDirPanel.h", <CEditText.h>, <CRadioGroupPane.h>, "CEquations.h"

```

```
enum
```

```
{
    kRadioEnEqID = 19, kSlidingEnEq = 27, kRollingEnEq
};
```

```
CEnergyEquations *theEnergyEq; extern CWindow **otherWindDial; extern CBartender
*gBartender; extern CBureaucrat *gGopher; extern int windowItemNum; int ChStart;
```

```
void    CEnergyEquations::IEnergyEquations(short DLOGid, CDirectorOwner *aSupervisor)
{
    CDLOGDirector::IDLOGDirector( DLOGid, aSupervisor);
}
```

```
void    CEnergyEquations::BegDialEnergy(long    theCommand)
{
    CDialog *theDialog = (CDialog*) itsWindow; CRadioGroupPane *radioGroupSet;
```

```
theDialog->Select(); inherited::BeginDialog(); radioGroupSet = (CRadioGroupPane *)itsWindow->
FindViewById(kRadioEnEqID); radioGroupSet -> SetStationID(kSlidingEnEq); ChStart = 1;
}
```

```
void    CEnergyEquations::ThrowEnergyDial(long closingButton)
{
    CDialog *theDialogStart = (CDialog*) itsWindow;
```

```
inherited::EndDialog(closingButton, FALSE); theDialogStart->Select(); theDialogStart->Close();
windowItemNum--;
```

```
void    CEnergyEquations::Activate(void)
{
    CLoopDoc *theWindowChanging = NULL;
```

```
inherited::Activate(); theWindowChanging = new (CLoopDoc); theWindowChanging->
ChangingMenus();
}
```

```
void    CEnergyEquations::DoCommand(long theCommand)
{
    CStartDirPanel *theStartDir; CEquations *theEquations; Str255 theTitle; short
theModifiers;
```

```

switch (theCommand) {
case cmdCHECKbtn: WhichEnergyRadBut(); break;
case cmdChStart: ChStart = 1; break;
case cmdChBot: ChStart = 2; break;
case cmdChTop: ChStart = 3; break;
case cmdChStartTop: ChStart = 4; break;
case cmdChStartBot: ChStart = 5; break;
case cmdChTopBot: ChStart = 6; break;
case cmdQUITbtn: theEnergyEq->CloseDialEnergy(); theEnergyEq->
ThrowEnergyDial(cmdQUITbtn); gBartender->RemoveMenu(MENUEnergyEq); theEnergyEq =
NULL; ForgetObject(theEnergyEq); break;
case ClarEnEqStart: EnergyEquationsDial(ClarEnEqStartDial); break;
case ClarEnEqBot: EnergyEquationsDial(ClarEnEqBotDial); break;
case ClarEnEqTop: EnergyEquationsDial(ClarEnEqTopDial); break;
case ClarBotEnEqRotbtn: EnergyEquationsDial(ClarBotEnEqRot); break;
case ClarTopEnEqRotbtn: EnergyEquationsDial(ClarTopEnEqRot); break;
case cmdStartEnEq: EnergyEquationsDial(SolutionHelp); break;
case cmdWriteEnEq: EnergyEquationsDial(WriteEnergyEq); break;
case cmdSubstQuant: EnergyEquationsDial(SubstQuant); break;
case cmdZero: case cmdOne: case cmdTwo: case cmdThree: case cmdFour: case cmdFive: case
cmdSix: case cmdSeven: case cmdEight: case cmdNine: gBartender->GetCmdText(theCommand,
theTitle); ((CEditText*) gGopher)->TypeChar(theTitle[1], theModifiers); break;
case cmdgbtn: case cmdhbtn: case cmdlbtn: case cmdMbtn: case cmdrbtn: case cmdRbtn: case
cmdybbtn: case cmdOpenParenbtn: case cmdCloseParenbtn: case cmdEqualbtn: case cmdMultbtn: case
cmdDividebtn: case cmdAddbtn: case cmdSubtbtn: case cmdSqRtbtn: case cmdReturnbtn: case
cmdAnyVel: case cmdTopVel: case cmdBottomVel: case cmdStartVel: case cmdAnyAngVel: case
cmdTopAngVel: case cmdBottomAngVel: theStartDir = new (CStartDirPanel); theStartDir->
PlacetheChar(theCommand); break;
case cmdSquarebtn: theStartDir = new (CStartDirPanel); theStartDir->Squaring(); break;
default: inherited::DoCommand(theCommand); }
}

```

```

void CEnergyEquations::CloseDialEnergy( void)
{ Str255 theTitle; short z = 0; CWindow *theWindow;

```

```

itsWindow->GetTitle(theTitle); z = gBartender->FindItemText( MENUWindow, theTitle);
gBartender
->RemoveMenuCmd(gBartender->FindCmdNumber( MENUWindow, z));
for (z = z; z < windowItemNum; z++)
*(otherWindDial + z) = *(otherWindDial + z + 1); *(otherWindDial + z) = NULL;
ForgetPtr(*(otherWindDial + z));
}

```

```

void CEnergyEquations::WhichEnergyRadBut(void)
{
switch (ChStart) {
case 1: SlidRollAnsDial(AnsStartEnEq, AnsStartEnEq); break;
case 2: SlidRollAnsDial(AnsBottomEnEq, BotEnEqRot); break;
case 3: SlidRollAnsDial(AnsTopEnEq, TopEnEqRot); break;
case 4: SlidRollAnsDial(AnsStartTopEnEq, StartTopEnEqRot); break;
case 5: SlidRollAnsDial(AnsStartBotEnEq, StartBotEnEqRot); break;
case 6: SlidRollAnsDial(AnsTopBotEnEq, BotTopEnEqRot); break;
default: break; }
}

```

```

void CEnergyEquations::SlidRollAnsDial(short dialogID1, short dialogID2)
{ CRadioGroupPane *radioGroupSet;

```

```

radioGroupSet = (CRadioGroupPane *)itsWindow->FindViewByID(kRadioEnEqID);
switch (radioGroupSet -> GetStationID()) {
case kSlidingEnEq: EnergyEquationsDial(dialogID1); break;
case kRollingEnEq: EnergyEquationsDial(dialogID2); break;
default: break; }
}

```

```

void CEnergyEquations::EnergyEquationsDial(short DialogueId)
{ CDLOGDirector *theInfo; CDialog *aDialog = (CDialog*) itsWindow;

```

```
theInfo = new (CDLOGDirector); theInfo->IDLOGDirector(DialogueId, this); theInfo->
DoModalDialog(cmdOK); theInfo->EndDialog(cmdOK, TRUE); theInfo->Close(TRUE); aDialog->
Update();}
```

CEquations.c

(This file includes the methods which perform the energy checks)

```
#include the following: "Commands.h", "CEquations.h", "Global.h", "AppCommands.h",
<CBartender.h>, "CLoopDoc.h", "CLoopApp.h", <CDesktop.h>, <CEditText.h>, <TBUtilities.h>
```

```
extern CBartender *gBartender; extern CBureaucrat *gGopher; extern CDesktop *gDesktop; extern
Handle toPrevEntry; extern short num, den; extern long Length; extern Boolean Good, Totals;
extern int whichPos12;
```

```
static void GetFraction(void);
```

```
void CEquations::SineCosinePlace(char first, char second, char third)
{ short theModifiers;
```

```
((CEditText*) gGopher)->TypeChar(first, theModifiers); ((CEditText*) gGopher)->
TypeChar(second, theModifiers); ((CEditText*) gGopher)->TypeChar(third, theModifiers); }
```

```
void CEquations::ModalDialPlace(Str255 theText0, Str255 theText1, Str255 theText2, Str255
theText3, short IdNum)
```

```
{
Good = FALSE; ParamText(theText0, theText1, theText2, theText3); PositionDialog('ALRT',
IdNum); InitCursor; NoteAlert(IdNum, NULL); return; }
```

```
void CEquations::GetStartSlidTKE(void)
{ int i = 0, velval = 0, mValue = 0;
```

```
if ((*toPrevEntry) == 0 || (*toPrevEntry) == '0') {
((CEditText*) gGopher)->SetTextString("\p0"); return; }
```

```
GetFraction();
```

```
if ((num == 1 || num == 0) && (den == 1 || den == 2))
;
```

```
else {
ModalDialPlace("\pThe velocity of the ball is zero.", "\pThis gives a zero translational kinetic
energy of the ball.", "\pTranslational Kinetic Energy", "\pAT THE START", 128); return; }
for ( i = 0; i <= Length - 1; i++) {
switch ((*toPrevEntry + i)) {
case '0': case '1': case '2': case '3': case '4': case '5': case '6': case '7': case '8': case '9': case '/': case ' ':
case '*': break;
case 'M': mValue++; break;
case 'm': ModalDialPlace("\pCapital M is the mass of the ball.", "\pReplace m by M.",
"\pTranslational Kinetic Energy", "\pAT THE START", 128); return; break;
case -110: velval++; break;
case -125: case -127: case -124: Good = FALSE; PositionDialog('ALRT', WrongVelSq); InitCursor;
NoteAlert(WrongVelSq, NULL); return; break;
default: ModalDialPlace("\pThe velocity of the ball is zero.", "\pThis gives zero translational kinetic
energy of the ball.", "\pTranslational Kinetic Energy", "\pAT THE START", 128); return; break; }
}
```

```
if (mValue == 1 && velval == 1) {
ModalDialPlace("\pThe expression for the translational kinetic energy is correct.", "\pThe velocity
of the ball is zero. \pThis gives a zero translational kinetic energy of the ball.", "\pTranslational
Kinetic Energy", "\pAT THE START", 128); Good = TRUE; return; }
```

```
else {
ModalDialPlace("\pThe velocity of the ball is zero.", "\pThis gives a zero translational kinetic
energy of the ball.", "\pTranslational Kinetic Energy", "\pAT THE START", 128); return; }
```

```
void CEquations::GetSlidRKE(void)
{ Str255 theText;
```

```
if ((*toPrevEntry) == 0 || (*toPrevEntry) == '0') {
((CEditText*) gGopher)->SetTextString("\p0"); return; }
```



```

else
{
switch (whichPos12 ) {
case 1: ModalDialPlace("\pFor a sliding ball:", "\pThere is no rotational kinetic energy involved.",
"\pRotational Kinetic Energy", "\pAT THE START", 128); return; break;
case 2: ModalDialPlace("\pFor a sliding ball:", "\pThere is no rotational kinetic energy involved.",
"\pRotational Kinetic Energy", "\pAT THE BOTTOM OF THE LOOP", 128); return; break;
case 3: ModalDialPlace("\pFor a sliding ball:", "\pThere is no rotational kinetic energy involved.",
"\pRotational Kinetic Energy", "\pAT THE TOP OF THE LOOP", 128); return; break;
case 4: ModalDialPlace("\pFor a sliding ball:", "\pThere is no rotational kinetic energy involved.",
"\pRotational Kinetic Energy", "\pALONG THE INCLINE", 128); return; break;
case 5: ModalDialPlace("\pFor a sliding ball:", "\pThere is no rotational kinetic energy involved.",
"\pRotational Kinetic Energy", "\pAT A POINT IN THE LOOP", 128); return; break; }
}
}

void CEquations::GetStartRKE(void)
{
if ((*toPrevEntry) == 0 || (*toPrevEntry) == '0') {
((CEditText*) gGopher)->SetTextString("\p0"); return; }
else {
ModalDialPlace("\pThe velocity of the ball is zero.", "\pThis gives a zero rotational kinetic energy
of the ball.", "\pRotational Kinetic Energy", "\pAT THE START", 128); return; }
}

void CEquations::GetStartSlidGPE(void)
{
int i = 0, mValue = 0, gValue = 0, hValue = 0;

if ((*toPrevEntry) == 0 ) {
ModalDialPlace("\pYou have not given an expression for the gravitational potential energy of the
ball.", "\pProvide a value before checking.\rClick on \"Need help with\" for guidance.",
"\pGravitational Potential Energy", "\pAT THE START", 128); return; }
if ( (*toPrevEntry) == '0') {
ModalDialPlace("\pThe gravitational potential energy of the ball is not 0.", "\pNOTE: Assuming
zero reference level at the lowest position of the ball.\rClick \"Need help with\" for guidance.",
"\pGravitational Potential Energy", "\pAT THE START", 128); return; }
GetFraction();
if (num == 2)
for ( i = 0; i <= Length - 1; i++)
if ((*toPrevEntry + i) == 'R') {
ModalDialPlace("\pThe height at the start is not 2R.", "\pReplace 2R by h, the initial
height of the ball.", "\pGravitational Potential Energy", "\pAT THE START", 128); return; }
if ((num == 1 || num == 0) && den == 1)
;
else {
ModalDialPlace("\pThe gravitational potential energy expression is incorrect.", "\pNOTE:
Assuming zero reference level at the lowest position of the ball.\rh is the starting height of the
ball.\rClick \"Need help with\" for guidance.", "\pGravitational Potential Energy", "\pAT THE
START", 128); return; }
for ( i = 0; i <= Length - 1; i++) {
switch ((*toPrevEntry + i)) {
case 'M': mValue++; break;
case 'g': gValue++; break;
case 'h': hValue++; break;
case '0': case '1': case '2': case '3': case '4': case '5': case '6': case '7': case '8': case '9': case '/': case ' ':
case '*': break;
case 'm': ModalDialPlace("\pCapital M is the mass of the ball.", "\pReplace m by M.",
"\pGravitational Potential Energy", "\pAT THE START", 128); return; break;
case 'R': case 'r': ModalDialPlace("\ph is the starting height of the ball.", "\pReplace R by h.",
"\pGravitational Potential Energy", "\pAT THE START", 128); return; break;
case 'y': ModalDialPlace("\pAt the start: y = h", "\pwhere h is the starting height of the ball.",
"\pGravitational Potential Energy", "\pAT THE START", 128); return; break;
default: ModalDialPlace("\pThe gravitational potential energy expression is wrong.", "\pNOTE:
Assuming zero reference level at the lowest position of the ball.\rClick \"Need help with\" for
expression.", "\pGravitational Potential Energy", "\pAT THE START", 128); return; break; }
}
if (mValue == 1 && gValue == 1 && hValue == 1) {
for (i = 0; i <= Length; i++)
*(*toPrevEntry + i) = 0; *(*toPrevEntry) = 'M'; *(*toPrevEntry + 1) = 'g'; *(*toPrevEntry + 2)
=

```

```

'h'; Length = 3; }
else {
    ModalDialPlace("\pThe gravitational potential energy of the ball is Mgh.", "\pNOTE: Assuming
zero reference level at the lowest position of the ball.", "\pGravitational Potential Energy", "\pAT
THE START", 128); return; }
}

void    CEquations::GetBotTKE(void)
{
    int i = 0, wB2Val = 0, mValue = 0, vB2Val = 0, R2Val = 0;

    if (*(toPrevEntry) == 0) {
        Good = FALSE; ModalDialPlace("\pYou have not given an expression for the translational kinetic
energy.", "\pProvide an expression before continuing check.\rClick \"Need help with\" for
expression.", "\pTranslational Kinetic Energy", "\pAT THE BOTTOM OF THE LOOP", 128);
        return; }
    if ( *(toPrevEntry) == '0') {
        Good = FALSE; ModalDialPlace("\pThe translational kinetic energy of the ball is not 0.",
"\pProvide an expression before continuing check.\rClick \"Need help with\" for expression.",
"\pTranslational Kinetic Energy", "\pAT THE BOTTOM OF THE LOOP", 128); return; }
    GetFraction();
    if ((num == 1 || num == 0) && (den == 2))
        ;
    else {
        Good = FALSE; ModalDialPlace("\pThe expression should have a 1/2.", "\pFor information, click
\"Need help with\".", "\pTranslational Kinetic Energy", "\pAT THE BOTTOM OF THE LOOP",
128); return; }
    for (i = 0; i <= Length - 1; i++) {
        switch (*(toPrevEntry + i)) {
            case 'M': mValue++; break;
            case -125: vB2Val++; break;
            case -120: wB2Val++; break;
            case -116: R2Val++; break;
            case '0': case '1': case '2': case '3': case '4': case '5': case '6': case '7': case '8': case '9': case '/': case ' ':
            case '*': break;
            case -127: case -124: case -110: case -128: case -126: case -109: case 'v': Good = FALSE;
            PositionDialog('ALRT', 1381); InitCursor; NoteAlert(1381, NULL); return; break;
            case -123: case -122: case -121: case -119: case -118: Good = FALSE; PositionDialog('ALRT',
1382); InitCursor; NoteAlert(1382, NULL); return; break;
            case 'm': ModalDialPlace("\pCapital M is the mass of the ball.", "\pReplace m by M.",
"\pTranslational Kinetic Energy", "\pAT THE BOTTOM OF THE LOOP", 128); return; break;
            default: ModalDialPlace("\p", "\pAT THE BOTTOM OF THE LOOP", "\pB", "\p1/2 Mv", 2000);
            return; break; }
        }
    if (wB2Val == 1 && R2Val == 1 && mValue == 1) {
        Good = FALSE; PositionDialog('ALRT', 2004); InitCursor; NoteAlert(2004, NULL); return; }
    else if (mValue == 1 && vB2Val == 1) {
        for (i = 0; i <= Length; i++)
            (*(toPrevEntry + i) = 0; *(toPrevEntry) = '1'; *(toPrevEntry + 1) = '/'; *(toPrevEntry + 2) =
            '2'; *(toPrevEntry + 3) = ' '; *(toPrevEntry + 4) = 'M'; *(toPrevEntry + 5) = -125; Length =
            6; }
        else {
            Good = FALSE; ModalDialPlace("\pNOTE: v = wR", "\pAT THE BOTTOM OF THE LOOP",
            "\pB", "\p1/2 Mv", 2000); return; }
        }

    void    CEquations::GetBotRKE(void)
    {
        int i = 0, l = 0, down = 0, vb = 0, M = 0, wb = 0, r = 0, fraction = 0;

        if (*(toPrevEntry) == 0) {
            Good = FALSE; ModalDialPlace("\pYou have not given an expression.", "\pProvide a value before
continuing check.\rClick \"Need help with\" for expression.", "\pRotational Kinetic Energy",
            "\pAT THE BOTTOM OF THE LOOP", 128); return; }
        if ( *(toPrevEntry) == '0') {
            Good = FALSE; ModalDialPlace("\pThe rotational kinetic energy of the ball is not 0.", "\pProvide
a value before continuing check.\rClick \"Need help with\" for expression.", "\pRotational Kinetic
Energy", "\pAT THE BOTTOM OF THE LOOP", 128); return; }
        GetFraction();
        if ((num == 1 || num == 0) && den == 2)
            fraction = 1;
    }

```

```

else if (den == 5 && (num == 1 || num == 0))
    fraction = 2;
else if (num == 2 && den == 10) {
    Good = FALSE; ModalDialogPlace("\pSimplify expression.", "\pSimplify the fraction to 1/5.",
    "\pRotational Kinetic Energy", "\pAT THE BOTTOM OF THE LOOP", 128); return; }
else {
    Good = FALSE; ModalDialogPlace("\pFraction incorrect.", "\pSimplify the fraction or correct it.",
    "\pRotational Kinetic Energy", "\pAT THE BOTTOM OF THE LOOP", 128); return; }
for ( i = 0; i <= Length - 1; i++) {
    switch (*(toPrevEntry + i)) {
    case 'M': M = 1; break;
    case -120: wb = 1; break;
    case 'I': I = 1; break;
    case -117: r = 1; break;
    case -125: vb = 1; break;
    case '/': if (*(toPrevEntry + i + 1) == -117) down = 1; break;
    case -116: case 'R': Good = FALSE; ModalDialogPlace("\pUse the radius, r, of the sphere.", "\pNot the
    radius, R, of the loop.", "\pRotational Kinetic Energy", "\pAT THE BOTTOM OF THE LOOP",
    128); return; break;
    case 'r': Good = FALSE; ModalDialogPlace("\pShould be r squared.", "\pSquare the r in your
    expression.", "\pRotational Kinetic Energy", "\pAT THE BOTTOM OF THE LOOP", 128); return;
    break;
    case '0': case '1': case '2': case '3': case '4': case '5': case '6': case '7': case '8': case '9': case ' ': case '*':
    break;
    case -127: case -124: case -110: case -128: case -109: case 'v': Good = FALSE;
    PositionDialog('ALRT', 1381); InitCursor; NoteAlert(1381, NULL); return; break;
    case -123: case -122: case -121: case -118: Good = FALSE; PositionDialog('ALRT', 1382);
    InitCursor; NoteAlert(1382, NULL); return; break;
    case -126: case -119: Good = FALSE; ModalDialogPlace("\pThe velocity needs to be squared.",
    "\pSquare the velocity of the ball.", "\pRotational Kinetic Energy", "\pAT THE BOTTOM OF THE
    LOOP", 128); return; break;
    case 'm': ModalDialogPlace("\pCapital M is the mass of the ball.", "\pReplace m by M.", "\pRotational
    Kinetic Energy", "\pAT THE BOTTOM OF THE LOOP", 128); return; break;
    default: Good = FALSE; if (M) { ModalDialogPlace("\pNOTE:I = 2/5 Mr", "\pAT THE BOTTOM
    OF THE LOOP", "\pB", "\p2", 2001); return; }
    else { ModalDialogPlace("\p", "\pAT THE BOTTOM OF THE LOOP", "\pB", "\p", 2001); return;
    break; }
    }
}
if (I == 1 && wb == 1 && fraction == 1 && !vb && !M && !r) {
    Good = FALSE; ModalDialogPlace("\pSubstitute for the moment of inertia I.", "\pFor the moment of
    inertia expression refer to 'Relations' in 'Need help with'.", "\pRotational
    Kinetic Energy", "\pAT THE BOTTOM OF THE LOOP", 128); return; }
else if (!I && wb == 1 && fraction != 2 && !vb && M == 1 && r == 1) { Good = FALSE;
    ModalDialogPlace("\pThe fraction is incorrect.", "\pIt should be 1/5.", "\pRotational Kinetic
    Energy", "\pAT THE BOTTOM OF THE LOOP", 128); return; }
else if (!I && wb == 1 && fraction == 2 && !vb && M == 1 && r == 1) { Good = FALSE;
    ModalDialogPlace("\pSubstitute for w.", "\pFor the expression refer to 'Relations' in 'Need help
    with'.", "\pRotational Kinetic Energy", "\pAT THE BOTTOM OF THE LOOP", 128); return;
    }
else if (!I && fraction == 1 && M == 1) {
    Good = FALSE; ModalDialogPlace("\pFraction incorrect.", "\pThe fraction should be 1/5.",
    "\pRotational Kinetic Energy", "\pAT THE BOTTOM OF THE LOOP", 128); return; }
else if (!I && !wb && fraction == 2 && vb == 1 && M == 1 && r == 1 && down == 1) {
    Good =
    FALSE; ModalDialogPlace("\pCancel r.", "\pThe r squared in the denominator should cancel the r
    squared in the numerator.", "\pRotational Kinetic Energy", "\pAT THE BOTTOM OF THE LOOP",
    128); return; }
else if (!I && !wb && fraction == 2 && vb == 1 && M == 1 && r == 1) {
    Good = FALSE; ModalDialogPlace("\pCancel r squared.", "\pA substitution error has been made. r
    squared should have cancelled.", "\pRotational Kinetic Energy", "\pAT THE BOTTOM OF THE
    LOOP", 128); return; }
else if ( M == 1 && vb == 1 && fraction == 2 && !r && !wb ) {
    for (i = 0; i <= Length; i++)
        *(toPrevEntry + i) = 0; *(toPrevEntry) = '7'; *(toPrevEntry + 1) = '/'; *(toPrevEntry + 2)
    =
    '1'; *(toPrevEntry + 3) = '0'; *(toPrevEntry + 4) = ' '; *(toPrevEntry + 4) = 'M';
    *(toPrevEntry + 5) = -125; Length = 7; }
else {

```

```

    Good = FALSE; ModalDialPlace("\pNOTE: v = wr", "\pAT THE BOTTOM OF THE LOOP",
    "\pB", "\p", 2001); return; }
}

void    CEquations::GetBotGPE(void)
{
if (*(toPrevEntry) == 0 || *(toPrevEntry) == '0') {
    ((CEditText*) gGopher)->SetTextString("\p0"); return; }
else {
    ModalDialPlace( "\pThe gravitational potential energy is 0.", "\pNOTE: Assuming zero reference
    level at the lowest position of the ball.", "\pGravitational Potential Energy", "\pAT THE BOTTOM
    OF THE LOOP", 128); return; }
}

void    CEquations::GetTopTKE(void)
{
    int i = 0, mValue = 0, wT2Val = 0, vT2Val = 0, R2Val = 0;

if (*(toPrevEntry) == 0) {
    ModalDialPlace("\pYou have not given an expression.", "\pProvide a value before continuing
    check.\rClick \"Need help with\" for expression.", "\pTranslational Kinetic Energy", "\pAT THE
    TOP OF THE LOOP", 128); return; }
if ( *(toPrevEntry) == '0') {
    ModalDialPlace("\pThe velocity of the ball is not zero at the top of the loop. Otherwise, it will fall
    off the track.", "\pTherefore the translational kinetic energy of the ball is not 0. \rProvide a value
    before continuing check.\rClick \"Need help with\" for K.E. expression.", "\pTranslational Kinetic
    Energy", "\pAT THE TOP OF THE LOOP", 128); return; }
GetFraction();
if ((num == 1 || num == 0) && (den == 2))
    ;
else {
    ModalDialPlace("\pThe expression should have a 1/2.", "\pClick \"Need help with\" for
    expression.", "\pTranslational Kinetic Energy", "\pAT THE TOP OF THE LOOP", 128); return; }
for ( i = 0; i <= Length - 1; i++) {
    switch (*(toPrevEntry + i)) {
    case 'M': mValue++; break;
    case -127: vT2Val++; break;
    case -121: wT2Val++; break;
    case -116: R2Val++; break;
    case -117: ModalDialPlace("\pFor a sliding ball", "\p v ≠ wr", "\pTranslational Kinetic Energy",
    "\pAT THE TOP OF THE LOOP", 128); break;
    case '0': case '1': case '2': case '3': case '4': case '5': case '6': case '7': case '8': case '9': case '/': case ' ':
    case '*': break;
    case -124: case -125: case -110: case -109: case -126: Good = FALSE; Good = FALSE;
    PositionDialog('ALRT', WrongVelSq); InitCursor; NoteAlert(WrongVelSq, NULL); return; break;
    case -120: case -118: Good = FALSE; PositionDialog('ALRT', WrongAngVelSq); InitCursor;
    NoteAlert(WrongAngVelSq, NULL); return; break;
    case -123 : case -122: case -119: Good = FALSE; PositionDialog('ALRT', 2004); InitCursor;
    NoteAlert(2004, NULL); return; break;
    case -128: Good = FALSE; ModalDialPlace("\pThe velocity needs to be squared.", "\pSquare the
    velocity of the ball.", "\pTranslational Kinetic Energy", "\pAT THE TOP OF THE LOOP", 128);
    return; break;
    case 'm': ModalDialPlace("\pCapital M is the mass of the ball.", "\pReplace m by M.",
    "\pTranslational Kinetic Energy", "\pAT THE TOP OF THE LOOP", 128); return; break;
    default: Good = FALSE; ModalDialPlace("\p", "\pAT THE TOP OF THE LOOP", "\pT", "\p1/2
    Mv", 2000); return; break; }
}
if (mValue == 1 && vT2Val == 1) {
    for (i = 0; i <= Length; i++)
        *(toPrevEntry + i) = 0;
    *(toPrevEntry) = '1'; *(toPrevEntry + 1) = '/'; *(toPrevEntry + 2) = '2'; *(toPrevEntry + 3) =
    ',';
    *(toPrevEntry + 4) = 'M'; *(toPrevEntry + 5) = -127; Length = 6; }
else if (wT2Val == 1 && R2Val == 1 && mValue == 1) {
    Good = FALSE; PositionDialog('ALRT', 2004); InitCursor; NoteAlert(2004, NULL); return; }
else {
    Good = FALSE; ModalDialPlace("\p", "\pAT THE TOP OF THE LOOP", "\pT", "\p1/2 Mv",
    2000); PositionDialog('ALRT', 2000); InitCursor; NoteAlert(2000, NULL); return; }
}

```

```

void CEquations::GetTopRKE(void)
{
    int i = 0, I = 0, down = 0, vb = 0, m = 0, wb = 0, r = 0, fraction = 0;

    if ((*toPrevEntry) == 0) {
        ModalDialogPlace("\pYou have not given an expression.", "\pProvide a value before continuing
        check.\rClick \"Need help with\" for expression.", "\pRotational Kinetic Energy", "\pAT THE
        TOP OF THE LOOP", 128); return; }
    if ( (*toPrevEntry) == '0') {
        ModalDialogPlace("\pThe velocity of the ball is not zero at the top of the loop. Otherwise, it will fall
        off the track.", "\pThe rotational kinetic energy of the ball is not 0.\rClick \"Need help with\" for
        expression.", "\pRotational Kinetic Energy", "\pAT THE TOP OF THE LOOP", 128); return; }
    GetFraction();
    if ((num == 1 || num == 0) && den == 2)
        fraction = 1;
    else if (den == 5 && (num == 1 || num == 0))
        fraction = 2;
    else if (num == 2 && den == 10) {
        Good = FALSE; ModalDialogPlace("\pSimplify expression.", "\pSimplify the fraction to 1/5.",
        "\pRotational Kinetic Energy", "\pAT THE TOP OF THE LOOP", 128); return; }
    else {
        Good = FALSE; ModalDialogPlace("\pFraction incorrect.", "\pSimplify the fraction or correct it.",
        "\pRotational Kinetic Energy", "\pAT THE TOP OF THE LOOP", 128); return; }
    for ( i = 0; i <= Length - 1; i++) {
        switch ((*toPrevEntry + i)) {
            case 'M': m = 1; break;
            case -121: wb = 1; break;
            case 'I': I = 1; break;
            case -117: r = 1; break;
            case -127: vb = 1; break;
            case '/': if ((*toPrevEntry + i + 1) == -117) down = 1; break;
            case -116: case 'R': ModalDialogPlace("\pUse the radius, r, of the sphere", "\p", "\pRotational Kinetic
            Energy", "\pAT THE TOP OF THE LOOP", 128); return; break;
            case 'r': Good = FALSE; ModalDialogPlace("\pShould be r squared.", "\pSquare the r in your
            expression.", "\pRotational Kinetic Energy", "\pAT THE TOP OF THE LOOP", 128); return; break;
            case '0': case '1': case '2': case '3': case '4': case '5': case '6': case '7': case '8': case '9': case ' ': case '*':
            break;
            case -124: case -125: case -110: case -109: case -126: Good = FALSE; Good = FALSE;
            PositionDialog('ALRT', WrongVelSq); InitCursor; NoteAlert(WrongVelSq, NULL); return; break;
            case -120: case -118: case -119: case -123: Good = FALSE; PositionDialog('ALRT',
            WrongAngVelSq); InitCursor; NoteAlert(WrongAngVelSq, NULL); return; break;
            case -128: case -122: Good = FALSE; ModalDialogPlace("\pThe velocity needs to be squared.",
            "\pSquare the velocity of the ball.", "\pRotational Kinetic Energy", "\pAT THE TOP OF THE
            LOOP", 128); return; break;
            case 'm': ModalDialogPlace("\pCapital M is the mass of the ball.", "\pReplace m by M.", "\pRotational
            Kinetic Energy", "\pAT THE TOP OF THE LOOP", 128); return; break;
            default:
                if (m ) {
                    ModalDialogPlace("\pNOTE:I = 2/5 Mr", "\pAT THE TOP F THE LOOP", "\pT", "\p2", 2001);
                    return; }
                else {
                    ModalDialogPlace("\p", "\pAT THE TOP OF THE LOOP", "\pT", "\p", 2001); return; } break; }
        }
    if (I == 1 && wb == 1 && fraction == 1 && !vb && !m && !r) {
        Good = FALSE; ModalDialogPlace("\pSubstitute for the moment of inertia I.", "\pFor the moment of
        inertia expression refer to \"Relations\" in \"Need help with\".", "\pRotational Kinetic Energy",
        "\pAT THE TOP OF THE LOOP", 128); return; }
    else if (!I && wb == 1 && fraction != 2 && !vb && m == 1 && r == 1) {
        Good = FALSE; ModalDialogPlace("\pThe fraction is incorrect.", "\pIt should be 1/5.", "\pRotational
        Kinetic Energy", "\pAT THE TOP OF THE LOOP", 128); return; }
    else if (!I && wb == 1 && fraction == 2 && !vb && m == 1 && r == 1) {
        Good = FALSE; ModalDialogPlace("\pSubstitute for w.", "\pFor the expression refer to \"Relations\"
        in \"Need help with\".", "\pRotational Kinetic Energy", "\pAT THE TOP OF THE LOOP", 128);
        return; }
    else if (!I && fraction == 1 && m == 1) {
        Good = FALSE; ModalDialogPlace("\pFraction incorrect.", "\pThe fraction should be 1/5.",
        "\pRotational Kinetic Energy", "\pAT THE TOP OF THE LOOP", 128); return; }
    else if (!I && !wb && fraction == 2 && vb == 1 && m == 1 && r == 1 && down == 1) {
        Good = FALSE; ModalDialogPlace("\pCancel r.", "\pThe r squared in the denominator should cancel
        the r squared in the numerator.", "\pRotational Kinetic Energy", "\pAT THE TOP OF THE LOOP",

```

```

128); return; }
else if (!I && !wb && fraction == 2 && vb == 1 && m == 1 && r == 1) {
    Good = FALSE; ModalDialPlace("\pCancel r squared.", "\pA substitution error has been made. r
squared should have cancelled.", "\pRotational Kinetic Energy", "\pAT THE TOP OF THE LOOP",
128); return; }
else if(fraction == 2 && m == 1 && vb == 1 && !r && !wb && !I) {
for (i = 0; i <= Length; i++)
    *(*toPrevEntry + i) = 0;
    *(*toPrevEntry) = '1'; *(*toPrevEntry + 1) = '/'; *(*toPrevEntry + 2) = '1'; *(*toPrevEntry + 3) =
'0'; *(*toPrevEntry + 4) = ' '; *(*toPrevEntry + 5) = 'M'; *(*toPrevEntry + 6) = -127; Length =
7; }
else {
    ModalDialPlace("\pNOTE: v = wr", "\pAT THE TOP OF THE LOOP", "\pT", "\p", 2001);
    return; }
}

void    CEquations::GetTopGPE(void)
{
    int i = 0, mVal = 0, gVal = 0, RVal = 0;

if (*(*toPrevEntry) == 0 ) {
    ModalDialPlace("\pYou have not given an expression.", "\pProvide a value before continuing
check.\rClick \"Need help with\" for expression.", "\pGravitational Potential Energy", "\pAT THE
TOP OF THE LOOP", 128); return; }
if ( *(*toPrevEntry) == '0') {
    ModalDialPlace("\pAt the top the gravitational potential energy is not 0.", "\pNOTE: Assuming
zero reference level at the lowest position of the ball.\rProvide a value before continuing
check.\rClick \"Need help with\" for expression.", "\pGravitational Potential Energy", "\pAT THE
TOP OF THE LOOP", 128); return; }
GetFraction();
if ((num == 1 || num == 0 || num == 2) && den == 1 )
    ;
else {
    ModalDialPlace("\pWrong expression.", "\pProvide a value before continuing check.\rClick \"Need
help with\" for expression.\rNOTE: Assume zero reference level at the lowest position of the ball.",
"\pGravitational Potential Energy", "\pAT THE TOP OF THE LOOP", 128); return; }
for ( i = 0; i <= Length - 1; i++) {
    switch (*(*toPrevEntry + i)) {
        case 'M': mVal++; break;
        case 'g': gVal++; break;
        case 'R': RVal++;
        if (num != 2) {
            ModalDialPlace("\pThe expression should have \"2\" in it.", "\pThe height of the ball at the top of
the loop is 2R.", "\pGravitational Potential Energy", "\pAT THE TOP OF THE LOOP", 128);
            return; } break;
        case '0': case '1': case '2': case '3': case '4': case '5': case '6': case '7': case '8': case '9': case '/': case ' ':
        case '*': break;
        case 'r': ModalDialPlace("\pReplace r by R", "\pThe height of the ball at the top of the loop is 2R.",
"\pGravitational Potential Energy", "\pAT THE TOP OF THE LOOP", 128); return; break;
        case 'y': ModalDialPlace("\pAt the top: y = 2R", "\pNOTE: Assuming zero reference level at the
lowest position of the ball.", "\pGravitational Potential Energy", "\pAT THE TOP OF THE LOOP",
128); return; break;
        case 'm': ModalDialPlace("\pCapital M is the mass of the ball.", "\pReplace m by M.",
"\pGravitational Potential Energy", "\pAT THE TOP OF THE LOOP", 128); return; break;
        default: ModalDialPlace("\pThe expression is wrong.", "\pClick \"Need help with\" for
expression.\rNOTE: Assume zero reference level at the lowest position of the ball.", "\pGravitational
Potential Energy", "\pAT THE TOP OF THE LOOP", 128); return; break; }
    }
    if (mVal == 1 && gVal == 1 && RVal == 1) {
        for (i = 0; i <= Length; i++)
            *(*toPrevEntry + i) = 0;
        *(*toPrevEntry) = '2'; *(*toPrevEntry + 1) = 'M'; *(*toPrevEntry + 2) = 'g'; *(*toPrevEntry + 3)
=
'R'; Length = 4; }
    else {
        ModalDialPlace("\pThe gravitational potential energy of the ball is 2MgR.", "\pNOTE: Assuming
zero reference level at the lowest position of the ball.", "\pGravitational Potential Energy", "\pAT
THE TOP OF THE LOOP", 128); return; }
    }
}

```

```

void    CEquations::GetInclineTKE(void)
{
    int i = 0, v2Val = 0, mVal = 0;

    if ((*toPrevEntry) == 0) {
        ModalDialogPlace("\pYou have not given an expression.", "\pProvide a value before checking.\rClick
        \Need help with\ for an expression.", "\pTranslational Kinetic Energy", "\pALONG THE
        INCLINE", 128); return; }
    if ( (*toPrevEntry) == '0') { ModalDialogPlace("\pThe translational kinetic energy of the ball is not
    0.", "\pProvide a value before checking.\rClick \Need help with\ for an expression.",
    "\pTranslational Kinetic Energy", "\pALONG THE INCLINE", 128); return; }
    GetFraction();
    if ((num == 1 || num == 0) && (den == 2 ))
        {
    else
        {
            ModalDialogPlace("\pThe expression should have a 1/2.", "\pClick \Need help with\ for an
            expression.", "\pTranslational Kinetic Energy", "\pALONG THE INCLINE", 128); return; }
        for ( i = 0; i <= Length - 1; i++) {
            switch ((*toPrevEntry + i)) {
                case 'M': mVal++; break;
                case -124: v2Val++; break;
                case '0': case '1': case '2': case '3': case '4': case '5': case '6': case '7': case '8': case '9': case '/': case ' ':
                case '*': break;
                case -125: case -127: case -110: case -109: case -128: case -126: Good = FALSE;
                PositionDialog('ALRT', WrongVelSq); InitCursor; NoteAlert(WrongVelSq, NULL); return; break;
                case 'v': Good = FALSE; ModalDialogPlace("\pThe velocity needs to be squared.", "\pSquare the
                velocity of the ball.", "\pTranslational Kinetic Energy", "\pALONG THE INCLINE", 128); return;
                break;
                case 'm': ModalDialogPlace("\pCapital M is the mass of the ball.", "\pReplace m by M.",
                "\pTranslational Kinetic Energy", "\pALONG THE INCLINE", 128); return; break;
                default: ModalDialogPlace("\p", "\pALONG THE INCLINE", "\p", "\p1/2 Mv", 2000); return; break; }
            }
            if (mVal == 1 && v2Val == 1) {
                for (i = 0; i <= Length; i++)
                    (*toPrevEntry + i) = 0;
                (*toPrevEntry) = '1'; (*toPrevEntry + 1) = '/'; (*toPrevEntry + 2) = '2'; (*toPrevEntry + 3) =
                ' '; (*toPrevEntry + 4) = 'M'; (*toPrevEntry + 5) = -124; Length = 6; }
            else
                {
                ModalDialogPlace("\p", "\pALONG THE INCLINE", "\p", "\p1/2 Mv", 2000); return; }
            }
        }

void    CEquations::GetInclineRKE(void)
{
    int i = 0, I = 0, down = 0, vb = 0, m = 0, wb = 0, r = 0, fraction = 0;

    if ((*toPrevEntry) == 0) {
        ModalDialogPlace("\pYou have not given an expression.", "\pProvide a value before checking.\rClick
        \Need help with\ for an expression.", "\pRotational Kinetic Energy", "\pAT A GENERAL
        POINT", 128); return; }
    if ( (*toPrevEntry) == '0') {
        ModalDialogPlace("\pThe rotational kinetic energy of a rolling ball is not 0.", "\pProvide a value
        before checking.\rClick \Need help with\ for an expression.", "\pRotational Kinetic Energy", "\pAT
        A GENERAL POINT", 128); return; }
    GetFraction();
    if ((num == 1 || num == 0) && den == 2)
        fraction = 1;
    else if (den == 5 && (num == 1 || num == 0))
        fraction = 2;
    else if (num == 2 && den == 10) {
        Good = FALSE; ModalDialogPlace("\pSimplify expression.", "\pSimplify the fraction to 1/5.",
        "\pRotational Kinetic Energy", "\pAT A GENERAL POINT", 128); return; }
    else
        {
        Good = FALSE; ModalDialogPlace("\pFraction incorrect.", "\pSimplify the fraction or correct it.",
        "\pRotational Kinetic Energy", "\pAT A GENERAL POINT", 128); return; }
        for ( i = 0; i <= Length - 1; i++) {
            switch ((*toPrevEntry + i)) {
                case 'M': m = 1; break;
                case 'm': ModalDialogPlace("\pCapital M is the mass of the ball.", "\pReplace m by M.", "\pRotational
                Kinetic Energy", "\pAT A GENERAL POINT", 128); return; break;
                case -118: wb = 1; break;
                case 'I': I = 1; break;
            }
        }
    }

```

```

case -117: r = 1; break;
case -124: vb = 1; break;
case '/': if ((*toPrevEntry + i + 1) == -117) down = 1; break;
case -116: case 'R': ModalDialPlace("\pUse the radius, r, of the sphere", "\p", "\pRotational Kinetic Energy", "\pAT A GENERAL POINT", 128); return; break;
case 'r': Good = FALSE; ModalDialPlace("\pShould be r squared.", "\pSquare the r in your expression.", "\pRotational Kinetic Energy", "\pAT A GENERAL POINT", 128); return; break;
case '0': case '1': case '2': case '3': case '4': case '5': case '6': case '7': case '8': case '9': case '.': case '*': break;
case -122: case -121: case -120: case -119: Good = FALSE; PositionDialog('ALRT', WrongAngVelSq); InitCursor; NoteAlert(WrongAngVelSq, NULL); return; break;
case -125: case -127: case -110: case -109: case -128: case -126: Good = FALSE; PositionDialog('ALRT', WrongVelSq); InitCursor; NoteAlert(WrongVelSq, NULL); return; break;
case -123: case 'v': Good = FALSE; ModalDialPlace("\pThe velocity needs to be squared.", "\pSquare the velocity of the ball.", "\pTranslational Kinetic Energy", "\pAT A GENERAL POINT", 128); return; break;
default: if (m ) {
    ModalDialPlace("\pNOTE: I = 2/5 mr", "\pAT A GENERAL POINT", "\p", "\p2", 2001); return; }
else {
    ModalDialPlace("\pNOTE: v = wr", "\pAT A GENERAL POINT", "\p", "\p", 2001); return; }
break; }
}
if (I == 1 && wb == 1 && fraction == 1 && !vb && !m && !r) {
    Good = FALSE; ModalDialPlace("\pSubstitute for the moment of inertia I.", "\pFor the moment of inertia expression refer to \"Relations\" in \"Need help with\".", "\pRotational Kinetic Energy", "\pAT A GENERAL POINT", 128); return; }
else if (!I && wb == 1 && fraction != 2 && !vb && m == 1 && r == 1) {
    Good = FALSE; ModalDialPlace("\pThe fraction is incorrect.", "\pIt should be 1/5.", "\pRotational Kinetic Energy", "\pAT A GENERAL POINT", 128); return; }
else if (!I && wb == 1 && fraction == 2 && !vb && m == 1 && r == 1) {
    Good = FALSE; ModalDialPlace("\pSubstitute for w.", "\pFor the expression refer to \"Relations\" in \"Need help with\".", "\pRotational Kinetic Energy", "\pAT A GENERAL POINT", 128); return; }
}
else if (!I && fraction == 1 && m == 1) {
    Good = FALSE; ModalDialPlace("\pFraction incorrect.", "\pThe fraction should be 1/5.", "\pRotational Kinetic Energy", "\pAT A GENERAL POINT", 128); return; }
else if (!I && !wb && fraction == 2 && vb == 1 && m == 1 && r == 1 && down == 1) {
    Good = FALSE; ModalDialPlace("\pCancel r.", "\pThe r squared in the denominator should cancel the r squared in the numerator.", "\pRotational Kinetic Energy", "\pAT A GENERAL POINT", 128); return; }
else if (!I && !wb && fraction == 2 && vb == 1 && m == 1 && r == 1) {
    Good = FALSE; ModalDialPlace("\pCancel r squared.", "\pA substitution error has been made. r squared should have cancelled.", "\pRotational Kinetic Energy", "\pAT A GENERAL POINT", 128); return; }
else if (fraction == 2 && m == 1 && vb == 1 && !r && !wb && !I) {
    for (i = 0; i <= Length; i++)
        (*toPrevEntry + i) = 0;
    (*toPrevEntry) = '7'; (*toPrevEntry + 1) = '/'; (*toPrevEntry + 2) = '1'; (*toPrevEntry + 3) = '0'; (*toPrevEntry + 4) = ' '; (*toPrevEntry + 5) = 'M'; (*toPrevEntry + 6) = -124; Length = 7;
}
else {
    ModalDialPlace("\pNOTE: v = wr", "\pAT A GENERAL POINT", "\p", "\p", 2001); return; }
}

void CEquations::GetInclineGPE(void)
{
    int i = 0, mVal = 0, gVal = 0, yVal = 0;

    if ((*toPrevEntry) == 0) {
        ModalDialPlace("\pYou have not given an expression.", "\pProvide a value before checking.\rClick \"Need help with\" for an expression.", "\pGravitational Potential Energy", "\pAT A GENERAL POINT", 128); return; }
    if ( (*toPrevEntry) == '0') {
        ModalDialPlace("\pAt a height y, the gravitational potential energy of the ball is Mgy.", "\pNOTE: Assuming zero reference level at the lowest position of the ball. \rClick \"Need help with\" for an expression.", "\pGravitational Potential Energy", "\pAT A GENERAL POINT", 128); return; }
    GetFraction();
    if ((num == 1 || num == 0) && den == 1)
        ;
    else
        {

```



```

ModalDialPlace("\pAt a height y, the gravitational potential energy of the ball is Mgy.", "\pNOTE:
Assuming zero reference level at the lowest position of the ball.", "\pGravitational Potential
Energy", "\pAT A GENERAL POINT", 128); return; }
for ( i = 0; i <= Length - 1; i++) {
switch (*(toPrevEntry + i)) {
case 'M': mVal++; break;
case 'm': ModalDialPlace("\pCapital M is the mass of the ball.", "\pReplace m by M.",
"\pGravitational Potential Energy", "\pAT A GENERAL POINT", 128); return; break;
case 'g': gVal++; break;
case 'y': yVal++; break;
case ' ': break;
default: ModalDialPlace("\pAt a height y, the gravitational potential energy of the ball is Mgy.",
"\pNOTE: Assuming zero reference level at the lowest position of the ball.", "\pGravitational Potential
Energy", "\pAT A GENERAL POINT", 128); return; break; }
}
if (mVal == 1 && gVal == 1 && yVal == 1) {
for (i = 0; i <= Length; i++)
*(toPrevEntry + i) = 0;
*(toPrevEntry) = 'M'; *(toPrevEntry + 1) = 'g'; *(toPrevEntry + 2) = 'y'; Length = 3; }
else {
ModalDialPlace("\pAt a height y, the gravitational potential energy of the ball is Mgy.", "\pNOTE:
Assuming zero reference level at the lowest position of the ball.", "\pGravitational Potential
Energy", "\pAT A GENERAL POINT", 128); return; }
}

void CEquations::GetLoopTKE(void)
{
int i = 0, mVal = 0, w2Val = 0, R2Val = 0, v2Val = 0;

if (*(toPrevEntry) == 0) {
ModalDialPlace("\pYou have not given an expression.", "\pProvide a value before checking.\rClick
\"Need help with\" for an expression.", "\pTranslational Kinetic Energy", "\pIN THE LOOP", 128);
return; }
if ( *(toPrevEntry) == '0') {
ModalDialPlace("\pThe translational kinetic energy of the ball is not 0.", "\pProvide a value before
checking.\rClick \"Need help with\" for an expression.", "\pTranslational Kinetic Energy", "\pIN
THE LOOP", 128); return; }
GetFraction();
if ((num == 1 || num == 0) && den == 2)
;
else {
ModalDialPlace("\pThe expression should have a 1/2.", "\p\rClick \"Need help with\" for an
expression.", "\pTranslational Kinetic Energy", "\pIN THE LOOP", 128); return; }
for ( i = 0; i <= Length - 1; i++) {
switch (*(toPrevEntry + i)) {
case 'M': mVal++; break;
case 'm': ModalDialPlace("\pCapital M is the mass of the ball.", "\pReplace m by M.",
"\pTranslational Kinetic Energy", "\pIN THE LOOP", 128); return; break;
case -124: v2Val++; break;
case -118: w2Val++; break;
case -116: R2Val++; break;
case '0': case '1': case '2': case '3': case '4': case '5': case '6': case '7': case '8': case '9': case '/': case ' ':
case '*': break;
case -127: case -125: case -110: Good = FALSE; PositionDialog('ALRT', WrongVelSq); InitCursor;
NoteAlert(WrongVelSq, NULL); return; break;
case -120: case -121: Good = FALSE; PositionDialog('ALRT', WrongAngVelSq); InitCursor;
NoteAlert(WrongAngVelSq, NULL); return; break;
default: ModalDialPlace("\p", "\pIN THE LOOP", "\p", "\p1/2 Mv", 2000); return; break; }
}
if (mVal == 1 && v2Val == 1) {
for (i = 0; i <= Length; i++)
*(toPrevEntry + i) = 0;
*(toPrevEntry) = '1'; *(toPrevEntry + 1) = '/'; *(toPrevEntry + 2) = '2'; *(toPrevEntry + 3) =
';'; *(toPrevEntry + 4) = 'M'; *(toPrevEntry + 5) = -124; Length = 6; }
else if (mVal == 1 && w2Val == 1 && R2Val == 1) {
Good = FALSE; PositionDialog('ALRT', 2004); InitCursor; NoteAlert(2004, NULL); return; }
else {
ModalDialPlace("\p", "\pIN THE LOOP", "\p", "\p1/2 Mv", 2000); return; }
}

```

```

static void GetFraction(void)
{
    int i = 0; num = 0; den = 1;

    for ( i = 0; i <= Length - 1; i++) {
        if (0 < ((*toPrevEntry + i) - '0') && ((*toPrevEntry + i) - '0') <= 9) {
            if (num == 0)
                num = ((*toPrevEntry + i)) - '0';
            else
                num = num * ((*toPrevEntry + i)) - '0';
            i++;
            if (0 < ((*toPrevEntry + i) - '0') && ((*toPrevEntry + i) - '0') <= 9) {
                num = 10 * num + ((*toPrevEntry + i) - '0');
                i++; }
        }
        if ((*toPrevEntry + i) == '/') {
            i++;
            if (0 < ((*toPrevEntry + i) - '0') && ((*toPrevEntry + i) - '0') <= 9) {
                if (den == 1)
                    den = ((*toPrevEntry + i) - '0');
                else
                    den = den * ((*toPrevEntry + i) - '0');
                i++;
                if (0 < ((*toPrevEntry + i) - '0') && ((*toPrevEntry + i) - '0') <= 9) {
                    den = den * 10 + ((*toPrevEntry + i) - '0');
                    i++;
                }
            }
        }
    }
}

```

CForces.c

(The director of the force equations windows)

#include the following: "Commands.h", "CForces.h", "Global.h", "AppCommands.h", <CBartender.h>, <CDialog.h>, "CLoopDoc.h", "CLoopApp.h", <CDesktop.h>, <TBUtilities.h>, "CForcesEqCheck.h", <CEditText.h>, <CRadioGroupPane.h>, "CStartDirPanel.h", "CEquations.h"

CForces *theForcesEq; extern CStartDirPanel *theStartWind; extern CWindow **otherWindDial; extern CBartender *gBartender; extern CBureaucrat *gGopher; extern int windowItemNum; int kForces;

```

enum
{
    kRadioRollSlidMinID = 34, kSlidingMin, kRollingMin    };

```

```

void    CForces::IForces(short DLOGid, CDirectorOwner *aSupervisor)
{
    CDLOGDirector::IDLOGDirector( DLOGid, aSupervisor);
}

```

```

void    CForces::BegDialForces(long theCommand)
{
    CRadioGroupPane *radioGroupSet; CDialog *theDialog = (CDialog*) itsWindow;

```

```

theDialog->Select(); inherited::BeginDialog(); kForces = 1; radioGroupSet = (CRadioGroupPane
*)itsWindow->FindViewByID(kRadioRollSlidMinID); radioGroupSet ->
SetStationID(kSlidingMin); }

```

```

void    CForces::ThrowForcesDial(long closingButton)
{
    CDialog *theDialogStart = (CDialog*) itsWindow;

```

```

inherited::EndDialog(closingButton, FALSE); theDialogStart->Select(); theDialogStart->Close();
windowItemNum--;
}

```

```

void    CForces::Activate(void)
{
    CLoopDoc *theWindowChanging = NULL;

```

```

inherited::Activate(); theWindowChanging = new (CLoopDoc); theWindowChanging->
ChangingMenus();
}

```

```

void    CForces::DoCommand(long theCommand)
{
    CStartDirPanel *theStartDir; CEquations *theEquations; CursHandle gWatchCursor;
    short theModifiers; Str255 theTitle;

```

```

switch (theCommand) {
case cmdQuitForces: theForcesEq->CloseDialForces(); theForcesEq->ThrowForcesDial
(cmdQuitForces); gBartender->RemoveMenu(MENUForcesEq); theForcesEq = NULL;
ForgetObject(theForcesEq); break;
case cmdRelationsForce: ForcesDialClar(ReferenceForces); break;
case cmdSolutionHelp: ForcesDialClar(SolutionHelp); break;
case cmdForcesIncline: kForces = 1; break;
case cmdForcesBotLoop: kForces = 2; break;
case cmdForcesTopLoop: kForces = 3; break;
case cmdForcesGenLoop: kForces = 4; break;
case cmdCheckForces: WhichTanNormalCheck(); break;
case cmdgbtn: case cmdhbtn: case cmdlbtn: case cmdMbtn: case cmdRbtn: case
cmdybtn: case cmdOpenParenbtn: case cmdCloseParenbtn: case cmdEqualbtn: case cmdMultbtn: case
cmdDividebtn: case cmdAddbtn: case cmdSubbtn: case cmdSqRtbtn: case cmdReturnbtn: case
cmdAnyVel: case cmdTopVel: case cmdBottomVel: case cmdStartVel: case cmdAnyAngVel: case
cmdTopAngVel: case cmdBottomAngVel: case cmdAccel: case cmdForces: case cmdReaction: case
cmdPhi: case cmdBeta: case cmdFriction: theStartDir = new (CStartDirPanel); theStartDir->
PlacetheChar(theCommand); break;
case cmdCosine: theEquations = new (CEquations); theEquations->SineCosinePlace('c', 'o', 's');
break;
case cmdSine: theEquations = new (CEquations); theEquations->SineCosinePlace('s', 'i', 'n'); break;
case cmdSquarebtn: theStartDir = new (CStartDirPanel); theStartDir->Squaring(); break;
case cmdZero: case cmdOne: case cmdTwo: case cmdThree: case cmdFour: case cmdFive: case
cmdSix: case cmdSeven: case cmdEight: case cmdNine: gBartender->GetCmdText(theCommand,
theTitle); ((CEditText*) gGopher->TypeChar(theTitle[1], theModifiers); break;
default: inherited::DoCommand(theCommand); }
}

void    CForces::CloseDialForces( void)
{
    Str255 theTitle; short z = 0; CWindow *theWindow;

itsWindow->GetTitle(theTitle); z = gBartender->FindItemText( MENUWindow, theTitle);
gBartender->RemoveMenuCmd(gBartender->FindCmdNumber( MENUWindow, z));
for (z = z; z < windowItemNum; z++)
    *(otherWindDial + z) = *(otherWindDial + z + 1);
*(otherWindDial + z) = NULL; ForgetPtr(*(otherWindDial + z));    }

void    CForces::WhichTanNormalCheck(void)
{
    CRadioGroupPane *radioGroupSet;

radioGroupSet = (CRadioGroupPane *)itsWindow->FindViewByID(kRadioRollSlidMinID);
switch (radioGroupSet -> GetStationID()) {
case kSlidingMin: WhichForcePosRad(); break;
case kRollingMin: WhichForcePosRoll(); break; }
}

void    CForces::WhichForcePosRad(void)
{
    switch (kForces) {
case 1: ForcesDialClar(ForcesAnsIncline); break;
case 2: ForcesDialClar(ForcesAnsBottom); break;
case 3: ForcesDialClar(ForcesAnsTop); break;
case 4: ForcesDialClar(ForcesAnsGeneral); break;
default: break; }
}

void    CForces::WhichForcePosRoll(void)
{
    switch (kForces) {
case 1: ForcesDialClar(AnsRollIncline); break;
case 2: ForcesDialClar(ForcesAnsBottom); break;
case 3: ForcesDialClar(ForcesAnsTop); break;
case 4: ForcesDialClar(AnsRollGeneral); break;
default: break; }
}

void    CForces::ForcesDialClar(short DialogueId)
{
    CDLOGDirector *theInfo; CDialog *aDialog = (CDialog*) itsWindow;

```

```
theInfo = new (CDLOGDirector); theInfo->IDLOGDirector(DialogueId, this); theInfo->
DoModalDialog(cmdOK); theInfo->EndDialog(cmdOK, TRUE); theInfo->Close(TRUE); aDialog->
Update(); }
```

CForcesCheck.c

(The methods which check the force equations)

```
#include the following: "Commands.h", "CForcesCheck.h", "Global.h", "AppCommands.h",
<CBartender.h>, <CDialog.h>, "CLoopDoc.h", "CLoopApp.h", <CDesktop.h>, <TBUtilities.h>,
"CStartDirPanel.h", "CForcesEqCheck.h", <CEditText.h>, <CRadioGroupPane.h>, "CEquations.h",
"CForcesSimulate.h", "CForcesEquations.h", "CEnergyCheck.h"
```

```
enum
{
    kRadioFCheckID = 1, kNormalFCheck = 25, kTangentFCheck      };
enum
{
    kRadioRollSlidCheckID = 39, kSlidForcesCheck, kRollForcesCheck  };
```

```
CForcesCheck *theForcesCheck; extern CStartDirPanel *theStartWind; extern CForcesSimulate
*theDrawResolve; extern CWindow **otherWindDial; extern CBartender *gBartender; extern
CBureaucrat *gGopher; extern int windowItemNum, posForces; short PosFEqCh; extern Handle
toPrevEntry; extern long Length; int friction, noFriction;
```

```
void    CForcesCheck::IForcesCheck(short DLOGid, CDirectorOwner *aSupervisor)
{
    CDLOGDirector::IDLOGDirector( DLOGid, aSupervisor);
}
```

```
void    CForcesCheck::BegDialForceCheck(long theCommand, long theDefaultCmd)
{
    CRadioGroupPane *radioGroupSet; CDialog *theDialog = (CDialog*) itsWindow;
```

```
theDialog->Select();
if (theDefaultCmd == cmdQUITForceDiag)
    theDialog->SetDefaultCmd(theDefaultCmd);
inherited::BeginDialog();
if (theCommand == cmdQuitForcesEq) {
    radioGroupSet = (CRadioGroupPane *)itsWindow->FindViewByID(kRadioFCheckID);
    radioGroupSet -> SetStationID(kNormalFCheck); radioGroupSet = (CRadioGroupPane
*)itsWindow->FindViewByID(kRadioRollSlidCheckID); radioGroupSet ->
    SetStationID(kSlidForcesCheck); }
}
```

```
void    CForcesCheck::ThrowForceCheckDial(long closingButton)
{
    CDialog *theDialogStart = (CDialog*) itsWindow;
```

```
inherited::EndDialog(closingButton, FALSE); theDialogStart->Select(); theDialogStart->Close();
windowItemNum--;
```

```
void    CForcesCheck::Activate(void)
{
    CLoopDoc *theWindowChanging = NULL;
```

```
inherited::Activate(); theWindowChanging = new (CLoopDoc); theWindowChanging->
ChangingMenus(); }
```

```
void    CForcesCheck::DoCommand(long theCommand)
{
    short theModifiers; CStartDirPanel *theDirPanel; CEquations *theEquations; CursHandle
gWatchCursor; Str255 theTitle;
```

```
switch (theCommand) {
case cmdQuitForcesEq: gWatchCursor = GetCursor(watchCursor); SetCursor(*gWatchCursor);
theForcesCheck->CloseForceCheckDial(); theForcesCheck->ThrowForceCheckDial
(cmdQuitForcesEq); gBartender->RemoveMenu(MENUForcesCheck); theForcesCheck = NULL;
ForgetObject(theForcesCheck); InitCursor(); break;
case cmdRelationsForce: ForcesCheckDial(ReferenceForces); break;
case cmdSolutionHelp: ForcesCheckDial(HowStartFEquatCheck); break;
case PosForcesEq: PosFEqCh = 0; break;
case PosFEqIncline: PosFEqCh = 1; break;
case PosFEqBottom: PosFEqCh = 2; break;
case PosFEqTop: PosFEqCh = 3; break;
```

```

case PosFEqGeneral: PosFEqCh = 4; break;
case cmdDiagramFEq: gWatchCursor = GetCursor(watchCursor); SetCursor(*gWatchCursor);
if (theDrawResolve == NULL) {
    posForces = 1; theDrawResolve = new(CForcesSimulate); theDrawResolve->IForcesSimulate
    (SimulateForces, this); theDrawResolve->BegDialDBut(0, cmdQUITForceDiag);
    (*(theDrawResolve).itsWindow).ID = SimulateForces; }
else
    (*(theDrawResolve).itsWindow->Select(); InitCursor(); break;
case cmdgbtn: case cmdMbtn: case cmdrbtn: case cmdRbtn: case cmdEqualbtn: case cmdMultbtn: case
cmdDividebtn: case cmdAddbtn: case cmdSubbtn: case cmdReturnbtn: case cmdAnyVel: case
cmdTopVel: case cmdBottomVel: case cmdStartVel: case cmdAnyAngVel: case cmdTopAngVel: case
cmdBottomAngVel: case cmdAccel: case cmdForces: case cmdReaction: case cmdPhi: case
cmdBetabtn: case cmdFriction: theDirPanel = new (CStartDirPanel); theDirPanel->
PlaceTheChar(theCommand); break;
case cmdSquarebtn: theDirPanel = new (CStartDirPanel); theDirPanel->Squaring(); break;
case cmdZero: case cmdOne: case cmdTwo: case cmdThree: case cmdFour: case cmdFive: case
cmdSix: case cmdSeven: case cmdEight: case cmdNine: gBartender->GetCmdText(theCommand,
theTitle); ((CEditText*) gGopher)->TypeChar(theTitle[1], theModifiers); break;
case cmdCosine: theEquations = new (CEquations); theEquations->SineCosinePlace('c', 'o', 's');
break;
case cmdSine: theEquations = new (CEquations); theEquations->SineCosinePlace('s', 'i', 'n'); break;
case cmdCheckForcesEq: ForcesSlidRollCheckChoice(); break;
default: inherited::DoCommand(theCommand); break; }

void    CForcesCheck::CloseForceCheckDial( void)
{
    Str255 theTitle; short z = 0; CWindow *theWindow;

    itsWindow->GetTitle(theTitle); z = gBartender->FindItemText( MENUWindow, theTitle);
    gBartender
->RemoveMenuCmd(gBartender->FindCmdNumber( MENUWindow, z));
    for (z = z; z < windowItemNum; z++)
        *(otherWindDial + z) = *(otherWindDial + z + 1);
    *(otherWindDial + z) = NULL; ForgetPtr(*(otherWindDial + z)); }

void    CForcesCheck::ForcesSlidRollCheckChoice(void)
{
    CRadioGroupPane *radioGroupSet; CEquations *theEquations;

    radioGroupSet = (CRadioGroupPane *)itsWindow->FindViewById(kRadioRollSlidCheckID);
    switch (radioGroupSet -> GetStationID()) {
    case kSlidForcesCheck: friction = 0; noFriction = 1; RadioTanNormChoice(); break;
    case kRollForcesCheck: friction = 1; noFriction = 0; RadioTanNormChoice(); break;
    default: theEquations = new (CEquations); theEquations->ModalDialPlace("\pSelect sliding or
    rolling.", "\pThere is a difference in forces involved along the tangential direction, due to the presence
    of friction for rolling. The forces in the normal direction are the same for sliding or rolling.",
    "\pSLIDING OR ROLLING NOT SPECIFIED", "\p", 128); break; }
}

void    CForcesCheck::RadioTanNormChoice(void)
{
    CRadioGroupPane *radioGroupSet; CEquations *theEquations;

    radioGroupSet = (CRadioGroupPane *)itsWindow->FindViewById(kRadioFCheckID);
    switch (radioGroupSet -> GetStationID()) {
    case kTangentFCheck: TanCheckForce(); break;
    case kNormalFCheck: NormalCheckForce(); break;
    default: theEquations = new (CEquations); theEquations->ModalDialPlace("\pSelect direction for
    force equation.", "\p", "\pDIRECTION NOT SELECTED", "\p", 128); break; }
}

void    CForcesCheck::TanCheckForce(void)
{
    CForcesEqCheck *theForcesEqCheck; CForcesEquations *theForcesEquat;

    switch (PosFEqCh) {
    case 1: EditWindForcesCheck(31); theForcesEqCheck = new (CForcesEqCheck);
    theForcesEqCheck->ForcesInclTan(); break;
    case 2: case 3: EditWindForcesCheck(31); theForcesEqCheck = new (CForcesEqCheck);
    theForcesEqCheck->TanLoopTopBot(); break;

```

```

case 4: EditWindForcesCheck(31); theForcesEqEquat = new (CForcesEquations); theForcesEqEquat->
TanLoopGen(); break;
default: theForcesEqCheck = new (CForcesEqCheck); theForcesEqCheck->
ModalDialEqCheck("\pSelect a position of the ball.", "\p", "\pPOSITION NOT SELECTED", "\p",
128); break; }
}

void CForcesCheck::NormalCheckForce(void)
{
    CForcesEqCheck *theForcesEqCheck; CForcesEquations *theForcesEqEquat;

switch (PosFEqCh) {
case 1: EditWindForcesCheck(31); theForcesEqCheck = new (CForcesEqCheck);
theForcesEqCheck-> ForcesInclNorm(); break;
case 2: EditWindForcesCheck(31); theForcesEqCheck = new (CForcesEqCheck);
theForcesEqCheck-> NormLoopBot(); break;
case 3: EditWindForcesCheck(31); theForcesEqCheck = new (CForcesEqCheck);
theForcesEqCheck-> NormLoopTop(); break;
case 4: EditWindForcesCheck(31); theForcesEqEquat = new (CForcesEquations); theForcesEqEquat->
NormLoopGen(); break;
default: theForcesEqCheck = new (CForcesEqCheck); theForcesEqCheck->
ModalDialEqCheck("\pSelect a position of the ball.", "\p", "\pPOSITION NOT SELECTED", "\p",
128); break; }
}

void CForcesCheck::ForcesCheckDial(short DialogueId)
{
    CDLOGDirector *theInfo; CDialog *aDialog = (CDialog*) itsWindow;

theInfo = new (CDLOGDirector); theInfo->IDLOGDirector(DialogueId, this); theInfo->
DoModalDialog(cmdOK); theInfo->EndDialog(cmdOK, TRUE); theInfo->Close(TRUE); aDialog->
Update();
}

void CForcesCheck::EditWindForcesCheck(short dialogItemNo)
{
    CDialog *theDialog = (CDialog*) itsWindow; tGopherInfo gopherInfo; CView
    *newGopher;
do {
    gopherInfo.currentGopher = (CView*) gGopher; theDialog->FindGophers(&gopherInfo); newGopher
    = gopherInfo.nextGopher; newGopher->BecomeGopher(TRUE); }
while(newGopher->ID != dialogItemNo);
Length = ((CEditText*) gGopher)->GetLength(); toPrevEntry = ((CEditText*) gGopher)->
CopyTextRange(0, ((CEditText*) gGopher)->GetLength());}

```

CForcesEqCheck.c

(Includes the methods for checking the force equations)

#include the following: "Commands.h", "CForcesEqCheck.h", "CLoopApp.h", <CDesktop.h>, <CEditText.h>, <TBUilities.h>, <CDialog.h>

extern CBureaucrat *gGopher; extern Handle toPrevEntry; extern long Length; extern short PosFEqCh; extern int friction, noFriction;

```

void CForcesEqCheck::ModalDialEqCheck(Str255 theText0, Str255 theText1, Str255 theText2,
Str255 theText3, short IdNum)
{
    ParamText(theText0, theText1, theText2, theText3); PositionDialog('ALRT', IdNum); InitCursor;
    NoteAlert(IdNum, NULL); return; }

```

```

int CForcesEqCheck::CheckforNumber(void)
{
    int i = 0;

```

```

for (i = 0; i <= Length - 1; i++) {
if (0 < ((*toPrevEntry + i) - '0') && ((*toPrevEntry + i) - '0') <= 9) {
    ModalDialEqCheck("\pYou have placed a number in the equation.", "\pIt should not have any
    number in it.", "\p", "\pNUMBER IN EQUATION", 128); return(1); }
if ((*toPrevEntry + i) == '0') {
    ModalDialEqCheck("\p", "\pThe equation needs to be simplified prior to checking. Use the
    worksheet to perform the working.", "\p", "\pSIMPLIFY EQUATION", 128); return(1); }
}

```

```

} return(0);
}

int CForcesEqCheck::IsThereEqual(void)
{
    int i = 0;

    for ( i = 0; i <= Length - 1; i++)
        if (*(toPrevEntry + i) == '=')
            return(1);
    ModalDialEqCheck("\p", "\pThere is no \"=\" symbol in your equation.", "\p", "\pNO EQUAL SIGN",
    128); return(0); }

void CFForcesEqCheck::ForcesInclTan(void)
{
    int i, Mval, gVal, Fval, aVal, sinVal, cosVal, PhiVal, frict;
    i = Mval = gVal = Fval = aVal = sinVal = cosVal = PhiVal = frict = 0;

    if (*(toPrevEntry) == 0) {
        ModalDialEqCheck("\pPlace the equation you wish to check in the \"Check window\".", "\pIf you
        are unsure, click on \"Force Diagram\".\rOtherwise, start with F = ma and click the \"CHECK\"
        button.", "\pALONG THE INCLINE", "\pNO EQUATION IN CHECK WINDOW", 128); return; }
    if (*(toPrevEntry) == '0') {
        ModalDialEqCheck("\pYou have placed \"0\" in the Check window.", "\pIf you are unsure, click on
        \"Force Diagram\".\rOtherwise, start with F = ma and click the \"CHECK\" button.", "\pALONG
        THE INCLINE", "\pNO EQUATION IN CHECK WINDOW", 128); return; }
    if (!IsThereEqual())
        return;
    if (CheckforNumber())
        return;
    for ( i = 0; i <= Length - 1; i++)
        switch (*(toPrevEntry + i)) {
            case '*': case ' ': case '=': break;
            case 'm': ModalDialEqCheck("\pYou have used \"m\".", "\pUse the capital alphabet \"M\".", "\p",
            "\pLOWER CASE ALPHABET USED", 128); return; break;
            case 'f': frict++; break;
            case 'M': Mval++; break;
            case 'g': gVal++; break;
            case 'F': Fval++; break;
            case 'a': aVal++; break;
            case -81: PhiVal++; break;
            case 's': if (*(toPrevEntry + i + 1) == 'i' && *(toPrevEntry + i + 2) == 'n') { i += 2; sinVal++;
            } break;
            case 'c': if (*(toPrevEntry + i + 1) == 'o' && *(toPrevEntry + i + 2) == 's') { i += 2; cosVal++;
            } break;
            case 'B': ModalDialEqCheck("\pUse Ø instead of β.", "\pØ is the angle of the incline.", "\pALONG
            THE INCLINE", "\pWRONG ANGLE SYMBOL", 128); return; break;
            case 'n': ModalDialEqCheck("\pYou have used \"n\".", "\pUse the capital alphabet \"N\".", "\p",
            "\pLOWER CASE ALPHABET USED", 128); return; break;
            case 'N': ModalDialEqCheck("\pN is the normal force of the track on the ball.", "\pIt acts
            perpendicular to the track. You are considering motion along the incline.\rFor information on
            components of forces, click \"Force Diagram\".\rOtherwise, start with F = Ma and click the
            \"CHECK\" button.", "\pALONG THE INCLINE", "\pWRONG FORCE", 128); return; break;
            default: ModalDialEqCheck("\pError in symbols used.", "\pFor information on components of forces,
            click \"Force Diagram\".\rOtherwise, start with F = Ma and click the \"CHECK\" button.", "\pALONG
            THE INCLINE", "\pUNRECOGNISED SYMBOL", 128); return; break; }
    if (noFriction) {
        if (cosVal && PhiVal == 1) {
            ModalDialEqCheck("\pN is the normal force of the track on the ball.", "\pIt acts perpendicular to the
            track. You are considering motion along the incline.\rFor information on components of forces,
            click \"Force Diagram\".\rOtherwise, start with F = Ma and click the \"CHECK\" button.",
            "\pALONG THE INCLINE", "\pWRONG FORCE", 128); return; }
    }
    if (frict) {
        ModalDialEqCheck("\pf is the frictional force.", "\pFor sliding motion there is no frictional force
        present.\rFor information on components of forces, click \"Force Diagram\".\rOtherwise, start with
        F = Ma and click the \"CHECK\" button.", "\pALONG THE INCLINE", "\pWRONG FORCE",
        128); return; }
    if (cosVal && PhiVal == 1) {
        ModalDialEqCheck("\pIt should be \"sin\" instead of \"cos\".", "\pFor information on components
        of forces, click \"Force Diagram\".", "\pALONG THE INCLINE", "\pWRONG TRIGONOMETRIC
        FUNCTION", 128); return; }
}

```

```

if (sinVal && PhiVal != 1) {
    ModalDialEqCheck("\pArgument of sine should be \"\sin\".", "\pFor information on components of
    forces, click \"Force Diagram\".", "\pALONG THE INCLINE", "\pWRONG TRIGONOMETRIC
    FUNCTION", 128); return; }
if (Fval == 1 && Mval == 1 && aVal == 1 && !gVal && !sinVal && !PhiVal){
    ModalDialEqCheck("\pYour equation should read F = Ma.", "\pReplace F by the component of Mg
    along the incline. For information on components of forces, click \"Force Diagram\".", "\pALONG
    THE INCLINE", "\pSUBSTITUTION REQUIRED", 128); return; }
else if ((Mval == 2 || Mval == 1) && aVal == 1 && gVal == 1 && !sinVal && !Fval) {
    ModalDialEqCheck("\pYour equation should read Ma = Mg.", "\pReplace Mg by
    Mgsin\theta. Mgsin\theta is the component of Mg along the incline. There is no other force in that
    direction.\r The M's can be cancelled. For information on components of forces, click \"Force
    Diagram\".", "\pALONG THE INCLINE", "\pSUBSTITUTION REQUIRED", 128); return; }
else if (Mval == 1 && !aVal && gVal == 1 && !sinVal && Fval == 1) {
    ModalDialEqCheck("\pYour equation should read F = Mg.", "\pReplace Mg by Mgsin\theta. Mgsin\theta
    is the component of Mg along the incline. There is no other force in that direction. For information
    on components of forces, click \"Force Diagram\".", "\pALONG THE INCLINE",
    "\pSUBSTITUTION REQUIRED", 128); return; }
else if ((Mval == 2 || Mval == 1) && aVal == 1 && gVal == 1 && sinVal == 1 && PhiVal == 1
    && !Fval) {
    ModalDialEqCheck("\pYour equation should read Ma = Mgsin\theta.", "\pSimplify the equation by
    dividing by M.", "\pALONG THE INCLINE", "\pSIMPLIFY", 128); return; }
else if (!Mval && aVal == 1 && gVal == 1 && !sinVal && !PhiVal && !Fval) {
    ModalDialEqCheck("\pYour equation should read a = g", "\pReplace g by gsin\theta. gsin\theta is the
    acceleration along the plane", "\pALONG THE INCLINE", "\pCHECK COMPLETED", 128);
    return; }
else if (!Mval && aVal == 1 && gVal == 1 && sinVal == 1 && PhiVal == 1 && !Fval) {
    ModalDialEqCheck("\pYour equation is correct if it reads: \"pa = gsin\theta\", \"\pALONG THE
    INCLINE\", \"\pCHECK COMPLETED\", 128); return; }
else if (Fval == 1 && Mval == 1 && !aVal && gVal == 1 && sinVal == 1 && PhiVal == 1) {
    ModalDialEqCheck("\pYour equation should read F = Mgsin\theta.", "\pSubstitute for F using equation
    from \"relations\" in \"Need help with\".", "\pALONG THE INCLINE", "\pSUBSTITUTION
    REQUIRED", 128); return; }
else if (Fval == 1 && Mval == 1 && aVal == 1 && !gVal && sinVal == 1 && PhiVal == 1) {
    ModalDialEqCheck("\pThe component of acceleration along the plane is \"gsin\theta\".", "\pYour
    equation should read F = Mgsin\theta. For information on components of forces, click \"Force
    Diagram\".", "\pALONG THE INCLINE", "\pSUBSTITUTION REQUIRED", 128); return; }
else {
    ModalDialEqCheck("\pError in equating the forces", "\pFor help, start with F = Ma and click the
    \"CHECK\" button.\r For information on components of forces, click \"Force Diagram\".",
    "\pALONG THE INCLINE", "\pEQUATING ERROR", 128); return; }
}
if (friction) {
    if (cosVal && PhiVal == 1) {
        ModalDialEqCheck("\plt should be \"\sin\" instead of \"\cos\".", "\pFor information on components
        of forces, click \"Force Diagram\".", "\pALONG THE INCLINE", "\pWRONG TRIGONOMETRIC
        FUNCTION", 128); return; }
    if (sinVal && PhiVal != 1) {
        ModalDialEqCheck("\pArgument of sine should be \"\sin\".", "\pFor information on components of
        forces, click \"Force Diagram\".", "\pALONG THE INCLINE", "\pWRONG TRIGONOMETRIC
        FUNCTION", 128); return; }
    if (Fval == 1 && Mval == 1 && aVal == 1 && !gVal && !sinVal && !PhiVal && !frict) {
        ModalDialEqCheck("\pYour equation should read F = Ma.", "\pReplace F by the frictional force f
        and the component of Mg along the incline. For information on components of forces, click \"Force
        Diagram\".", "\pALONG THE INCLINE", "\pSUBSTITUTION REQUIRED", 128); return; }
    else if ((Mval == 2 || Mval == 1) && aVal == 1 && gVal == 1 && sinVal == 1 && PhiVal == 1
        && !Fval && frict == 1) {
        ModalDialEqCheck("\pYour equation is correct if it reads: Ma = Mgsin\theta - f.", "\pMgsin\theta acts
        along the direction of acceleration and the frictional force f, acts in the opposed direction.\r The
        direction of acceleration is considered positive.", "\pALONG THE INCLINE", "\pSIMPLIFY",
        128); return; }
    else if (Fval == 1 && Mval == 1 && !aVal && gVal == 1 && sinVal == 1 && PhiVal == 1 &&
        frict == 1) {
        ModalDialEqCheck("\pYour equation should read F = Mgsin\theta - f.", "\pSubstitute for F using the
        equation given in \"relations\" in \"Need help with\".", "\pALONG THE INCLINE",
        "\pSUBSTITUTION REQUIRED", 128); return; }
    else if (!Fval && Mval == 2 && aVal == 1 && gVal == 1 && sinVal == 1 && PhiVal == 1 &&
        frict == 1) {
        ModalDialEqCheck("\pIf your equation reads Ma = Mgsin\theta - f.", "\pThe equation is correct.",

```



```

    "pALONG THE INCLINE", "pCHECK COMPLETED", 128); return; }
else if (Fval == 1 && Mval == 1 && !aVal && gVal == 1 && !sinVal && !PhiVal && frict ==
    1) {
    ModalDialEqCheck("pYour equation is missing the factor \"sinØ\".", "pYour equation should read
    F = MgsinØ - f. For information on components of forces, click \"Force Diagram\".", "pALONG
    THE INCLINE", "pSUBSTITUTION REQUIRED", 128); return; }
else if (Fval == 1 && Mval == 1 && aVal == 1 && !gVal && sinVal == 1 && PhiVal == 1 &&
    frict == 1) {
    ModalDialEqCheck("pThe component of acceleration along the plane is \"gsinØ\".", "pYour
    equation should read F = MgsinØ - f. For information on components of forces, click \"Force
    Diagram\".", "pALONG THE INCLINE", "pSUBSTITUTION REQUIRED", 128); return; }
else {
    ModalDialEqCheck("pError in equating the forces", "pFor help, start with F = Ma and click the
    \"CHECK\" button.\rFor information on components of forces, click \"Force Diagram\".",
    "pALONG THE INCLINE", "pEQUATING ERROR", 128); return; }
}

void CFforcesEqCheck::ForcesInclNorm(void)
{
    int i, Nval, Mval, gVal, sinVal, cosVal, PhiVal, Fval, aVal;
    i = Nval = Mval = gVal = sinVal = cosVal = PhiVal = Fval = aVal = 0;

    if ((*toPrevEntry) == 0) {
        ModalDialEqCheck("pPlace the equation you wish to check in the \"Check window\".", "pFor
        information on components of forces, click \"Force Diagram\".\rOtherwise, start with F = ma and
        click the \"CHECK\" button.", "pPERPENDICULAR TO THE INCLINE", "pNO EQUATION
        IN CHECK WINDOW", 128); return; }
    if ((*toPrevEntry) == '0') {
        ModalDialEqCheck("pYou have placed \"0\" in the \"Check window\".", "pFor
        information on components of forces, click \"Force Diagram\".\rOtherwise, start with F = ma and
        click the \"CHECK\" button.", "pPERPENDICULAR TO THE INCLINE", "pNO EQUATION
        IN CHECK WINDOW", 128); return; }

    if (!IsThereEqual())
        return;
    if (CheckforNumber())
        return;
    for (i = 0; i <= Length - 1; i++)
        switch ((*toPrevEntry + i)) {
            case '*': case ' ': case '=': break;
            case 'M': Mval++; break; case 'm': ModalDialEqCheck("pYou have used \"m\".", "pUse the capital
            alphabet \"M\".", "p", "pLOWER CASE ALPHABET USED", 128); return; break;
            case 'g': gVal++; break;
            case 'F': Fval++; break;
            case 'f': ModalDialEqCheck("pYou have used \"f\" the symbol for the frictional force.", "pUse the
            capital alphabet \"F\" to represent the resultant force.", "p", "pWRONG SYMBOL USED", 128);
            return; break;
            case 'a': aVal++; break;
            case '-81': PhiVal++; break;
            case 'N': Nval++; break;
            case 'n': ModalDialEqCheck("pYou have used \"n\".", "pUse the capital alphabet \"N\".", "p",
            "pLOWER CASE ALPHABET USED", 128); return; break;
            case 's': if ((*toPrevEntry + i + 1) == 'i' && (*toPrevEntry + i + 2) == 'n') { i += 2; sinVal++;
            } break;
            case 'c': if ((*toPrevEntry + i + 1) == 'o' && (*toPrevEntry + i + 2) == 's') { i += 2; cosVal++;
            } break;
            case 'B': ModalDialEqCheck("pUse Ø instead of ß.", "pØ is the angle of the incline.",
            "pPERPENDICULAR TO THE INCLINE", "pWRONG ANGLE SYMBOL", 128); return; break;
            default: ModalDialEqCheck("pError in equating the forces.", "pStart by equating N to the
            component of the gravitational force perpendicular to the track.\rFor information on components of
            forces, click \"Force Diagram\".", "pPERPENDICULAR TO THE INCLINE", "pEQUATING
            ERROR", 128); return; break; }
    if (sinVal && PhiVal == 1) {
        ModalDialEqCheck("pIt should be \"cos\" instead of \"sin\".", "pFor information on components of
        forces, click \"Force Diagram\".", "pPERPENDICULAR TO THE INCLINE", "pWRONG
        TRIGONOMETRIC FUNCTION", 128); return; }
    if (cosVal && !PhiVal) { ModalDialEqCheck("pArgument of cosine should be \"Ø\".", "pØ is the
    angle of slope of the incline.\rFor information on components of forces, click \"Force Diagram\".",
    "pALONG THE INCLINE", "pWRONG TRIGONOMETRIC FUNCTION", 128); return; }
}

```

```

if (Fval == 1 && Mval == 1 && aVal == 1 && !gVal && !cosVal && !PhiVal && !Nval ) {
    ModalDialogEqCheck("\pYour equation should read F = Ma.", "\pThe resultant force F, perpendicular to
    the plane is zero. Forces in opposite directions balance. \rEquate N to the component of the
    gravitational force. \rFor information on components of forces, click \"Force Diagram\". ",
    "\pPERPENDICULAR TO THE INCLINE", "\pEQUATE OPPOSITE FORCES", 128); return;}
else if (Fval && Mval == 1 && aVal == 1 && !gVal && cosVal == 1 && PhiVal == 1 &&
    !Nval) {
    ModalDialogEqCheck("\pYour equation should read F = MacosØ.", "\pReplace a by g. The component
    of the gravitational force perpendicular to the plane is MgcossØ. \rFor information on components of
    forces, click \"Force Diagram\". ", "\pPERPENDICULAR TO THE INCLINE", "\pSUBSTITUTION
    REQUIRED", 128); return; }
else if (Fval == 1 && Mval == 1 && !aVal && gVal == 1 && cosVal == 1 && PhiVal == 1 &&
    !Nval) {
    ModalDialogEqCheck("\pYour equation should read F = MgcossØ.", "\pReplace F by N. The reaction,
    N, of the plane on the ball is balanced by the component of the gravitational force perpendicular to
    the plane.", "\pPERPENDICULAR TO THE INCLINE", "\pSUBSTITUTION REQUIRED", 128);
    return; }
else if (Mval == 1 && !aVal && gVal == 1 && cosVal == 1 && PhiVal == 1 && !Fval &&
    Nval == 1) {
    ModalDialogEqCheck("\pIf your equation reads N = MgcossØ", "\pIt is correct.",
    "\pPERPENDICULAR TO THE INCLINE", "\pCHECK COMPLETED", 128); return; }
else if (Mval == 1 && aVal == 1 && !gVal && !cosVal && !PhiVal && !Fval && Nval == 1){
    ModalDialogEqCheck("\pN = Ma", "\pReplace a by gcosØ, the component of the gravitational
    acceleration perpendicular to the incline. For information on components, click \"Force Diagram\". ",
    "\pPERPENDICULAR TO THE INCLINE", "\pCHECK COMPLETED", 128); return;}
else if (Mval == 1 && !aVal && gVal == 1 && !cosVal && !PhiVal && !Fval && Nval == 1){
    ModalDialogEqCheck("\pN = Mg", "\pReplace g by gcosØ, the component of the gravitational
    acceleration perpendicular to the incline. For information on components, click \"Force Diagram\". ",
    "\pPERPENDICULAR TO THE INCLINE", "\pCHECK COMPLETED", 128); return;}
else {
    ModalDialogEqCheck("\pError in equating the forces", "\pStart by equating N to the component of the
    gravitational force perpendicular to the plane. For information on components, click \"Force
    Diagram\". ", "\pPERPENDICULAR TO THE INCLINE", "\pEQUATING ERROR", 128); return;}
}

void CForcesEqCheck::NormLoopBot(void)
{
    int i, Mval, gVal, Nval, Fval, aVal, Rval, v2Bot;
    i = Mval = gVal = Nval = Fval = Rval = aVal = v2Bot = 0;

    if (*(toPrevEntry) == 0) {
        ModalDialogEqCheck("\pPlace the equation you wish to check in the \"Check window\".",
        "\pIf you are unsure, click on \"Force Diagram\". \rOtherwise, start with F = ma and click the
        \"CHECK\" button.", "\pPERPENDICULAR TO THE TRACK", "\pNO EQUATION IN CHECK
        WINDOW", 128); return; }
    if (*(toPrevEntry) == '0') {
        ModalDialogEqCheck("\pNO EQUATION IN CHECK WINDOW", "\pIf you are unsure, click on
        \"Force Diagram\". \rOtherwise, start with F = ma and click the \"CHECK\" button.",
        "\pPERPENDICULAR TO THE TRACK", "\pBOTTOM OF THE LOOP", 128); return;}
    if (!IsThereEqual())
        return;
    if (CheckforNumber())
        return;
    for ( i = 0; i <= Length - 1; i++)
        switch (*(toPrevEntry + i)) {
            case '*': case ':': case '=': case '+': case '-': case '/': break;
            case 'M': Mval++; break;
            case 'm': ModalDialogEqCheck("\pYou have used \"m\".", "\pUse the capital alphabet \"M\".", "\p",
            "\pLOWER CASE ALPHABET USED", 128); return; break;
            case 'g': gVal++; break;
            case 'f': ModalDialogEqCheck("\pYou have used \"f\".", "\pUse the capital alphabet \"F\".", "\p",
            "\pLOWER CASE ALPHABET USED", 128); return; break;
            case 'F': Fval++; break;
            case 'a': aVal++; break;
            case -81: ModalDialogEqCheck("\pUse β not Ø.", "\pØ is the angle of the incline. β is the angle the
            radius vector makes with the vertical. Click \"Force diagram\" for more help.", "\pIN THE LOOP",
            "\pREPLACE Ø BY β", 128); return; break;
            case 'N': Nval++; break;
            case 'n': ModalDialogEqCheck("\pYou have used \"n\".", "\pUse the capital alphabet \"N\".",
            "\p", "\pLOWER CASE ALPHABET USED", 128); return; break;
        }
}

```

```

case 'r': ModalDialEqCheck("\pr is the radius of the ball.", "\pUse R, the radius of the loop.",
"\pPERPENDICULAR TO THE TRACK", "\pBOTTOM OF THE LOOP", 128); return; break;
case 'R': Rval++; break;
case 's': if ((*toPrevEntry + i + 1) == 'i' && (*toPrevEntry + i + 2) == 'n') { i += 2;
ModalDialEqCheck("\pWRONG TRIGONOMETRIC FUNCTION", "\pIt should be cos instead of
sin. \rClick on \"Force Diagram\" for more help.", "\pPERPENDICULAR TO THE TRACK",
"\pBOTTOM OF THE LOOP", 128); return; } break;
case 'B': ModalDialEqCheck("\p $\beta = 0$  at the bottom of the loop.",
"\pcos $\beta = 0$ . \rFor more information, click \"Force diagram\".", "\pPERPENDICULAR TO THE
TRACK", "\pBOTTOM OF THE LOOP", 128); return; break;
case 'c': if ((*toPrevEntry + i + 1) == 'o' && (*toPrevEntry + i + 2) == 's') { i += 2;
ModalDialEqCheck("\p $\beta = 0$  at the bottom of the loop.", "\pcos $\beta = 0$ . \rFor more information, click
\"Force diagram\".", "\pPERPENDICULAR TO THE TRACK", "\pBOTTOM OF THE LOOP",
128); return; } break;
case -125: v2Bot++; break;
case -124: case -127: case -110: case -128: case -109: case 'v':
ModalDialEqCheck("\p", "\p", "\p", "\p", 1381); return; break;
case -126: ModalDialEqCheck("\pThe velocity needs to be squared.", "\pSquare the velocity.",
"\pPERPENDICULAR TO THE TRACK", "\pBOTTOM OF THE LOOP", 128); return; break;
case -123: case -122: case -120: case -121: case -119: case -118: ModalDialEqCheck("\pUse the
appropriate translational velocity symbol.", "\pUsing w can cause confusion between the angular
velocity of rotation in the loop and the angular velocity of the rolling ball.", "\pPERPENDICULAR
TO THE TRACK", "\pBOTTOM OF THE LOOP", 128); return; break;
default: ModalDialEqCheck("\p", "\pIf you are unsure start with F = ma and click the \"CHECK\"
button.", "\pPERPENDICULAR TO THE TRACK", "\pBOTTOM OF THE LOOP", 128); return;
break; }
if (Fval == 1 && Mval == 1 && aVal == 1 && gVal && !Nval && !Rval && !v2Bot) {
ModalDialEqCheck("\pYour equation should read F = Ma.", "\pF is the resultant force towards the
centre of the track. \rReplace it by a combination of N and Mg. \rFor more information, click \"Force
Diagram\".", "\pPERPENDICULAR TO THE TRACK", "\pBOTTOM OF THE LOOP", 128);
return; }
else if (!Fval && !aVal && gVal == 1 && Nval == 1 && Rval == 1 && v2Bot == 1
&& (Mval == 2 || Mval == 1)) {
ModalDialEqCheck("\pB", "\p-", "\pPERPENDICULAR TO THE TRACK", "\pBOTTOM OF THE
LOOP", 2007); return; }
else if (!Fval && Mval == 1 && !aVal && gVal == 1 && Nval == 1 && !Rval && !v2Bot) {
ModalDialEqCheck("\p", "\pN is not equal to Mg. \rThere is an acceleration towards the centre of the
loop. \rThe difference of the two forces equals the resultant force Ma.", "\pPERPENDICULAR TO
THE TRACK", "\pBOTTOM OF THE LOOP", 128); return; }
else if (Fval == 1 && Mval == 1 && gVal == 1 && Nval == 1 && !aVal && !Rval &&
!v2Bot) {
ModalDialEqCheck("\pYour equation should read F = N - Mg.", "\pF is the resultant force towards
the centre of the track. Replace it by Ma", "\pPERPENDICULAR TO THE TRACK", "\pBOTTOM
OF THE LOOP", 128); return; }
else if (aVal == 1 && gVal == 1 && Nval == 1 && !Fval && !Rval && !v2Bot &&
(Mval == 2 || Mval == 1)) {
ModalDialEqCheck("\pYour equation should read Ma = N - Mg.", "\pa is the acceleration towards the
centre of the track. Replace it using \"relations\" in \"Need help with\".", "\pPERPENDICULAR TO
THE TRACK", "\pBOTTOM OF THE LOOP", 128); return; }
else if (!aVal && gVal == 1 && Nval == 1 && !Fval && !Rval && v2Bot == 1 && Mval ==
2){
ModalDialEqCheck("\pYour equation is missing the term R.", "\pThe acceleration should be given by
the velocity square divided by R.", "\pPERPENDICULAR TO THE TRACK", "\pBOTTOM OF
THE LOOP", 128); return; }
else if (!aVal && gVal == 1 && Nval == 1 && !Fval && Rval == 1 && v2Bot == 1 && Mval
== 1) {
ModalDialEqCheck("\pYour equation is missing an M.", "\pCheck your working.",
"\pPERPENDICULAR TO THE TRACK", "\pBOTTOM OF THE LOOP", 128); return;}
else if (Fval == 1 && Mval == 1 && v2Bot == 1 && Rval == 1 && !gVal && !aVal && !Nval)
{
ModalDialEqCheck("\pB", "\pF is the resultant force towards the centre of the track. \rReplace it by
the difference of N and Mg. \rFor more information, click \"Force Diagram\".", "\pPERPENDICULAR
TO THE TRACK", "\pBOTTOM OF THE LOOP", 2006); return; }
else if (!Fval && v2Bot == 1 && Rval == 1 && gVal == 1 && !aVal && !Nval) {
ModalDialEqCheck("\pB", "\pThe resultant force towards the centre is not Mg. \rReplace it by the
difference of N and Mg. \rFor more information, click \"Force Diagram\".", "\pPERPENDICULAR TO
THE TRACK", "\pBOTTOM OF THE LOOP", 2011);return;}
else if (!Fval && v2Bot == 1 && Rval == 1 && !gVal && !aVal && Nval == 1) {

```

```

ModalDialEqCheck("\pB" "\pThe resultant force towards the centre is not N. \rReplace it by the
difference of N and Mg. \rFor more information, click \"Force Diagram\".", "\pPERPENDICULAR TO
THE TRACK", "\pBOTTOM OF THE LOOP", 2012); return; }
else
{
ModalDialEqCheck("\pEquation wrong.", "\pChange the equation or start with F = ma and click the
\"CHECK\" button. \rFor more information click \"Force Diagram\" or \"Need help with\".",
"\pPERPENDICULAR TO THE TRACK", "\pBOTTOM OF THE LOOP", 128); return;}
}

void CForcesEqCheck::NormLoopTop(void)
{
int i, Mval, gVal, Nval, Fval, aVal, Rval, v2Top;
i = Mval = gVal = Nval = Fval = aVal = Rval = v2Top = 0;

if ((*toPrevEntry) == 0) {
ModalDialEqCheck("\pPlace the equation you wish to check in the \"Check window\".", "\pIf you are
unsure, click on \"Force Diagram\". \rOtherwise, start with F = ma and click the \"CHECK\" button.",
"\pPERPENDICULAR TO THE TRACK", "\pNO EQUATION IN CHECK WINDOW", 128);\r
return; }
if ((*toPrevEntry) == '0') {
ModalDialEqCheck("\pPlace the equation you wish to check in the \"Check\" window.", "\pIf you are
unsure, click on \"Force Diagram\". \rOtherwise, start with F = ma and click the \"CHECK\" button.",
"\pPERPENDICULAR TO THE TRACK", "\pTOP OF THE LOOP", 128); return; }
if (!IsThereEqual())
return;
if (CheckforNumber())
return;
for ( i = 0; i <= Length - 1; i++)
switch ((*toPrevEntry + i)) {
case '*': case ':': case '=': case '+': case '-': case '/': break;
case 'B': ModalDialEqCheck("\pB = 180 degrees at the top of the loop.", "\pcosB = -1. \rFor more
information, click \"Force diagram\".", "\pPERPENDICULAR TO THE TRACK", "\pTOP OF THE
LOOP", 128);return;break;
case 'M': Mval++;break;
case 'm': ModalDialEqCheck("\pYou have used \"m\".", "\pUse the capital alphabet \"M\".", "\p",
"\pLOWER CASE ALPHABET USED", 128); return; break;
case 'g': gVal++; break;
case 'f': ModalDialEqCheck("\pYou have used \"f\".", "\pUse the capital alphabet \"F\".", "\p",
"\pLOWER CASE ALPHABET USED", 128); return; break;
case 'F': Fval++; break;
case 'a': aVal++; break;
case -81: ModalDialEqCheck("\pØ is the angle of the incline.", "\pB is the angle the radius vector
makes with the vertical. \rClick on \"Force Diagram\" for more help.", "\pIN THE LOOP",
"\pREPLACE Ø BY B", 128); return; break;
case 'n': ModalDialEqCheck("\pYou have used \"n\".", "\pUse the capital alphabet \"N\".", "\p",
"\pLOWER CASE ALPHABET USED", 128); return; break;
case 'N': Nval++; break;
case 'r': ModalDialEqCheck("\pr is the radius of the ball.", "\pUse R the radius of the loop.",
"\pPERPENDICULAR TO THE TRACK", "\pTOP OF THE LOOP", 128); return; break;
case 'R': Rval++; break;
case 's': if ((*toPrevEntry + i + 1) == 'i' && (*toPrevEntry + i + 2) == 'n') { i += 2;
ModalDialEqCheck("\pWRONG TRIGONOMETRIC FUNCTION", "\pIt should be cos instead of
sin. \rClick on \"Force Diagram\" for more help.", "\pPERPENDICULAR TO THE TRACK", "\pTOP
OF THE LOOP", 128); return; } break;
case 'c': if ((*toPrevEntry + i + 1) == 'o' && (*toPrevEntry + i + 2) == 's') { i += 2;
ModalDialEqCheck("\pB = 180 degrees at the top of the loop.", "\pcosB = -1. \rFor more information,
click \"Force diagram\".", "\pPERPENDICULAR TO THE TRACK", "\pTOP OF THE LOOP",
128); return; } break;
case -127: v2Top++; break;
case -124: case -125: case -110: case -126: case -109: ModalDialEqCheck("\p", "\p", "\p", "\p", 1381);
return; break;
case -128: ModalDialEqCheck("\pThe velocity needs to be squared.", "\pSquare the velocity.",
"\pPERPENDICULAR TO THE TRACK", "\pTOP OF THE LOOP", 128); return; break;
case -123: case -122: case -120: case -121: case -119: case -118: ModalDialEqCheck("\pUse the
translational velocity symbol", "\pUsing w can cause confusion between the angular velocity of
rotation in the loop and the angular velocity of the rolling ball.", "\pPERPENDICULAR TO THE
TRACK", "\pTOP OF THE LOOP", 128); return; break;
default: ModalDialEqCheck("\p", "\pIf you are unsure, click on \"Force Diagram\". \rOtherwise, start
with F = ma and click the \"Check\" button.", "\pPERPENDICULAR TO THE TRACK", "\pTOP OF
THE LOOP", 128); return; break; }
}

```

```

if (Fval == 1 && Mval == 1 && !aVal && gVal == 1 && !Nval && !Rval && !v2Top) {
ModalDialEqCheck("\pYour equation should read  $F = Mg$ .", "\pF is the resultant force towards the
centre of the track. \rIt is equal to the sum of N and Mg. \rFor more information, click \"Force
diagram\".", "\pPERPENDICULAR TO THE TRACK", "\pTOP OF THE LOOP", 128);
return; }
if (!Fval && Mval == 1 && !aVal && gVal == 1 && Nval == 1 && !Rval && !v2Top) {
ModalDialEqCheck("\pN is not equal to Mg.", "\pF the resultant force towards the centre of the track
is equal to the sum of N and Mg. The equation should be  $F = N + Mg$ . \rFor more information, click
\"Force diagram\".", "\pPERPENDICULAR TO THE TRACK", "\pTOP OF THE LOOP", 128);
return; }
if (Fval == 1 && Mval == 1 && aVal == 1 && !gVal && !Nval && !Rval && !v2Top) {
ModalDialEqCheck("\pYour equation should read  $F = Ma$ .", "\pF is the resultant force towards the
centre of the track. \rReplace it by the sum of N and Mg. \rFor more information, click \"Force
diagram\".", "\pPERPENDICULAR TO THE TRACK", "\pTOP OF THE LOOP", 128);
return; }
else if (Fval == 1 && Mval == 1 && !aVal && gVal == 1 && Nval == 1 && !Rval && !v2Top)
{
ModalDialEqCheck("\pYour equation should read  $F = N + Mg$ .", "\pF is the resultant force towards
the centre of the track. \rReplace it by  $Ma$ ", "\pPERPENDICULAR TO THE TRACK", "\pTOP OF
THE LOOP", 128); return; }
else if (!Fval && Mval == 2 && aVal == 1 && gVal == 1 && Nval == 1 && !Rval &&
!v2Top) {
ModalDialEqCheck("\pYour equation should read  $Ma = N + Mg$ .", "\p\"a\" is the acceleration towards
the centre of the track. \rReplace it using the \"relationship\" given in \"Need help with\".",
"\pPERPENDICULAR TO THE TRACK", "\pTOP OF THE LOOP", 128); return; }
else if (!Fval && !aVal && gVal == 1 && Nval == 1 && !Rval && v2Top == 1 && (Mval == 1
|| Mval == 2)) {
ModalDialEqCheck("\pYour equation should include R.", "\pThe acceleration is the square of the
velocity divided by R.", "\pPERPENDICULAR TO THE TRACK", "\pTOP OF THE LOOP", 128);
return; }
else if (!Fval && Mval == 1 && !aVal && gVal == 1 && Nval == 1 && Rval == 1 && v2Top
== 1) {
ModalDialEqCheck("\pYour equation is missing an M.", "\pCheck your
working.", "\pPERPENDICULAR TO THE TRACK", "\pTOP OF THE LOOP", 128); return; }
else if (Fval == 1 && Mval == 1 && v2Top == 1 && Rval == 1 && !gVal && !aVal && !Nval)
{
ModalDialEqCheck("\pT", "\pF is the resultant force towards the centre of the track. Replace it by N
and the gravitational force Mg. \rFor more information, click \"Force Diagram\".",
"\pPERPENDICULAR TO THE TRACK", "\pTOP OF THE LOOP", 2006); return; }
else if (Nval == 1 && v2Top == 1 && Rval == 1 && gVal == 1 && (Mval == 2 || Mval ==
1)) {
ModalDialEqCheck("\pT", "\p+", "\pPERPENDICULAR TO THE TRACK", "\pTOP OF THE
LOOP", 2007); return; }
else if (!Fval && v2Top == 1 && Rval == 1 && gVal == 1 && !aVal && !Nval) {
ModalDialEqCheck("\pT", "\pThe resultant force towards the centre is not Mg. \rReplace it by the
sum of N and Mg. \rFor more information, click \"Force Diagram\".", "\pPERPENDICULAR TO THE
TRACK", "\pTOP OF THE LOOP", 2011); return; }
else if (!Fval && v2Top == 1 && Rval == 1 && !gVal && !aVal && Nval == 1) {
ModalDialEqCheck("\pT", "\pThe resultant force towards the centre is not N. \rReplace it by the sum
of N and Mg. \rFor more information, click \"Force Diagram\".", "\pPERPENDICULAR TO THE
TRACK", "\pTOP OF THE LOOP", 2012); return; }
else {
ModalDialEqCheck("\pEquation wrong.", "\pChange the equation or start with  $F = Ma$  and click the
\"CHECK\" button. \rFor more information click \"Force Diagram\" or \"Need help
with\".", "\pPERPENDICULAR TO THE TRACK", "\pTOP OF THE LOOP", 128); return; }
}

void CForgesEqCheck::TanLoopTopBot(void)
{
if (noFriction)
if ((*toPrevEntry) == 0 || (*toPrevEntry) == '0')
((CEditText*) gGopher)->SetTextString("\p0");
else {
ModalDialEqCheck("\pThere is no force in the tangential or horizontal direction.", "\pThe component
of the gravitational force in the horizontal direction is zero.", "\pTANGENTIAL TO THE TRACK",
"\pAT THE BOTTOM AND TOP OF THE LOOP", 128); return; }
if (friction)
if ((*toPrevEntry) == 0 || (*toPrevEntry) == '0')
((CEditText*) gGopher)->SetTextString("\p0");
}

```

```

else {
ModalDialEqCheck( "\pThere is no force in the tangential or horizontal direction.",
"\pThe gravitational force is in the vertical direction. \rThe frictional force is zero when there is no net
force in the tangential direction.", "\pTANGENTIAL TO THE TRACK", "\pAT THE BOTTOM
AND TOP OF THE LOOP", 128); return; }
}

```

CForcesEquations.c

(Further methods for checking equations)

#include the following: "Commands.h", "CForcesEquations.h", "CLoopApp.h", <CDesktop.h>, <CEditText.h>, <TBUtilities.h>, "CForcesEqCheck.h", <CDialog.h>

extern CBureaucrat *gGopher; extern Handle toPrevEntry; extern long Length; extern int friction, noFriction;

```

void CForcesEquations::TanLoopGen(void)
{
    CForcesEqCheck *theFunction;
    int i, Mval, gVal, Fval, aVal, sinVal, Bval, cosVal, frict;
    i = Mval = gVal = Fval = aVal = sinVal = Bval = cosVal = frict = 0;

    if ((*toPrevEntry) == 0) {
        ModalDialEqChTwo("\pPlace the equation you wish to check in the \"Check window\". ", "\pIf you
        are unsure, click \"Force Diagram\". \rOtherwise, start with F = ma and click the \"CHECK\"
        button.", "\pTANGENTIAL TO THE TRACK", "\pGENERALLY IN THE LOOP", 128); return; }
    if ((*toPrevEntry) == '0') {
        ModalDialEqChTwo("\pYou have placed \"0\" in the Check window.", "\pPlace the equation you
        wish to check in the \"Check window\". \rIf you are unsure, click \"Force Diagram\". \rOtherwise,
        start with F = ma and click the \"CHECK\" button.", "\pTANGENTIAL TO THE TRACK",
        "\pGENERALLY IN THE LOOP", 128); return; }
    theFunction = new (CForcesEqCheck);
    if (!(theFunction->IsThereEqual()))
        return;
    if (theFunction->CheckforNumber())
        return;
    for (i = 0; i <= Length - 1; i++)
        switch ((*toPrevEntry + i)) {
            case '*': case ' ': case '=': case '-': case '+': break;
            case 'M': Mval++; break;
            case 'm': ModalDialEqChTwo("\pYou have used \"m\".", "\pUse the capital alphabet \"M\".", "\p",
            "\pLOWER CASE ALPHABET USED", 128); return; break;
            case 'g': gVal++; break;
            case 'f': frict++; break;
            case 'F': Fval++; break;
            case 'a': aVal++; break;
            case -81: ModalDialEqChTwo("\pREPLACE Ø BY β", "\pØ is the angle of the incline. β is the
            angle the radius vector makes with the vertical. \rFor more information, click \"Force diagram\".",
            "\pTANGENTIAL TO THE TRACK", "\pGENERALLY IN THE LOOP", 128); return; break;
            case 's': if ((*toPrevEntry + i + 1) == 'i' && (*toPrevEntry + i + 2) == 'n') { i += 2; sinVal++;
            } break;
            case 'c': if ((*toPrevEntry + i + 1) == 'o' && (*toPrevEntry + i + 2) == 's') { i += 2; cosVal++;
            } break;
            case 'B': Bval++; break;
            case 'n': ModalDialEqChTwo("\pYou have used \"n\".", "\pUse the capital alphabet \"N\".", "\p",
            "\pLOWER CASE ALPHABET USED", 128); return; break;
            case 'N': ModalDialEqChTwo("\pN is the normal force of the track on the ball.", "\pIt acts
            perpendicular to the track and towards the centre. \rYou are considering motion tangential to the
            track.", "\pALONG THE INCLINE", "\pWRONG FORCE", 128); return; break;
            case 'v': case -128: case -127: case -126: case -125: case -124: case -110: case -109:
            ModalDialEqChTwo("\pUse acceleration, a instead of velocity.", "\pThere is no simple relationship
            connecting the tangential acceleration to the velocity. \rFor more information click \"Force Diagram\".",
            "\pTANGENTIAL TO THE TRACK", "\pGENERALLY IN THE LOOP", 128); return; break;
            case -123: case -122: case -121: case -120: case -119: case -118: ModalDialEqChTwo("\pUse
            translational quantities.", "\pUse acceleration to write the equation. \rFor more information click
            \"Force Diagram\".", "\pTANGENTIAL TO THE TRACK", "\pGENERALLY IN THE LOOP", 128);
            return; break;
        }
}

```

```

default: ModalDialEqChTwo("\pEquation wrong.", "\pChange the equation or start with  $F = ma$  and
click the \"CHECK\" button. \rFor more information click \"Force Diagram\" or \"Need help with\".",
"\pTANGENTIAL TO THE TRACK", "\pGENERALLY IN THE LOOP", 128); return; break; }
if (noFriction) {
if (cosVal) {
ModalDialEqChTwo("\pWRONG TRIGONOMETRIC FUNCTION", "\pIt should be \"sin\" instead
of \"cos\". \rFor help, click \"Force Diagram\".", "\pTANGENTIAL TO THE TRACK",
"\pGENERALLY IN THE LOOP", 128); return; }
else if (Fval == 1 && Mval == 1 && aVal == 1 && !gVal && !sinVal && !Bval) {
ModalDialEqChTwo("\pYour equation should read  $F = Ma$ .", "\pReplace F by the component of
Mg
along the tangent. For information on components of forces, click \"Force Diagram\".",
"\pTANGENTIAL TO THE TRACK", "\pGENERALLY IN THE LOOP", 128); return; }
else if (Fval == 1 && Mval == 1 && !aVal && gVal == 1 && !sinVal && !Bval) {
ModalDialEqChTwo("\pYour equation is incorrect.", "\pReplace Mg by -Mgsin $\beta$ , its component in
the tangential direction. \rFor information on components of forces, click \"Force Diagram\".",
"\pTANGENTIAL TO THE TRACK", "\pGENERALLY IN THE LOOP", 128); return; }
else if (!Fval && aVal == 1 && gVal == 1 && !sinVal && !Bval) {
ModalDialEqChTwo("\pThe equation is incorrect.", "\pa equals -gsin $\beta$ , the component of the
gravitational acceleration along the tangential direction. \rFor information on components of forces,
click \"Force Diagram\".", "\pTANGENTIAL TO THE TRACK", "\pGENERALLY IN THE
LOOP", 128); return; }
else if (Fval == 1 && Mval == 1 && !aVal && gVal == 1 && sinVal == 1 && Bval == 1) {
ModalDialEqChTwo("\pYour equation should read  $F = -Mgsin\beta$ .", "\pSubstitute using  $F = Ma$ .
For
information on components of forces, click \"Force Diagram\".", "\pTANGENTIAL TO THE
TRACK", "\pGENERALLY IN THE LOOP", 128); return; }
else if (Mval == 2 && aVal == 1 && gVal == 1 && sinVal == 1 && Bval == 1 && !Fval) {
ModalDialEqChTwo("\pYour equation should read  $Ma = -Mgsin\beta$ .", "\pSimplify the equation by
dividing it by M.", "\pTANGENTIAL TO THE TRACK", "\pGENERALLY IN THE LOOP",
128); return; }
else if (aVal == 1 && gVal == 1 && sinVal == 1 && Bval == 1 && !Fval) {
ModalDialEqChTwo("\pThe correct equation is:  $a = -gsin\beta$ ", "\pa is the tangential acceleration.",
"\pTANGENTIAL TO THE TRACK", "\pCHECK COMPLETED", 128); return; }
else {
ModalDialEqChTwo("\pEquation wrong.", "\pChange the equation or start with  $F = ma$  and click
the \"CHECK\" button. \rFor more information click \"Force Diagram\" or \"Need help with\".",
"\pTANGENTIAL TO THE TRACK", "\pGENERALLY IN THE LOOP", 128); return; }
}
if (friction) {
if (cosVal) {
ModalDialEqChTwo("\pINCORRECT TRIGONOMETRIC FUNCTION", "\pIt should be \"sin\"
instead of \"cos\". \rFor help, click \"Force Diagram\".", "\pTANGENTIAL TO THE TRACK",
"\pGENERALLY IN THE LOOP", 128); return; }
else if (Fval == 1 && Mval == 1 && aVal == 1 && !gVal && !sinVal && !Bval) {
ModalDialEqChTwo("\pYour equation should read  $F = Ma$ .", "\pReplace F by a combination of the
frictional force (f) and the component of Mg tangential to the track. For information on
components of forces, click \"Force Diagram\".", "\pTANGENTIAL TO THE TRACK",
"\pGENERALLY IN THE LOOP", 128); return; }
else if (!Fval && Mval == 1 && aVal == 1 && gVal == 1 && sinVal == 1 && Bval == 1 &&
frict
== 1) {
ModalDialEqChTwo("\pYour equation is correct if it reads  $Ma = -Mgsin\beta - f$ .", "\pCHECK
COMPLETED. \rThe component of Mg and the frictional force act opposite to the direction of
motion. For information on components of forces, click \"Force Diagram\".", "\pTANGENTIAL
TO THE TRACK", "\pGENERALLY IN THE LOOP", 128); return; }
else if (!Fval && Mval == 1 && aVal == 1 && gVal == 1 && !sinVal && !Bval && frict == 1)
{
ModalDialEqChTwo("\pYour equation should read  $Ma = -Mg - f$ .", "\pReplace Mg by Mgsin $\beta$ .
The negative signs indicate that the component of Mg, and the frictional force act opposite to the
direction of acceleration. For information on components of forces, click \"Force Diagram\".",
"\pTANGENTIAL TO THE TRACK", "\pGENERALLY IN THE LOOP", 128); return; }
else if (Fval == 1 && Mval == 1 && !aVal && gVal == 1 && !sinVal && !Bval && !frict) {
ModalDialEqChTwo("\pYour equation is incorrect.", "\pReplace Mg by -Mgsin $\beta$ , its component in
the tangential direction and include the frictional force. \rFor information on components of forces,
click \"Force Diagram\".", "\pTANGENTIAL TO THE TRACK", "\pGENERALLY IN THE
LOOP", 128); return; }
else if (Fval == 1 && Mval == 1 && !aVal && gVal == 1 && sinVal == 1 && Bval == 1 &&
!frict) {

```

```

ModalDialEqChTwo("\pYour equation should read  $F = -Mg\sin\beta$ ", "\pInclude the frictional force.
For information on components of forces, click '\Force Diagram\'", "\pTANGENTIAL TO THE
TRACK", "\pGENERALLY IN THE LOOP", 128); return; }
else if (Mval == 2 && aVal == 1 && gVal == 1 && sinVal == 1 && Bval == 1 && !Fval &&
!frict) {
    ModalDialEqChTwo("\pYour equation should read  $Ma = -Mg\sin\beta$ ", "\pInclude the frictional force
into the equation.", "\pTANGENTIAL TO THE TRACK", "\pGENERALLY IN THE LOOP",
128); return; }
else {
    ModalDialEqChTwo("\pEquation wrong.", "\pChange the equation or start with  $F = ma$  and click
the '\CHECK\' button. \rFor more information click '\Force Diagram\' or '\Need help with\'",
"\pTANGENTIAL TO THE TRACK", "\pGENERALLY IN THE LOOP", 128); return; }
}

void    CForgesEquations::ModalDialEqChTwo(Str255 theText0, Str255 theText1, Str255
        theText2, Str255 theText3, short IdNum)
{
    ParamText(theText0, theText1, theText2, theText3); PositionDialog('ALRT', IdNum); InitCursor;
    NoteAlert(IdNum, NULL); return; }

void    CForgesEquations::NormLoopGen(void)
{
    CForgesEqCheck *theFunction;
    int i, Mval, gVal, Nval, Fval, aVal, Bval, cosVal, Rval, v2Gen, sinVal;
    i = Mval = gVal = Nval = Fval = aVal = Bval = cosVal = Rval = v2Gen = sinVal = 0;
    if ((*toPrevEntry) == 0) {
        ModalDialEqChTwo("\pPlace the equation you wish to check in the '\Check window\'", "\pIf you
are unsure, click '\Force Diagram\' \r Otherwise, start with  $F = ma$  and click the '\CHECK\'
button.", "\pPERPENDICULAR TO THE TRACK", "\pNO EQUATION IN CHECK WINDOW",
128); return; }
    if ((*toPrevEntry) == '0') {
        ModalDialEqChTwo("\pNO EQUATION IN CHECK WINDOW", "\pIf you are unsure, click
'\Force Diagram\' \r Otherwise, start with  $F = ma$  and click the '\CHECK\' button.",
"\pPERPENDICULAR TO THE TRACK", "\pGENERALLY IN THE LOOP", 128); return; }
    theFunction = new (CForgesEqCheck);
    if (!(theFunction->IsThereEqual()))
        return;
    if (theFunction->CheckforNumber())
        return;
    for (i = 0; i <= Length - 1; i++)
        switch ((*toPrevEntry + i)) {
            case '*': case ' ': case '=': case '+': case '-': case '/': break;
            case 'M': Mval++; break;
            case 'm': ModalDialEqChTwo("\pYou have used '\m\'", "\pUse the capital alphabet '\M\'", "\p",
"\pLOWER CASE ALPHABET USED", 128); return; break;
            case 'g': gVal++; break;
            case 'f': ModalDialEqChTwo("\pf is the symbol for frictional force.", "\pReplace it with the symbol
for the resultant force F.\rFor a rolling ball, the frictional force acts in the tangential direction.", "\p",
"\pLOWER CASE ALPHABET USED", 128); return; break;
            case 'F': Fval++; break;
            case 'a': aVal++; break;
            case -81: ModalDialEqChTwo("\p $\emptyset$  is the angle of the incline.", "\p $\beta$  is the angle the radius vector
makes with the vertical. For more information, click '\Force Diagram\'", "\pIN THE LOOP",
"\pREPLACE  $\emptyset$  BY  $\beta$ ", 128); return; break;
            case 'N': Nval++; break;
            case 'r': ModalDialEqChTwo("\pr is the radius of the ball.", "\pUse R, the radius of the loop.", "\p",
"\pLOWER CASE ALPHABET USED", 128); return; break;
            case 'R': Rval++; break;
            case 's': if ((*toPrevEntry + i + 1) == 'i' && (*toPrevEntry + i + 2) == 'n') { i += 2; sinVal++;
} break;
            case 'c': if ((*toPrevEntry + i + 1) == 'o' && (*toPrevEntry + i + 2) == 's') { i += 2; cosVal++;
} break;
            case 'B': Bval++; break;
            case -124: v2Gen++; break;
            case -127: case -125: case -110: case -128: case -126: case -109: ModalDialEqChTwo("\p", "\p", "\p",
"\p", 1381); return; break;
            case 'v': ModalDialEqChTwo("\pThe velocity needs to be squared.", "\pSquare the velocity.",
"\pPERPENDICULAR TO THE TRACK", "\pGENERALLY IN THE LOOP", 128); return; break;

```



```

case -123: case -122: case -120: case -121: case -119: case -118: ModalDialEqChTwo("pUse the
appropriate translational velocity symbol.", "pUsing w can cause confusion between the angular
velocity of rotation in the loop and the angular velocity of the rolling ball.", "pPERPENDICULAR
TO THE TRACK", "pGENERALLY IN THE LOOP", 128); return; break;
default: ModalDialEqChTwo("p", "pChange the equation or start with  $F = ma$  and click the
\CHECK\ button. rFor more information click \Force Diagram\ or \Need help with\.",
"pPERPENDICULAR TO THE TRACK", "pGENERALLY IN THE LOOP", 128); return; break; }
if (sinVal) {
    ModalDialEqChTwo("pWRONG TRIGONOMETRIC FUNCTION", "pIt should be \cos\ instead
of \sin\.", "pFor information on components of forces, click \Force Diagram\.",
"pPERPENDICULAR TO THE TRACK", "pGENERALLY IN THE LOOP", 128); return; }
else if (!cosVal || !Bval) {
    ModalDialEqChTwo("pYour equation should have  $\cos\beta$ .", "pMgcos $\beta$  is the component of the
gravitational force perpendicular to the track. rFor information on components of forces, click
\Force Diagram\.", "pPERPENDICULAR TO THE TRACK", "pGENERALLY IN THE
LOOP", 128); return; }
else if (!Fval && Mval == 1 && gVal == 1 && Nval == 1 && !Rval && !v2Gen && !aVal &&
cosVal == 1 && Bval == 1) {
    ModalDialEqChTwo("pIf your equation reads  $N = Mg\cos\beta$ , it is incorrect.", "pThe difference of N
and  $Mg\cos\beta$  equals the resultant force F.", "pPERPENDICULAR TO THE TRACK",
"pGENERALLY IN THE LOOP", 128); return; }
else if (Fval == 1 && Mval == 1 && !aVal && gVal == 1 && !Nval && !Rval && !v2Gen) {
    ModalDialEqChTwo("pYour equation should read  $F = Mg\cos\beta$ .", "pThe resultant force F towards
the centre of the track equals the difference of N and  $Mg\cos\beta$ ", "pPERPENDICULAR TO THE
TRACK", "pGENERALLY IN THE LOOP", 128); return; }
else if (Fval == 1 && Mval == 1 && aVal == 1 && !gVal && !Nval && !Rval && !v2Gen) {
    ModalDialEqChTwo("pYour equation should read  $F = Ma$ .", "pF is the resultant force towards the
centre of the track. Replace it by N and the gravitational force component perpendicular to the track.
rFor information on components of forces, click \Force Diagram\.", "pPERPENDICULAR TO
THE TRACK", "pGENERALLY IN THE LOOP", 128); return; }
else if (Fval == 1 && Mval == 1 && gVal == 1 && Nval == 1 && !Rval && !v2Gen && !aVal)
{
    ModalDialEqChTwo("pYour equation should read  $F = N - Mg\cos\beta$ .", "pF is the resultant force
towards the centre of the track. rReplace F by Ma.", "pPERPENDICULAR TO THE TRACK",
"pGENERALLY IN THE LOOP", 128); return; }
else if (aVal == 1 && Mval == 2 && gVal == 1 && Nval == 1 && cosVal == 1 && Bval == 1)
{
    ModalDialEqChTwo("pYour equation should read  $Ma = N - Mg\cos\beta$ .", "pa is the acceleration
towards the centre of the track. Substitute for \a\ using \relations\ given in \Need help
with\.", "pPERPENDICULAR TO THE TRACK", "pGENERALLY IN THE LOOP", 128);
return; }
else if (!aVal && Mval == 1 && gVal == 1 && !Nval && Fval == 1 && cosVal == 1 && Bval
== 1 && !Rval && !v2Gen) {
    ModalDialEqChTwo("pN needs to be included.", "pFor information on components of forces, click
\Force Diagram\.", "pPERPENDICULAR TO THE TRACK", "pGENERALLY IN THE
LOOP", 128); return; }
else if (Fval == 1 && Mval == 1 && v2Gen == 1 && Rval == 1 && !gVal && !aVal &&
!Nval) {
    ModalDialEqChTwo("p", "pF is the resultant force towards the centre of the track. Replace it by N
and the component of the gravitational force perpendicular to the track. rFor information on
components of forces, click \Force Diagram\.", "pPERPENDICULAR TO THE TRACK",
"pGENERALLY IN THE LOOP", 2006); return; }
else if (!Fval && v2Gen == 1 && Rval == 1 && gVal == 1 && !aVal && !Nval && (Mval ==
2 || !Mval)) {
    ModalDialEqChTwo("pN needs to be included in the equation." "pThe resultant force is the
difference between N and the component of the gravitational force perpendicular to the track. rFor
information on components of forces, click \Force Diagram\.", "pPERPENDICULAR TO THE
TRACK", "pGENERALLY IN THE LOOP", 128); return; }
else if (!Fval && v2Gen == 1 && Rval == 1 && !gVal && !aVal && Nval == 1) {
    ModalDialEqChTwo("p", "pReplace N by the resultant force towards the centre. rThe resultant
force is the difference between N and the component of the gravitational force perpendicular to the
track. rFor more information, click \Force Diagram\.", "pPERPENDICULAR TO THE
TRACK", "pGENERALLY IN THE LOOP", 2012); return; }
else if (Nval == 1 && v2Gen == 1 && Rval == 1 && gVal == 1 && !Fval && !aVal && (Mval
== 2 || Mval == 1)) {
    ModalDialEqChTwo("p", "pCHECK COMPLETED. rIf you wish to use the equation to solve the
problem, go to the \Start Panel\ and select \Solve the question\.", "pPERPENDICULAR TO
THE TRACK", "pGENERALLY IN THE LOOP", 2008); return; }
else {

```

```

ModalDialEqChTwo("\p", "\pIf you are unsure start with F = ma and click on \"Check\" button or
click on \"Force Diagram\".", "\pPERPENDICULAR TO THE TRACK", "\pGENERALLY IN
THE LOOP", 128); return; }
}

```

CForcesPane.c

(Methods for the interactive force drawings)

```

#include the following: "CForcesPane.h", <Commands.h>, <Constants.h>, "CLoopDoc.h",
"AppCommands.h", "Global.h", <SANE.h>, <ctype.h>, <math.h>, <errno.h>, <CDialog.h>,
"CForcesSimulate.h", "CForcesPaneSec.h"

```

```

extern CForcesPane *theTrackForces; extern double Pi, X, Y, X0, Y0; double Rad, Phi, verHt,
slopeLen; int Angle, ballSize, Left, Top, Right, Bottom; int heightNum, ForcesTrackDrawn,
InclineBall, LoopBall, HorizBall, Acceleration, ResForces, TotForces; extern int Angle, posForces,
drawRollSlid;

```

```

void CForcesPane::IForcesPane(short resID, CView *anEnclosure, CBureaucrat *aSupervisor)
{
    CPane::IViewRes('Pane', resID, anEnclosure, aSupervisor);
}

```

```

void CForcesPane::Draw(Rect *area)
{ CForcesSimulate *theAccel; CursHandle gWatchCursor;

```

```

    gWatchCursor = GetCursor(watchCursor); SetCursor(*gWatchCursor); theTrackForces->Prepare();if
(ForcesTrackDrawn) DrawForcesTrack(ForcesPane, 40, 20);
if (InclineBall) DrawingInclineBall(40, 20, 1, heightNum);
if (LoopBall) DrawingLoopBall(40, 20, 1, Angle);
if (HorizBall) DrawingHorzBall(40, 20, 1);
if (Acceleration) {
    theAccel = new (CForcesSimulate); theAccel->DrawAcceleration(); }
inherited::Draw(area); InitCursor(); }

```

```

void CForcesPane::DrawForcesTrack(short PaneId, short startX, short startY)
{ Rect Circle; int i = 0;
ballSize = 14; verHt = 150; Rad = 50; Phi = Pi/4; slopeLen = (verHt - Rad*(1 -
cos(Phi)))/sin(Phi);
MoveTo(startX, startY); X = slopeLen*cos(Phi); Y = slopeLen*sin(Phi); LineTo (X + startX, Y +
startY); X = X + startX; Y = Y + startY; Left = X + Rad * sin(Phi) - Rad; Top = Y - Rad *
cos(Phi) - Rad; Right = Left + 2 * Rad; Bottom = Top + 2 * Rad; SetRect(&Circle, Left, Top,
Right, Bottom); FrameOval(&Circle); X = Left + Rad; Y = Top + 2 * Rad; MoveTo(X, Y);
LineTo(ForcesSimPaneWidth, Y); MoveTo(X, Y); X0 = X;
for (i = 0; i < 40; i++) {
    Move(-4, 0); LineTo(X0 - 8, Y); X0 -= 8; }
X0 = X - Rad*tan(Phi/2)*cos(Phi) - Rad*tan(Phi/2); Y0 = Y - Rad*tan(Phi/2)*sin(Phi);
MoveTo(X0, Y0); LineTo(X0 - 5, Y); MoveTo(X0 - 20, (Y0 + Y)/2); DrawString("\pØ");
ForcesTrackDrawn = 1; }

```

```

void CForcesPane::FrameForcesBall(void)
{ Rect forceBall;
Right = Left + ballSize; Bottom = Top + ballSize; SetRect(&forceBall, Left, Top, Right, Bottom);
FrameOval(&forceBall); }

```

```

void CForcesPane::DrawingInclineBall(short startX, short startY, short TrackWidth, int height)
{ int aHor = 0, aVert = 0;
heightNum = height; Angle = height/tan(Phi) + startX + TrackWidth +
ballSize*sin(Phi)/2; Left = X - ballSize/2 + 1; Y = height + startY - ballSize*cos(Phi)/2;
Top = Y - ballSize/2; FrameForcesBall(); aHor = X - ballSize * sin(Phi)/2; aVert = Y + ballSize *
cos(Phi)/2; MoveTo(aHor, aVert); LineTo(aHor + 20, aVert - 20); LineTo(aHor + 15, aVert - 20);
MoveTo(aHor + 20, aVert - 20); LineTo(aHor + 20, aVert - 15); MoveTo(aHor + 25, aVert - 20);
DrawString("\pN"); MoveTo(X, Y);
if (TotForces) {
    LineTo(X, Y + 40); LineTo(X - 4, Y + 36); MoveTo(X, Y + 40); LineTo(X + 4, Y + 36);
    MoveTo(X - 8, Y + 50); DrawString("\pMg"); }
if (ResForces) {
    LineTo(X - 15, Y + 15); LineTo(X - 15, Y + 10); MoveTo(X - 15, Y + 15); LineTo(X - 10, Y +
15); MoveTo(X - 55, Y + 28); DrawString("\pMgcosØ"); MoveTo(X, Y); LineTo(X + 15, Y + 15);
}

```

```

LineTo(X + 15, Y + 10); MoveTo(X + 15, Y + 15); LineTo(X + 10, Y + 15); MoveTo(X + 18, Y
+ 16); DrawString("\pMgsinØ"); }
if (drawRollSlid) {
MoveTo(X - ballSize*sin(Pi/4)/2 + 1, Y + ballSize*cos(Pi/4)/2 - 1); LineTo(X -
ballSize*sin(Pi/4)/2 - 14, Y + ballSize*cos(Pi/4)/2 - 16); LineTo(X - ballSize*sin(Pi/4)/2 - 9, Y +
ballSize*cos(Pi/4)/2 - 16); MoveTo(X - ballSize*sin(Pi/4)/2 - 14, Y + ballSize*cos(Pi/4)/2 - 16);
LineTo(X - ballSize*sin(Pi/4)/2 - 14, Y + ballSize*cos(Pi/4)/2 - 11); MoveTo(X -
ballSize*sin(Pi/4)/2 - 8, Y + ballSize*cos(Pi/4)/2 - 15); DrawString("\pf"); }
InclineBall = 1; }

```

```

void CFForcesPane::DrawingLoopBall(short startX, short startY, short TrackWidth, double Beta)
{ CFForcesPaneSec *theForcesDraw;
slopeLen = (verHt - Rad*(1 - cos(Phi)))/sin(Phi); Angle = Beta; Beta = Beta * Pi/ 180;
if (!Beta) X0 = startX + TrackWidth + ballSize*sin(Phi)/2 - 2;
else X0 = startX + TrackWidth + ballSize*sin(Phi)/2;
X = X0 + (Rad - ballSize/2 - 1)*(sin(Beta) - sin(-Phi)) + slopeLen*cos(Phi); Y0 = startY -
ballSize*cos(Phi)/2; Y = Y0 + (Rad - ballSize/2 - 1)*(cos(Beta) - cos(-Phi)) + slopeLen*sin(Phi);
Left = X - ballSize/2; Top = Y - ballSize/2; FrameForcesBall();
switch (Angle) {
case 0: theForcesDraw = new (CFForcesPaneSec); theForcesDraw->ForcesDrawBot(); break;
case 180: theForcesDraw = new (CFForcesPaneSec); theForcesDraw->ForcesDrawTop(); break;
case 45: theForcesDraw = new (CFForcesPaneSec); theForcesDraw->ForcesDrawLRQ(); break;
case 135: theForcesDraw = new (CFForcesPaneSec); theForcesDraw->ForcesDrawURQ(); break;
case 225: theForcesDraw = new (CFForcesPaneSec); theForcesDraw->ForcesDrawULQ(); break;
case 315: theForcesDraw = new (CFForcesPaneSec); theForcesDraw->ForcesDrawLLQ(); break; }
LoopBall = 1;}

```

```

void CFForcesPane::DrawingHorBall(short startX, short startY, short TrackWidth)
{ Angle = 360; Left = startX + slopeLen*cos(Phi) + 3*Rad; Top = startY + Rad - Rad * cos(Phi)
+ slopeLen*sin(Phi) - ballSize - TrackWidth; Right = Left + ballSize; Bottom = Top + ballSize;
FrameForcesBall(); X = Left + ballSize/2; Y = Top + ballSize/2; MoveTo(X, Y); LineTo(X, Y -
28);
LineTo(X - 4, Y - 24); MoveTo(X, Y - 28); LineTo(X + 4, Y - 24); MoveTo(X - 4, Y - 30);
DrawString("\pN"); MoveTo(X, Y); LineTo(X, Y + 28); LineTo(X - 4, Y + 24); MoveTo(X, Y +
28); LineTo(X + 4, Y + 24); MoveTo(X + 6, Y + 28); DrawString("\pMg"); HorizBall = 1; }

```

```

void CFForcesPane::DrawMgLine(void)
{ DrawString("\pN"); MoveTo(X, Y); LineTo(X, Y + 39); LineTo(X - 4, Y + 35); MoveTo(X, Y +
39); LineTo(X + 4, Y + 35); MoveTo(X + 7, Y + 39); DrawString("\pMg"); }

```

CFForcesPane.c

(Methods for the interactive force drawings)

```

#include the following: "CFForcesPane.h", <Commands.h>, <Constants.h>, "CLoopDoc.h",
"AppCommands.h", "Global.h", <SANE.h>, <ctype.h>, <math.h>, <errno.h>, <CDialog.h>,
"CFForcesSimulate.h", "CFForcesPaneSec.h"

```

```

extern CFForcesPane *theTrackForces; extern double Pi, X, Y, X0, Y0; double Rad, Phi, verHt,
slopeLen; int Angle, ballSize, Left, Top, Right, Bottom; int heightNum, ForcesTrackDrawn,
InclineBall, LoopBall, HorizBall, Acceleration, ResForces, TotForces; extern int Angle, posForces,
drawRollSlid;

```

```

void CFForcesPane::IForcesPane(short resID, CView *anEnclosure, CBureaucrat *aSupervisor)
{ CPane::IViewRes('Pane', resID, anEnclosure, aSupervisor); }

```

```

void CFForcesPane::Draw(Rect *area)
{ CFForcesSimulate *theAccel; CursHandle gWatchCursor;

```

```

gWatchCursor = GetCursor(watchCursor); SetCursor(*gWatchCursor); theTrackForces->Prepare();if
(ForcesTrackDrawn) DrawForcesTrack(ForcesPane, 40, 20);
if (InclineBall) DrawingInclineBall(40, 20, 1, heightNum);
if (LoopBall) DrawingLoopBall(40, 20, 1, Angle);
if (HorizBall) DrawingHorBall(40, 20, 1);
if ( Acceleration) {
theAccel = new (CFForcesSimulate); theAccel->DrawAcceleration(); }
inherited::Draw(area); InitCursor(); }

```

```

void CForcesPane::DrawForcesTrack(short PanelId, short startX, short startY)
{ Rect Circle; int i = 0;
ballSize = 14; verHt = 150; Rad = 50; Phi = Pi/4; slopeLen = (verHt - Rad*(1 -
cos(Phi)))/sin(Phi);
MoveTo(startX, startY); X = slopeLen*cos(Phi); Y = slopeLen*sin(Phi); LineTo (X + startX, Y +
startY); X = X + startX; Y = Y + startY; Left = X + Rad * sin(Phi) - Rad; Top = Y - Rad *
cos(Phi) - Rad; Right = Left + 2 * Rad; Bottom = Top + 2 * Rad; SetRect(&Circle, Left, Top,
Right, Bottom); FrameOval(&Circle); X = Left + Rad; Y = Top + 2 * Rad; MoveTo(X, Y);
LineTo(ForcesSimPaneWidth, Y); MoveTo(X, Y); X0 = X;
for (i = 0; i < 40; i++) {
Move(-4, 0); LineTo(X0 - 8, Y); X0 -= 8; }
X0 = X - Rad*tan(Phi/2)*cos(Phi) - Rad*tan(Phi/2); Y0 = Y - Rad*tan(Phi/2)*sin(Phi);
MoveTo(X0, Y0); LineTo(X0 - 5, Y); MoveTo(X0 - 20, (Y0 + Y)/2); DrawString("pØ");
ForcesTrackDrawn = 1; }

void CForcesPane::FrameForcesBall(void)
{ Rect forceBall;
Right = Left + ballSize; Bottom = Top + ballSize; SetRect(&forceBall, Left, Top, Right, Bottom);
FrameOval(&forceBall); }

void CForcesPane::DrawingInclineBall(short startX, short startY, short TrackWidth, int height)
{ int aHor = 0, aVert = 0;
heightNum = height; Angle = height; X = height/tan(Phi) + startX + TrackWidth +
ballSize*sin(Phi)/2; Left = X - ballSize/2 + 1; Y = height + startY - ballSize*cos(Phi)/2;
Top = Y - ballSize/2; FrameForcesBall(); aHor = X - ballSize * sin(Phi)/2; aVert = Y + ballSize *
cos(Phi)/2; MoveTo(aHor, aVert); LineTo(aHor + 20, aVert - 20); LineTo(aHor + 15, aVert - 20);
MoveTo(aHor + 20, aVert - 20); LineTo(aHor + 20, aVert - 15); MoveTo(aHor + 25, aVert - 20);
DrawString("pN"); MoveTo(X, Y);
if (TotForces) {
LineTo(X, Y + 40); LineTo(X - 4, Y + 36); MoveTo(X, Y + 40); LineTo(X + 4, Y + 36);
MoveTo(X - 8, Y + 50); DrawString("pMg"); }
if (ResForces) {
LineTo(X - 15, Y + 15); LineTo(X - 15, Y + 10); MoveTo(X - 15, Y + 15); LineTo(X - 10, Y +
15); MoveTo(X - 55, Y + 28); DrawString("pMgcosØ"); MoveTo(X, Y); LineTo(X + 15, Y + 15);
LineTo(X + 15, Y + 10); MoveTo(X + 15, Y + 15); LineTo(X + 10, Y + 15); MoveTo(X + 18, Y
+ 16); DrawString("pMgsinØ"); }
if (drawRollSlid) {
MoveTo(X - ballSize*sin(Pi/4)/2 + 1, Y + ballSize*cos(Pi/4)/2 - 1); LineTo(X -
ballSize*sin(Pi/4)/2 - 14, Y + ballSize*cos(Pi/4)/2 - 16); LineTo(X - ballSize*sin(Pi/4)/2 - 9, Y +
ballSize*cos(Pi/4)/2 - 16); MoveTo(X - ballSize*sin(Pi/4)/2 - 14, Y + ballSize*cos(Pi/4)/2 - 16);
LineTo(X - ballSize*sin(Pi/4)/2 - 14, Y + ballSize*cos(Pi/4)/2 - 11); MoveTo(X -
ballSize*sin(Pi/4)/2 - 8, Y + ballSize*cos(Pi/4)/2 - 15); DrawString("pf"); }
InclineBall = 1; }

void CForcesPane::DrawingLoopBall(short startX, short startY, short TrackWidth, double Beta)
{ CForcesPaneSec *theForcesDraw;
slopeLen = (verHt - Rad*(1 - cos(Phi)))/sin(Phi); Angle = Beta; Beta = Beta * Pi/ 180;
if (!Beta) X0 = startX + TrackWidth + ballSize*sin(Phi)/2 - 2;
else X0 = startX + TrackWidth + ballSize*sin(Phi)/2;
X = X0 + (Rad - ballSize/2 - 1)*(sin(Beta) - sin(-Phi)) + slopeLen*cos(Phi); Y0 = startY -
ballSize*cos(Phi)/2; Y = Y0 + (Rad - ballSize/2 - 1)*(cos(Beta) - cos(-Phi)) + slopeLen*sin(Phi);
Left = X - ballSize/2; Top = Y - ballSize/2; FrameForcesBall();
switch (Angle) {
case 0: theForcesDraw = new (CForcesPaneSec); theForcesDraw->ForcesDrawBot(); break;
case 180: theForcesDraw = new (CForcesPaneSec); theForcesDraw->ForcesDrawTop(); break;
case 45: theForcesDraw = new (CForcesPaneSec); theForcesDraw->ForcesDrawLRQ(); break;
case 135: theForcesDraw = new (CForcesPaneSec); theForcesDraw->ForcesDrawURQ(); break;
case 225: theForcesDraw = new (CForcesPaneSec); theForcesDraw->ForcesDrawULQ(); break;
case 315: theForcesDraw = new (CForcesPaneSec); theForcesDraw->ForcesDrawLLQ(); break; }
LoopBall = 1;}

void CForcesPane::DrawingHorBall(short startX, short startY, short TrackWidth)
{ Angle = 360; Left = startX + slopeLen*cos(Phi) + 3*Rad; Top = startY + Rad - Rad * cos(Phi)
+ slopeLen*sin(Phi) - ballSize - TrackWidth; Right = Left + ballSize; Bottom = Top + ballSize;
FrameForcesBall(); X = Left + ballSize/2; Y = Top + ballSize/2; MoveTo(X, Y); LineTo(X, Y -
28);
LineTo(X - 4, Y - 24); MoveTo(X, Y - 28); LineTo(X + 4, Y - 24); MoveTo(X - 4, Y - 30);
DrawString("pN"); MoveTo(X, Y); LineTo(X, Y + 28); LineTo(X - 4, Y + 24); MoveTo(X, Y +
28); LineTo(X + 4, Y + 24); MoveTo(X + 6, Y + 28); DrawString("pMg"); HorizBall = 1; }

```

```
void    CFForcesPane::DrawMgLine(void)
{ DrawString("\pN"); MoveTo(X, Y); LineTo(X, Y + 39); LineTo(X - 4, Y + 35); MoveTo(X, Y + 39); LineTo(X + 4, Y + 35); MoveTo(X + 7, Y + 39); DrawString("\pMg"); }
```

CFForcesPaneSec.c

(Methods for the force drawing pane.)

#include the following: "CFForcesPaneSec.h", <Commands.h>, <Constants.h>, "CLoopDoc.h", "AppCommands.h", "Global.h", <SANE.h>, <ctype.h>, <math.h>, <errno.h>, "CFForcesPane.h"

extern double Rad, Pi, X, Y, Phi; extern int ballSize, TotForces, ResForces, drawRollSlid; CFForcesPane *theMgLine;

```
void    CFForcesPaneSec::ForcesDrawBot(void)
{ MoveTo(X, Y); LineTo(X, Y - 28); LineTo(X - 4, Y - 24); MoveTo(X, Y - 28); LineTo(X + 4, Y - 24); MoveTo(X - 4, Y - 30); DrawString("\pN"); MoveTo(X, Y); LineTo(X, Y + 26); LineTo(X - 4, Y + 22); MoveTo(X, Y + 26); LineTo(X + 4, Y + 22); MoveTo(X + 6, Y + 26); DrawString("\pMg"); }
```

```
void    CFForcesPaneSec::ForcesDrawTop(void)
{ MoveTo(X - 2, Y - ballSize/2); LineTo(X - 2, Y - ballSize/2 + 28); LineTo(X - 6, Y - ballSize/2 + 24); MoveTo(X - 2, Y - ballSize/2 + 28); LineTo(X + 2, Y - ballSize/2 + 24); MoveTo(X - 14, Y - ballSize/2 + 24); DrawString("\pN"); MoveTo(X, Y); LineTo(X, Y + 32); LineTo(X - 4, Y + 28); MoveTo(X, Y + 32); LineTo(X + 4, Y + 28); MoveTo(X, Y + 45); DrawString("\pMg"); }
```

```
void    CFForcesPaneSec::ForcesDrawLRQ(void)
{ MoveTo(X, Y); LineTo(X - 20, Y - 20); LineTo(X - 15, Y - 20); MoveTo(X - 20, Y - 20); LineTo(X - 20, Y - 15); MoveTo(X, Y); LineTo(X + ballSize*sin(Phi)/2, Y + ballSize*sin(Phi)/2); MoveTo(X - 16, Y - 22);
if (TotForces) {
theMgLine = new (CFForcesPane); theMgLine->DrawMgLine(); }
if (ResForces) {
DrawString("\pN"); MoveTo(X, Y); LineTo(X + 18, Y + 18); LineTo(X + 18, Y + 13); MoveTo(X + 18, Y + 18); LineTo(X + 13, Y + 18); MoveTo(X + 22, Y + 14); DrawString("\pMgcosβ");
MoveTo(X, Y); LineTo(X - 25, Y + 25); LineTo(X - 25, Y + 20); MoveTo(X - 25, Y + 25); LineTo(X - 20, Y + 25); MoveTo(X - 73, Y + 37); DrawString("\pMgsinβ"); MoveTo(X - 27, Y - 2); DrawString("\pβ"); DrawBetaAngle(45); }
if (drawRollSlid) {
MoveTo(X + ballSize*sin(Phi)/2, Y + ballSize*sin(Phi)/2); LineTo(X + ballSize*sin(Phi)/2 - 18, Y + ballSize*sin(Phi)/2 + 18); LineTo(X + ballSize*sin(Phi)/2 - 18, Y + ballSize*sin(Phi)/2 + 13);
MoveTo(X + ballSize*sin(Phi)/2 - 18, Y + ballSize*sin(Phi)/2 + 18); LineTo(X + ballSize*sin(Phi)/2 - 13, Y + ballSize*sin(Phi)/2 + 18); MoveTo(X + ballSize*sin(Phi)/2 - 19, Y + ballSize*sin(Phi)/2 + 30); DrawString("\pφ"); }
}
```

```
void    CFForcesPaneSec::ForcesDrawURQ(void)
{ MoveTo(X, Y); LineTo(X - 20, Y + 20); LineTo(X - 15, Y + 20); MoveTo(X - 20, Y + 20); LineTo(X - 20, Y + 15); MoveTo(X, Y); LineTo(X + ballSize*sin(Phi)/2, Y - ballSize*sin(Phi)/2); MoveTo(X - 14, Y + 26);
if (TotForces) {
theMgLine = new (CFForcesPane); theMgLine->DrawMgLine(); }
if (ResForces) {
MoveTo(X - 22, Y + 11); DrawString("\pN"); MoveTo(X + ballSize*sin(Phi)/2 - 1, Y - ballSize*sin(Phi)/2 - 1); LineTo(X + ballSize*sin(Phi)/2 - 21, Y - ballSize*sin(Phi)/2 + 19);
LineTo(X + ballSize*sin(Phi)/2 - 21, Y - ballSize*sin(Phi)/2 + 16); MoveTo(X + ballSize*sin(Phi)/2 - 21, Y - ballSize*sin(Phi)/2 + 19); LineTo(X + ballSize*sin(Phi)/2 - 16, Y - ballSize*sin(Phi)/2 + 19); MoveTo(X - 120, Y + 25); DrawString("\pMgcos(180 - β)");
MoveTo(X, Y); LineTo(X + 22, Y + 22); LineTo(X + 22, Y + 17); MoveTo(X + 22, Y + 22); LineTo(X + 17, Y + 22); MoveTo(X + 31, Y + 29); DrawString("\pMgsin(180 - β)");
MoveTo(X - 14, Y + 44); DrawString("\pβ"); DrawBetaAngle(135); }
if (drawRollSlid) {
MoveTo(X + ballSize*sin(Phi)/2, Y - ballSize*sin(Phi)/2); LineTo(X + ballSize*sin(Phi)/2 + 18, Y - ballSize*sin(Phi)/2 + 18); LineTo(X + ballSize*sin(Phi)/2 + 18, Y - ballSize*sin(Phi)/2 + 13);
MoveTo(X + ballSize*sin(Phi)/2 + 18, Y - ballSize*sin(Phi)/2 + 18); LineTo(X + ballSize*sin(Phi)/2 + 13, Y - ballSize*sin(Phi)/2 + 18); MoveTo(X + ballSize*sin(Phi)/2 + 20, Y - ballSize*sin(Phi)/2 + 20); DrawString("\pφ"); }
}
```

```

}

void CFforcesPaneSec::ForcesDrawULQ(void)
{ MoveTo(X, Y); LineTo(X + 20, Y + 20); LineTo(X + 15, Y + 20); MoveTo(X + 20, Y + 20);
  LineTo(X + 20, Y + 15); MoveTo(X, Y); LineTo(X - ballSize*sin(Phi)/2, Y - ballSize*sin(Phi)/2);
  MoveTo(X + 15, Y + 32);
  if (TotForces) {
    DrawString("\pN"); MoveTo(X, Y); LineTo(X, Y + 39); LineTo(X - 4, Y + 35); MoveTo(X, Y +
    39); LineTo(X + 4, Y + 35); MoveTo(X, Y + 50); DrawString("\pMg"); }
  if (ResForces) {
    MoveTo(X - ballSize*sin(Phi)/2 + 20, Y - ballSize*sin(Phi)/2 + 12); DrawString("\pN");
    MoveTo(X - ballSize*sin(Phi)/2 + 2, Y - ballSize*sin(Phi)/2 - 3); LineTo(X - ballSize*sin(Phi)/2
    + 22, Y - ballSize*sin(Phi)/2 + 17); LineTo(X - ballSize*sin(Phi)/2 + 22, Y - ballSize*sin(Phi)/2
    + 12); MoveTo(X - ballSize*sin(Phi)/2 + 22, Y - ballSize*sin(Phi)/2 + 17); LineTo(X -
    ballSize*sin(Phi)/2 + 17, Y - ballSize*sin(Phi)/2 + 17); MoveTo(X + 26, Y + 20);
    DrawString("\pMgcos(β - 180)"); MoveTo(X, Y); LineTo(X - 23, Y + 23); LineTo(X - 23, Y +
    18);
    MoveTo(X - 23, Y + 23); LineTo(X - 18, Y + 23); MoveTo(X - 120, Y + 35);
    DrawString("\pMgsin(β - 180)"); MoveTo(X + 42, Y + 40); DrawString("\pβ");
    DrawBetaAngle(225); }
  if (drawRollSlid) {
    MoveTo(X - ballSize*sin(Phi)/2, Y - ballSize*sin(Phi)/2); LineTo(X - ballSize*sin(Phi)/2 + 18, Y -
    ballSize*sin(Phi)/2 - 18); LineTo(X - ballSize*sin(Phi)/2 + 18, Y - ballSize*sin(Phi)/2 - 13);
    MoveTo(X - ballSize*sin(Phi)/2 + 18, Y - ballSize*sin(Phi)/2 - 18); LineTo(X -
    ballSize*sin(Phi)/2 + 13, Y - ballSize*sin(Phi)/2 - 18); MoveTo(X - ballSize*sin(Phi)/2 + 18, Y -
    ballSize*sin(Phi)/2 - 20); DrawString("\pf"); }
}

void CFforcesPaneSec::ForcesDrawLLQ(void)
{ MoveTo(X, Y); LineTo(X + 20, Y - 20); LineTo(X + 15, Y - 20); MoveTo(X + 20, Y - 20);
  LineTo(X + 20, Y - 15); MoveTo(X, Y); LineTo(X - ballSize*sin(Phi)/2, Y + ballSize*sin(Phi)/2);
  MoveTo(X + 7, Y - 22);
  if (TotForces) {
    theMgLine = new (CFforcesPane); theMgLine->DrawMgLine(); }
  if (ResForces) {
    DrawString("\pN"); MoveTo(X, Y); LineTo(X + 26, Y + 26); LineTo(X + 26, Y + 21); MoveTo(X
    + 26, Y + 26); LineTo(X + 21, Y + 26); MoveTo(X + 32, Y + 36); DrawString("\pMgsin(360 -
    β)"); MoveTo(X, Y); LineTo(X - 26, Y + 26); LineTo(X - 26, Y + 21); MoveTo(X - 26, Y + 26);
    LineTo(X - 21, Y + 26); MoveTo(X - 120, Y + 41); DrawString("\pMgcos(360 - β)"); MoveTo(X
    + 44, Y - 30); DrawString("\pβ"); DrawBetaAngle(315); }
  if (drawRollSlid) {
    MoveTo(X - ballSize*sin(Phi)/2, Y + ballSize*sin(Phi)/2); LineTo(X - ballSize*sin(Phi)/2 - 18, Y
    + ballSize*sin(Phi)/2 - 18); LineTo(X - ballSize*sin(Phi)/2 - 18, Y + ballSize*sin(Phi)/2 - 13);
    MoveTo(X - ballSize*sin(Phi)/2 - 18, Y + ballSize*sin(Phi)/2 - 18); LineTo(X -
    ballSize*sin(Phi)/2 - 13, Y + ballSize*sin(Phi)/2 - 18); MoveTo(X - ballSize*sin(Phi)/2 - 27, Y +
    ballSize*sin(Phi)/2 - 8); DrawString("\pf"); }
}

void CFforcesPaneSec::DrawBetaAngle(double Teta)
{ double XC = 0, YC = 0, ZX = 0, ZY = 0, i = 0;

  Teta = Teta * Pi/180; XC = X - (Rad - ballSize/2)*sin(Teta); YC = Y - (Rad -
  ballSize/2)*cos(Teta); MoveTo(XC, YC); ZX = XC; ZY = YC;
  for (i = 0; i <= 8; i++) {
    LineTo(ZX, ZY + 3); ZY += 6; MoveTo(ZX, ZY); }
  MoveTo(XC, YC + 10);
  for (i = 0.2; i <= Teta; i += 0.2) {
    ZX = XC + 10*sin(i); ZY = YC + 10*cos(i); LineTo(ZX, ZY); }
  MoveTo(XC, YC);
  for (i = 0; i < 4; i++) {
    Line(4*sin(Teta), 4*cos(Teta)); Move(4*sin(Teta), 4*cos(Teta)); }
}

```

CFforcesSimulate.c

(Methods for the sforce simulations)

#include the following: "Commands.h", "CFforcesSimulate.h", "Global.h", "AppCommands.h",
 <CBartender.h>, <CDialog.h>, "CLoopDoc.h", "CLoopApp.h", <CDesktop.h>, <TBUilities.h>,

```
"CForcesEqCheck.h", "CForcesPane.h", <math.h>, <errno.h>, "CEquations.h", "CEnergyCheck.h",
<CRadioGroupPane.h>
```

```
CForcesSimulate *theDrawResolve; CForcesPane *theTrackForces; extern CEnergyCheck
*theEnergyCheck; extern double X, Y; int posForces, drawRollSlid; extern int ForcesTrackDrawn,
InclineBall, LoopBall, HorizBall, Acceleration, Angle, ballSize, TotForces, ResForces;
```

```
enum { kRadioDrawSlidRoll = 10, kDrawSliding, kDrawRolling };
```

```
void CForcesSimulate::IForcesSimulate(short DLOGid, CDirectorOwner *aSupervisor)
{ CDLOGDirector::IDLOGDirector( DLOGid, aSupervisor); }
```

```
void CForcesSimulate::BegDialDBut(long theCommand, long theDefaultCmd)
{ CRadioGroupPane *radioGroupSet; CDialog *theDialog = (CDialog*) itsWindow;
```

```
theDialog->Select(); theDialog->SetDefaultCmd(theDefaultCmd); theDialog->
SetCmdEnable(cmdAcceleration, FALSE); theDialog->SetCmdEnable(cmdDrawForces, FALSE);
theDialog->SetCmdEnable(cmdResolveForce, FALSE); theDialog->SetCmdEnable(cmdEraseAll,
FALSE); inherited::BeginDialog(); radioGroupSet = (CRadioGroupPane *)itsWindow
->FindViewByID(kRadioDrawSlidRoll); radioGroupSet -> SetStationID(kDrawSliding);
drawRollSlid = 0; }
```

```
void CForcesSimulate::ThrowAwayDial(long closingButton)
{ CDialog *theDialogStart = (CDialog*) itsWindow;
```

```
inherited::EndDialog(closingButton, FALSE); theDialogStart->Select(); theDialogStart->Close();}
```

```
void CForcesSimulate::Activate(void)
{ CLoopDoc *theWindowChanging = NULL;
```

```
inherited::Activate(); theWindowChanging = new (CLoopDoc); theWindowChanging->
ChangingMenus(); }
```

```
void CForcesSimulate::DrawFrictorNot(void)
{ CRadioGroupPane *radioGroupSet;
```

```
radioGroupSet = (CRadioGroupPane *)itsWindow ->FindViewByID(kRadioDrawSlidRoll);
switch (radioGroupSet -> GetStationID()) {
case kDrawSliding: drawRollSlid = 0; break;
case kDrawRolling: drawRollSlid = 1; break;
default: break; }
}
```

```
void CForcesSimulate::DoCommand(long theCommand)
{ CDialog *theDialog = (CDialog*) itsWindow;
switch (theCommand) {
case cmdQUITForceDiag: theDrawResolve->ThrowAwayDial(cmdQUITForceDiag); theDrawResolve
= NULL; ForgetObject(theDrawResolve); break;
case cmdSimStartForces: posForces = 1; break;
case cmdSimInclineForces: posForces = 2; break;
ase cmdSimBotForces: posForces = 3; break;
case cmdSimLRQForces: posForces = 4; break;
case cmdSimTRQForces: posForces = 5; break;
case cmdSimTopForces: posForces = 6; break;
case cmdSimTLQForces: posForces = 7; break;
case cmdSimBLQForces: posForces = 8; break;
case cmdSimHorForces: posForces = 9; break;
case cmdAcceleration: theDialog->SetCmdEnable(cmdAcceleration, FALSE); theTrackForces->
Prepare(); DrawAcceleration(); break;
case cmdDrawTrackForce: theDialog->SetCmdEnable(cmdDrawTrackForce, FALSE);
DrawForcesTrack(); theDialog->SetCmdEnable(cmdDrawForces, TRUE); theDialog->
SetCmdEnable(cmdResolveForce, TRUE); theDialog->SetCmdEnable(cmdResolveForce, TRUE);
theDialog->SetCmdEnable(cmdEraseAll, TRUE); break;
case cmdResolveForce: theDialog->SetCmdEnable(cmdResolveForce, FALSE); theDialog->
SetCmdEnable(cmdDrawForces, FALSE); Acceleration = 0; InclineBall = 0; LoopBall = 0;
HorizBall = 0; ResForces = 1; TotForces = 0; DrawFrictorNot(); ChooseForcesPosition();
theDialog
->SetCmdEnable(cmdAcceleration, TRUE); break;
```

```

case cmdEraseAll: theDialog->SetCmdEnable(cmdDrawTrackForce, FALSE); theDialog->
SetCmdEnable(cmdAcceleration, FALSE); InclineBall = 0; LoopBall = 0; HorizBall = 0;
Acceleration = 0; InclineBall = 0; theDialog->SetCmdEnable(cmdDrawForces, TRUE); theDialog->
SetCmdEnable(cmdResolveForce, TRUE); theDialog->SetCmdEnable(cmdAcceleration, FALSE);
theTrackForces->Refresh(); break;
case cmdDrawForces: theDialog->SetCmdEnable(cmdDrawForces, FALSE); theDialog->
SetCmdEnable(cmdResolveForce, FALSE); Acceleration = 0; TotForces = 1; ResForces = 0;
LoopBall = 0; HorizBall = 0; DrawFrictorNot(); ChooseForcesPosition(); theDialog->
SetCmdEnable(cmdAcceleration, TRUE); break;
default: inherited::DoCommand(theCommand); }
}

```

```

void CFforcesSimulate::DrawForcesTrack(void)
{ CDialog *theDialogWindow = (CDialog*) itsWindow; CEquations *theEquation;

```

```

theTrackForces = new (CFforcesPane); theTrackForces->IFforcesPane(ForcesPane, itsWindow, this);
theTrackForces->Prepare(); theTrackForces->DrawForcesTrack(ForcesPane, 40, 20); }

```

```

void CFforcesSimulate::ChooseForcesPosition(void)
{ switch (posForces) {
case 1: theTrackForces->Prepare(); theTrackForces->DrawingInclineBall(40, 20, 1, 10); break;
case 2: theTrackForces->Prepare(); theTrackForces->DrawingInclineBall(40, 20, 1, 70); break;
case 3: theTrackForces->Prepare(); theTrackForces->DrawingLoopBall(40, 20, 1, 0); break;
case 4: theTrackForces->Prepare(); theTrackForces->DrawingLoopBall(40, 20, 1, 45); break;
case 5: theTrackForces->Prepare(); theTrackForces->DrawingLoopBall(40, 20, 1, 135); break;
case 6: theTrackForces->Prepare(); theTrackForces->DrawingLoopBall(40, 20, 1, 180); break;
case 7: theTrackForces->Prepare(); theTrackForces->DrawingLoopBall(40, 20, 1, 225); break;
case 8: theTrackForces->Prepare(); theTrackForces->DrawingLoopBall(40, 20, 1, 315); break;
case 9: theTrackForces->Prepare(); theTrackForces->DrawingHorBall(40, 20, 1); break; }
}

```

```

void CFforcesSimulate::ForceSimDial(Str255 theText0, Str255 theText1, Str255 theText2,
Str255 theText3, short IdNum)
{ ParamText(theText0, theText1, theText2, theText3); PositionDialog("ALRT", IdNum); InitCursor;
NoteAlert(IdNum, NULL); return; }

```

```

void CFforcesSimulate::DrawAcceleration(void)
{ Acceleration = 1;
switch (Angle) {
case 0: theTrackForces->Prepare(); MoveTo(X + ballSize/2 + 4, Y - ballSize/4); LineTo(X +
ballSize/2 + 4, Y - ballSize/4 - 16); LineTo(X + ballSize/2, Y - ballSize/4 - 12); MoveTo(X +
ballSize/2 + 4, Y - ballSize/4 - 16); LineTo(X + ballSize/2 + 8, Y - ballSize/4 - 12); MoveTo(X +
ballSize/2 + 4, Y - ballSize/4 - 12); LineTo(X + ballSize/2, Y - ballSize/4 - 8); MoveTo(X +
ballSize/2 + 4, Y - ballSize/4 - 12); LineTo(X + ballSize/2 + 8, Y - ballSize/4 - 8); MoveTo(X +
ballSize/2 + 11, Y - ballSize/4 - 6); DrawString("\pa"); break;
case 10: DrawInclAccel(); break;
case 45: theTrackForces->Prepare(); MoveTo(X + ballSize, Y - ballSize/2); LineTo(X + ballSize -
16, Y - ballSize/2 - 16); LineTo(X + ballSize - 11, Y - ballSize/2 - 16); MoveTo(X + ballSize - 16,
Y - ballSize/2 - 16); LineTo(X + ballSize - 16, Y - ballSize/2 - 11); MoveTo(X + ballSize - 11, Y -
ballSize/2 - 11); LineTo(X + ballSize - 6, Y - ballSize/2 - 11); MoveTo(X + ballSize - 11, Y -
ballSize/2 - 11); LineTo(X + ballSize - 11, Y - ballSize/2 - 6); MoveTo(X + ballSize - 4, Y -
ballSize/2 - 6); DrawString("\pa"); MoveTo(X - ballSize, Y); LineTo(X - ballSize - 16, Y + 16);
LineTo(X - ballSize - 12, Y + 16); MoveTo(X - ballSize - 16, Y + 16); LineTo(X - ballSize - 16, Y
+ 12); MoveTo(X - ballSize - 12, Y + 12); LineTo(X - ballSize - 12, Y + 8); MoveTo(X - ballSize
- 12, Y + 12); LineTo(X - ballSize - 8, Y + 12); MoveTo(X - ballSize - 33, Y + 10);
DrawString("\pa"); Move(1, 4); DrawString("\pt"); break;
case 70: DrawInclAccel(); break;
case 135: theTrackForces->Prepare(); MoveTo(X - 4, Y - ballSize - 5); LineTo(X - 20, Y - ballSize
+ 11); LineTo(X - 15, Y - ballSize + 11); MoveTo(X - 20, Y - ballSize + 11); LineTo(X - 20, Y -
ballSize + 6); MoveTo(X - 15, Y - ballSize + 6); LineTo(X - 10, Y - ballSize + 6);
MoveTo(X - 15, Y - ballSize + 6); LineTo(X - 15, Y - ballSize + 1); MoveTo(X - 25, Y - ballSize
+ 5); DrawString("\pa"); MoveTo(X + ballSize, Y - 15); LineTo(X + ballSize + 16, Y + 1);
LineTo(X + ballSize + 16, Y - 3); MoveTo(X + ballSize + 16, Y + 1); LineTo(X + ballSize + 12,
Y + 1); MoveTo(X + ballSize + 12, Y - 3); LineTo(X + ballSize + 12, Y - 7); MoveTo(X +
ballSize + 12, Y - 3); LineTo(X + ballSize + 8, Y - 3); MoveTo(X + ballSize + 15, Y - 5);
DrawString("\pa");
Move(1,5); DrawString("\pt"); break;
case 180: theTrackForces->Prepare(); MoveTo(X + ballSize, Y + ballSize/2); LineTo(X + ballSize,
Y + ballSize/2 + 16); LineTo(X + ballSize - 4, Y + ballSize/2 + 12); MoveTo(X + ballSize, Y +

```



```

ballSize/2 + 16); LineTo(X + ballSize + 4, Y + ballSize/2 + 12); MoveTo(X + ballSize, Y +
ballSize/2 + 12); LineTo(X + ballSize - 4, Y + ballSize/2 + 8); MoveTo(X + ballSize, Y +
ballSize/2 + 12); LineTo(X + ballSize + 4, Y + ballSize/2 + 8); MoveTo(X + ballSize + 6, Y +
ballSize/2 + 10); DrawString("\pa"); break;
case 225: theTrackForces->Prepare(); MoveTo(X + 25, Y - 2*ballSize + 20); LineTo(X + 37, Y -
2*ballSize + 32); LineTo(X + 37, Y - 2*ballSize + 28); MoveTo(X + 37, Y - 2*ballSize + 32);
LineTo(X + 33, Y - 2*ballSize + 32); MoveTo(X + 33, Y - 2*ballSize + 28); LineTo(X + 33, Y -
2*ballSize + 24); MoveTo(X + 33, Y - 2*ballSize + 28); LineTo(X + 29, Y - 2*ballSize + 28);
MoveTo(X + 32, Y - 2*ballSize + 24); DrawString("\pa"); MoveTo(X - 2*ballSize + 5, Y + 10);
LineTo(X - 2*ballSize - 7, Y + 22); LineTo(X - 2*ballSize - 7, Y + 18); MoveTo(X - 2*ballSize -
7, Y + 22); LineTo(X - 2*ballSize - 3, Y + 22); MoveTo(X - 2*ballSize - 3, Y + 18); LineTo(X -
2*ballSize - 3, Y + 14); MoveTo(X - 2*ballSize - 3, Y + 18); LineTo(X - 2*ballSize + 1, Y + 18);
MoveTo(X - 2*ballSize - 15, Y + 6); DrawString("\pa"); Move(1, 5); DrawString("\pt"); break;
case 315: theTrackForces->Prepare(); MoveTo(X - ballSize - 4, Y - 4); LineTo(X - ballSize + 12, Y -
20); LineTo(X - ballSize + 7, Y - 20); MoveTo(X - ballSize + 12, Y - 20); LineTo(X - ballSize +
12, Y - 15); MoveTo(X - ballSize + 7, Y - 15); LineTo(X - ballSize + 2, Y - 15); MoveTo(X -
ballSize + 7, Y - 15); LineTo(X - ballSize + 7, Y - 10); MoveTo(X - ballSize - 2, Y - 18);
DrawString("\pa"); MoveTo(X + ballSize - 2, Y); LineTo(X + ballSize + 14, Y + 16); LineTo(X +
ballSize + 14, Y + 12); MoveTo(X + ballSize + 14, Y + 16); LineTo(X + ballSize + 10, Y + 16);
MoveTo(X + ballSize + 10, Y + 12); LineTo(X + ballSize + 10, Y + 8); MoveTo(X + ballSize +
10, Y + 12); LineTo(X + ballSize + 6, Y + 12); MoveTo(X + ballSize + 12, Y + 10);
DrawString("\pa"); Move(1, 5); DrawString("\pt"); break;
case 360: theTrackForces->Prepare(); MoveTo(X - 70, Y - 50); DrawString("\pThere is no
acceleration."); break; }
if (Angle == 45 || Angle == 135 || Angle == 225 || Angle == 315) {
theTrackForces->Prepare(); MoveTo(200, 15); DrawString("\pa is the radial acceleration.");
MoveTo(200, 35); DrawString("\pa"); Move(1, 5); DrawString("\pt"); Move(5, -5);
DrawString("\pis the tangential acceleration."); }
}

void CFocesSimulate::DrawInclAccel(void)
{ theTrackForces->Prepare(); MoveTo(X - ballSize - 23, Y - 3); LineTo(X - ballSize - 10, Y + 10);
LineTo(X - ballSize - 14, Y + 10); MoveTo(X - ballSize - 10, Y + 10); LineTo(X - ballSize - 10, Y
+ 6); MoveTo(X - ballSize - 14, Y + 6); LineTo(X - ballSize - 18, Y + 6); MoveTo(X - ballSize -
14, Y + 6); LineTo(X - ballSize - 14, Y + 2); MoveTo(X - ballSize - 27, Y + 15);
DrawString("\pa"); }

```

CSimPane.c

(The methods for the simulation pane depicting the motion of the ball)

```

#include the following: "CSimPane.h", <Commands.h>, <Constants.h>, "CLoopDoc.h",
"AppCommands.h", "Global.h", <SANE.h>, <ctype.h>, <math.h>, <errno.h>, <CDialog.h>,
"CStartDirPanel.h"

```

```

extern CSimPane *theSimPane, *theScalePane, *theLargeScalePane, *theCharPane;
extern double Pi, ht, R, g, Alpha, H; extern int Track, SIZFACTOR, PaneWidth, side, Sliding,
Rolling; extern Str255 scaleNum; extern int SIZFACTORExt, picture, Oscillating;
double X, Y, X0, Y0, T, t, ti, Teta, xPos[350], yPos[350], l;
double Omega, angDisp[350], noPos; extern PicHandle theSimPic;
extern Rect picRect;

```

```

void CSimPane::ISimPane(short resID, CView *anEnclosure, CBureaucrat *aSupervisor)
{ CPane::IViewRes('Pane', resID, anEnclosure, aSupervisor); }

```

```

void CSimPane::Draw(Rect *area)
{ CursHandle gWatchCursor;

```

```

gWatchCursor = GetCursor(watchCursor); SetCursor(*gWatchCursor);
if (Track & !picture) {
theSimPane->Prepare(); DrawTrack( SimulationPane, 2, 15, 1, 1); }
if (picture) {
theSimPane->Prepare(); OffsetRect(&picRect, -141, -29); DrawPicture(theSimPic, &picRect);
OffsetRect(&picRect, 141, 29);
if (PaneWidth == DrawingPaneWidth) WriteScaleValue(ScalePane);
else WriteScaleValue(LargeScalePane); }
inherited::Draw(area); InitCursor(); }

```

```

void    CSimPane::WriteScaleValue(short PanelId)
{
    int scalePaneHeight = 20, scalePaneWidth = 96, i;
    Str255 UnitsFrom = "\p1cm : ", UnitsTo = "\pmm";

if (PanelId == ScalePane)
theScalePane->Prepare();
else
theLargeScalePane->Prepare(); MoveTo (scalePaneWidth/20, 3*scalePaneHeight/4);
DrawString(UnitsFrom); DrawString(scaleNum); DrawString(UnitsTo);
}

void    CSimPane::FramingSlope(short startX, short startY, short TrackWidth, short TrackHeight,
                                double deltaT)
{
    int i = 0, w = 0, z = 0;
    double dispAng = 0, l = 0;

if (PaneWidth == LargeDrawPaneWidth) {
side = side * SIZFACTORExt / SIZFACTOR;
if (side < 10) side = 10; }
else {
if (ht/R <= 4) side = 14;
else side = 10; }
for (i = 0; i <= 100; i++) {
xPos[i] = 0; yPos[i] = 0; angDisp[i] = 0; }
l = (ht - R*(1 - cos(Alpha)))/sin(Alpha);
if (Sliding)T = sqrt(2*l/(g*sin(Alpha)));
else
T = sqrt(2*l*7/(5*g*sin(Alpha))); X0 = (H - ht)/tan(Alpha) + startX + TrackWidth +
side*sin(Alpha)/2; Y0 = (H - ht) + startY - side*cos(Alpha)/2 ; i = 0; xPos[i] = X0 - side/2 + 1;
yPos[i] = Y0 - side/2;
if (Rolling)
angDisp[i] = 0;
for (t = 0, i = 1; t < T; t += deltaT, i++) {
if (Sliding)
X = X0 + g*t*t*sin(Alpha)*cos(Alpha)/2;
else
X = X0 + 5*g*t*t*sin(Alpha)*cos(Alpha)/14;
Y = Y0 + (X - X0)*tan(Alpha); xPos[i] = X - side/2 + 1; yPos[i] = Y - side/2;
if (Rolling) {
dispAng = 5*g*t*t*sin(Alpha)/(7*side);
while (dispAng >= 2*Pi){
dispAng -= 2*Pi;
} angDisp[i] = dispAng; }
}
noPos = i; X0 = X0 + l*cos(Alpha); Y0 = Y0 + l*sin(Alpha);
}

void    CSimPane::FramingLowerQuad(double deltaT)
{
    int i = 0;
    double dispAng = 0;
    Teta = - Alpha;
for(Teta = Teta, i = noPos; Teta <= Pi/2 ; t += deltaT, i++) {
TransitSlopeLower(Pi/2, deltaT);
if (Teta <= 0) { xPos[i] = X - side/2 + 0.5; yPos[i] = Y - side/2; }
if (Teta > 0 && Teta <= Pi/2) { xPos[i] = X - side/2 - 0.5; yPos[i] = Y - side/2; }
if (Teta > Pi/2) { xPos[i] = X - side/2 - 0.5; yPos[i] = Y - side/2 + 1; }
if (Rolling) { angDisp[i] = angDisp[i-1] + 2*R*Omega*deltaT/side ; while (angDisp[i] >=
2*Pi) { angDisp[i] -= 2*Pi; } }
}
noPos = i;}

void    CSimPane::FramingTopRightSlid(double deltaT)
{
    double Ncheck = 0, dt = 0;
    int i = 0;

for(Teta = Teta, i = noPos; Teta < Pi; i++) {
ti = 0;
for (dt = 0.1; ti <= deltaT; ti += dt) {
Ncheck = 3 * cos(Teta) + 2 * ht/R - 2; Omega = sqrt(2*g*ht/(R*R) - 2*g*(1 - cos(Teta))/R);

```

```

if (Ncheck < 0) { Teta += Omega*dt; noPos = i; return; }
Teta += Omega*dt; }
X = X0 + (R - side/2)*(sin(Teta) - sin(-Alpha)); Y = Y0 + (R - side/2)*(cos(Teta) - cos(-Alpha));
xPos[i] = X - side/2 - 0.5; yPos[i] = Y - side/2 + 0.5;
if (Teta > Pi) { xPos[i] = X - side/2 + 0.5; yPos[i] = Y - side/2 + 0.5; }
}
noPos = i; }

void CSimPane::FramingTopRightRoll(double deltaT)
{
    double Ncheck = 0, dispAng = 0, dt = 0;
    int i = 0;

    for(Teta = Teta, i = noPos; Teta < Pi; t += deltaT, i++) {
        ti = 0;
        for (dt = 0.1; ti <= deltaT; ti += dt){ Ncheck = 1.7 * cos(Teta) + ht/R - 1 ; Omega = sqrt(10*g*(ht
        - R*(1 - cos(Teta)))/(7*R*R));
        if (Ncheck < 0) { Teta += Omega*dt; noPos = i; return; }
        Teta += Omega*dt; }
        X = X0 + (R - side/2)*(sin(Teta) - sin(-Alpha)); Y = Y0 + (R - side/2)*(cos(Teta) - cos(-Alpha));
        xPos[i] = X - side/2 - 0.5; yPos[i] = Y - side/2 + 0.5;
        if (Teta > Pi) { xPos[i] = X - side/2 + 0.5; yPos[i] = Y - side/2 + 0.5; }
        angDisp[i] = angDisp[i-1] + 2*R*Omega*deltaT/side ;
        while (angDisp[i] >= 2*Pi) { angDisp[i] -= 2*Pi; }
    } noPos = i; }

void CSimPane::FramingLeftQuads(double deltaT)
{
    int i = 0;
    double dispAng = 0, dt = 0;

    for(Teta = Teta, i = noPos; Teta <= 2*Pi; t += deltaT, i++) {
        ti = 0;
        for (dt = 0.1; ti <= deltaT; ti += dt){
            if (Sliding) Omega = sqrt(2*g*ht/(R*R) - 2*g*(1 - cos(Teta))/R);
            else Omega = sqrt(10*g*(ht - R*(1 - cos(Teta)))/(7*R*R)); Teta += Omega*dt; }
        if (Teta > Pi*2) { X = X0 + (R - side/2)*(sin(Teta) - sin(-Alpha)); noPos = i; return; }
        X = X0 + (R - side/2)*(sin(Teta) - sin(-Alpha)); Y = Y0 + (R - side/2)*(cos(Teta) - cos(-Alpha));
        if (Teta > Pi && Teta <= 3*Pi/2) { xPos[i] = X - side/2 + 0.5; yPos[i] = Y - side/2; }
        if (Teta >= 3*Pi/2) { xPos[i] = X - side/2 + 0.5; yPos[i] = Y - side/2; }
        if (Rolling) { angDisp[i] = angDisp[i-1] + 2*R*Omega*deltaT/side ;
        while (angDisp[i] >= 2*Pi) { angDisp[i] -= 2*Pi; }
        }
    } noPos = i; }

void CSimPane::FramingHoriz(short startY, short TrackHeight, double deltaT)
{
    int i = 0; double dispAng = 0, vel = 0;

    if (Sliding) vel = sqrt(2*g*ht);
    else vel = sqrt(10*g*ht/7);
    for (X = X, i = noPos; X <= PaneWidth; X += vel*deltaT, i++){
        xPos[i] = X; yPos[i] = H + startY - side - TrackHeight;
        if (Rolling) { dispAng += 2*vel*deltaT/side; while (dispAng >= 2*Pi)
            { dispAng -= 2*Pi; } angDisp[i] = dispAng; }
    } noPos = i; }

void CSimPane::DrawProj(double deltaT)
{
    double vel = 0, velY = 0, velX = 0, time = 0, Omega0 = 0, dt = 0.1; int i = 0;
    if (Sliding) { X = X0 + (R - side/2)*(sin(Teta) - sin(-Alpha)); Y = Y0 + (R -
    side/2)*(cos(Teta) - cos(-Alpha));
    if (Omega > 0.015) Omega = sqrt(2*g*ht/(R*R) - 2*g*(1 - cos(Teta))/R);
    else Omega = 0.013; }
    if (Rolling) { X = X0 + (R - side/2)*(sin(Teta) - sin(-Alpha)); Y = Y0 + (R -
    side/2)*(cos(Teta) - cos(-Alpha));
    if (Omega > 0.014) Omega = sqrt(10*g*(ht - R*(1 - cos(Teta)))/(7*R*R));
    else Omega = 0.012; }
    vel = Omega * R; Omega0 = vel*2/side; velX = vel * cos(Teta); velY = vel * sin(Teta) - g * time;
    time = deltaT - ti; t += time; i = noPos;
    if (xPos[i] != 0) { X += velX * (time - 0.1); Y -= (velY * (time - 0.1) - g * (time - 0.1) *
    (time - 0.1) / 2 ); velY -= g * (time - 0.1); i++; xPos[i] = X - side/2 + 0.5; yPos[i] = Y - side/2;
    if (Rolling) { angDisp[i] = angDisp[i - 1] + Omega0*deltaT;

```

```

while (angDisp[i] >= 2*Pi) { angDisp[i] -= 2*Pi; }
}
if (Sliding)      { X += velX * (deltaT - ti + 0.1); Y -= (velY * (deltaT - ti) - g * (deltaT - ti) *
(deltaT - ti) / 2 - 2.5); velY -= g* (deltaT - ti + dt); xPos[i] = X - side/2 + 2; yPos[i] = Y - side/2 -
1 ;
                i++; }
if (Rolling)      { X += velX * (deltaT - ti - 0.1); Y -= (velY * (deltaT - ti) - g * (deltaT - ti) *
(deltaT - ti) / 2 - 1.5); velY -= g* (deltaT - ti + dt); xPos[i] = X - side/2 + 2; yPos[i] = Y - side/2 -
1;
i = noPos; xPos[i - 2] += 0.5; yPos[i - 2] += 1.5; xPos[i - 1] += 1.5; yPos[i - 1] += 2.5; X +=
0.5;
Y += 1.5; xPos[i] = X - side/2; yPos[i] = Y - side/2; angDisp[i] = angDisp[i - 1] + Omega*deltaT;
while (angDisp[i] >= 2*Pi) { angDisp[i] -= 2*Pi; } i++; }
for (time = time; Y < H + 1.5*R; time += deltaT, i++) {
if (ht <= 60) X += velX * deltaT;
else X += velX * (deltaT - 0.1); Y -= (velY * (deltaT - 0.1) - g * (deltaT) * (deltaT) / 2 - 2.5); velY
-= g* deltaT; xPos[i] = X - side/2; yPos[i] = Y - side/2 - 2;
if (Rolling)      { angDisp[i] = angDisp[i - 1] + Omega0*deltaT;
while (angDisp[i] >= 2*Pi) { angDisp[i] -= 2*Pi; }
}
} noPos = i; }

void    CSimPane::SimSlidRoll(short startX, short startY, short TrackWidth, short TrackHeight,
double deltaT, double Critical)
{ if (Sliding) Critical = 2.5 * R;
else Critical = 2.7 * R;
if ( ht >= Critical)      { FramingSlope(startX, startY, TrackWidth, TrackHeight, deltaT);
FramingLowerQuad(deltaT);
if (Sliding) FramingTopRightSlid(deltaT);
else FramingTopRightRoll(deltaT); FramingLeftQuads(deltaT); FramingHoriz(startY, TrackHeight,
deltaT); FrameSlopeBall(); }
if (ht > R && ht < Critical)      { FramingSlope(startX, startY, TrackWidth, TrackHeight, deltaT);
FramingLowerQuad(deltaT);
if (Sliding) FramingTopRightSlid(deltaT);
else FramingTopRightRoll(deltaT); DrawProj(deltaT); FrameSlopeBall(); }
}

void    CSimPane::FramingOscill(double deltaT)
{ double  Gamma = 0; int i = 0;

Gamma = acos(1 - ht/R); Teta = - Alpha;
for(Teta = Teta, i = noPos; Teta < Gamma; t += deltaT, i++) {
TransitSlopeLower(Gamma, deltaT);
if (Teta <= 0)      { xPos[i] = X - side/2 - 0.5; yPos[i] = Y - side/2; }
if (Teta > 0 && Teta <= Pi)      { xPos[i] = X - side/2 - 0.5; yPos[i] = Y - side/2 + 1; }
if (Rolling)      { angDisp[i] = angDisp[i-1] + 2*R*Omega*deltaT/side ;
while (angDisp[i] >= 2*Pi) { angDisp[i] -= 2*Pi; } }
noPos = i;
}

void    CSimPane::DrawTrack(short PaneId, short startX, short startY, int penWidth, int penHeight)
{ int Left = 0, Top = 0, Right = 0, Bottom = 0; double l = 0, X = 0, Y = 0;
Rect Circle;

l = (H - R*(1 - cos(Alpha)))/sin(Alpha); MoveTo(startX, startY); X = l*cos(Alpha); Y =
l*sin(Alpha); LineTo (X + startX , Y + startY); X = X + startX; Y = Y + startY; Left = X + R *
sin(Alpha) - R; Top = Y - R * cos(Alpha) - R; Right = Left + 2 * R + 2; Bottom = Top + 2 * R +
1; SetRect(&Circle, Left, Top, Right, Bottom); FrameOval(&Circle);
switch (PaneId) {
case SimulationPane: PaneWidth = DrawingPaneWidth; break;
case DrawingPane: PaneWidth = LargeDrawPaneWidth; break; case AboutPane: PaneWidth =
AboutPaneWidth; break; } X = Left + R; Y = Top + 2 * R; MoveTo(X, Y); LineTo(PaneWidth,
Y); Track = 1;}

void    CSimPane::TransitSlopeLower(double Angle, double deltaT)
{ double dt = 0; ti = 0;
if (Teta != -Alpha)
for (dt = 0.1; ti < deltaT + dt; ti += dt) {
if (Sliding) Omega = 2*g*ht/(R*R) - 2*g*(1 - cos(Teta))/R;

```

```

else Omega = 10*g*(ht - R*(1 - cos(Teta)))/(7*R*R);
if (Omega <= 0) return; Omega = sqrt(Omega); Teta += Omega*dt; }
else
for (dt = 0.1; ti < (t - T); ti += dt) {
if (Sliding) Omega = sqrt(2*g*ht/(R*R) - 2*g*(1 - cos(Teta))/R);
else Omega = sqrt(10*g*(ht - R*(1 - cos(Teta)))/(7*R*R)); Teta += Omega*dt; }
X = X0 + (R - side/2)*(sin(Teta) - sin(-Alpha)); Y = Y0 + (R - side/2)*(cos(Teta) - cos(-Alpha));
}

void CSimPane::FrameSlopeBall(void)
{
    unsigned long *Ticks; double L2 = 0, T2 = 0, R2 = 0, B2 = 0;
    int Left = 0, Top = 0, Right = 0, Bottom = 0, i = 0; Rect ball;

theSimPane->Prepare();
for (i = 1; i < noPos; i++) { Left = xPos[i]; Top = yPos[i]; Right = Left + side; Bottom =
Top + side; SetRect(&ball, Left, Top, Right, Bottom); FrameOval(&ball);
if (Rolling) { MoveTo(Left + side/2, Top + side/2); LineTo( Left + side/2 - (side -
2)*sin(angDisp[i]/2, Top + side/2 + (side - 2)*cos(angDisp[i])/2); }
Delay(10, &Ticks); }
}

void CSimPane::FrameOscillations(void)
{
    unsigned long *Ticks; Rect ball; int Left = 0, Top = 0, Right = 0, Bottom = 0, i =
noPos;
theSimPane->Prepare();
for (i = i - 1; i > 1; i--) { Left = xPos[i]; Top = yPos[i]; Right = Left + side; Bottom = Top +
side; SetRect(&ball, Left, Top, Right, Bottom); FrameOval(&ball);
if (Rolling) { MoveTo(Left + side/2, Top + side/2); LineTo( Left + side/2 - (side -
2)*sin(angDisp[i]/2, Top + side/2 + (side - 2)*cos(angDisp[i])/2); }
Delay(10, &Ticks); }
}

void CSimPane::FrameInLoopOsc(short startX, short startY, short TrackWidth, double deltaT)
{
    double Gamma = 0, dt = 0; int i = 0;
for (i = 0; i <= 80; i++) { xPos[i] = 0; yPos[i] = 0; angDisp[i] = 0; }
Omega = 0; Gamma = acos(1 - ht/R);
for(Teta = - Gamma, i = 1; Teta < Gamma; t += deltaT, i++) { ti = 0;
for (dt = 0.1; ti < deltaT; ti += dt) { Omega += g*dt*sin(-Teta)/R; Teta += Omega*dt; }
X0 = (H - R*(1 - cos(Alpha)))/tan(Alpha) + startX + TrackWidth + R*sin(Alpha) - R
*sin(Gamma) + side*sin(Gamma)/2 ; Y0 = startY + H - TrackWidth - ht - side*cos(Gamma)/2;
xPos[i] = X0 + (R - side/2)*(sin(Teta) - sin(-Gamma)) - side/2 + 0.5; yPos[i] = Y0 + (R -
side/2)*(cos(Teta) - cos(-Gamma)) - side/2;
if (ht <= 0.01*R){ xPos[i] = X0 + (R - side/2)*(sin(Teta) - sin(-Gamma)) - side/2 + 0.5; yPos[i] =
Y0 + (R - side/2)*(cos(Teta) - cos(-Gamma)) - side/2 + 1.5; }
if (Omega <= 0) { noPos = i; return; }
if (Rolling) { angDisp[i] = angDisp[i-1] + 2*R*Omega*deltaT/side ;while (angDisp[i] >=
2*Pi) { angDisp[i] -= 2*Pi; }
}} noPos = i; }

void CSimPane::StatOsc(short startX, short startY, short TrackWidth)
{
    Rect ball; int Left = 0, Top = 0, Right = 0, Bottom = 0;

theSimPane->Prepare(); X0 = (H - R*(1 - cos(Alpha)))/tan(Alpha) + startX + R*sin(Alpha); Y0 =
startY + H - TrackWidth; Left = X0 - side/2; Top = Y0 - side; Right = Left + side; Bottom = Top +
side; SetRect(&ball, Left, Top, Right, Bottom); FrameOval(&ball); }

```

CSolution.c

(The methods for checks and clues on the solution of the problem)

```

#include the following: "Commands.h", "CSolution.h", "Global.h", "AppCommands.h",
<CBartender.h>, <CDialog.h>, "CLoopDoc.h", "CLoopApp.h", <CDesktop.h>, <TBUtilities.h>,
"CSStartDirPanel.h", <CEditText.h>, <CRadioGroupPane.h>, "CEquations.h",

```

```
enum { kRadioSolutionID = 22, kSlidingSolution = 34, kRollingSolution };
```

```
extern CWindow **otherWindDial; extern CBartender *gBartender; extern CBureaucrat *gGopher;
```

```

extern int windowItemNum, Motion; CSolution *theSolution, *theSolutionCheck; Handle
textEntry; int StrLength, EnergyPos, Energy1, Energy2;

void CSolution::ISolution(short DLOGid, CDirectorOwner *aSupervisor)
{
    CDialogDirector::IDLOGDirector( DLOGid, aSupervisor); }

void CSolution::BegDialSolution(long theCommand)
{
    CDialog *theDialog = (CDialog*) itsWindow; CRadioGroupPane *radioGroupSet;

theDialog->Select(); inherited::BeginDialog();
if (theCommand == cmdSolutionCheck) {
    radioGroupSet = (CRadioGroupPane *)itsWindow->FindViewByID(kRadioSolutionID);
    radioGroupSet -> SetStationID(kSlidingSolution); }
}

void CSolution::ThrowSolutionDial(long closingButton)
{
    CDialog *theDialogStart = (CDialog*) itsWindow;

inherited::EndDialog(closingButton, FALSE); theDialogStart->Select(); theDialogStart->Close();
windowItemNum--; }

void CSolution::CloseDialSolution( void)
{
    Str255 theTitle; short z = 0; CWindow *theWindow;

itsWindow->GetTitle(theTitle); z = gBartender->FindItemText( MENUWindow, theTitle);
gBartender->RemoveMenuCmd(gBartender->FindCmdNumber( MENUWindow, z));
for (z = z; z < windowItemNum; z++) *(otherWindDial + z) = *(otherWindDial + z + 1);
*(otherWindDial + z) = NULL; ForgetPtr(*(otherWindDial + z));
}

void CSolution::Activate(void)
{
    CLoopDoc *theWindowChanging = NULL;

inherited::Activate(); theWindowChanging = new (CLoopDoc); theWindowChanging->
ChangingMenus(); }

void CSolution::DoCommand(long theCommand)
{
    CStartDirPanel *theStartDir; Str255 theTitle; short theModifiers; CEquations
    *theEquations;
    switch (theCommand) {
    case cmdSolutionQuit: theSolution->CloseDialSolution(); theSolution->
    ThrowSolutionDial(cmdSolutionQuit); theSolution = NULL; ForgetObject(theSolution); break;
    case ExplainNormal: SolutionDial(WhyNormal); break;
    case ExplainTop: SolutionDial(WhyTop); break;
    case cmdSolutionCheck: WhichCheckRB(); break;
    case cmdSolutionHelp: SolutionDial(SolutionHelp); break;
    case cmdStartEnEq: SolutionDial(StartCheckSolution); break;
    case cmdWriteEnEq: SolutionDial(WriteEnergyEq); break;
    case cmdSubstQuant: case cmdSolutionRelate: SolutionDial(SolutionRelations); break;
    case cmdgbtn: case cmdhbtn: case cmdlbtn: case cmdMbtn: case cmdrbtn: case cmdRbtn: case
    cmdybbtn: case cmdEqualbtn: case cmdMultbtn: case cmdDividebtn: case cmdAddbtn: case
    cmdSubtbtn: case cmdSqRtbtn: case cmdReturnbtn: case cmdAnyVel: case cmdTopVel: case
    cmdBottomVel: case cmdStartVel: case cmdAnyAngVel: case cmdTopAngVel: case
    cmdBottomAngVel: case cmdAccel: case cmdForces: case cmdReaction: case cmdPhi: case
    cmdBetabtn: theStartDir = new (CStartDirPanel); theStartDir->PlacetheChar(theCommand); break;
    case cmdSquarebtn: theStartDir = new (CStartDirPanel); theStartDir->Squaring(); break;
    case cmdZero: case cmdOne: case cmdTwo: case cmdThree: case cmdFour: case cmdFive: case
    cmdSix: case cmdSeven: case cmdEight: case cmdNine: gBartender->GetCmdText(theCommand,
    theTitle);break;
    case cmdCosine: theEquations = new (CEquations); theEquations->SineCosinePlace('c', 'o', 's');
    break;
    case cmdSine: theEquations = new (CEquations); theEquations->SineCosinePlace('s', 'i', 'n'); break;
    case cmdCheckSolQuit: theSolutionCheck->CloseDialSolution(); theSolutionCheck->
    ThrowSolutionDial(cmdCheckSolQuit); theSolutionCheck = NULL;
    ForgetObject(theSolutionCheck); break;
    case cmdStartCheckSol: SolutionDial(StartCheckSolution); break;
    case cmdCheckSolForce: ForceSolutionCheck(); break;
    case cmdIntSolEnergy: EnergySolutionCheck(); break;
    case cmdGPE: EnergyPos = 1; break;
    }
}

```

```

case cmdTKE: EnergyPos = 2; break;
case cmdRKE: EnergyPos = 3; break;
case cmdGpeTke: EnergyPos = 4; break;
case cmdGpeRke: EnergyPos = 5; break;
case cmdTkeRke: EnergyPos = 6; break;
case cmdGpeTkeRke: EnergyPos = 7; break;
case cmdPosition1: Energy1 = 0; break;
case cmdPosStart1: Energy1 = 1; break;
case cmdPosBot1: Energy1 = 2; break;
case cmdPosTop1: Energy1 = 3; break;
case cmdPosition2: Energy1 = 0; break;
case cmdPosStart2: Energy2 = 1; break;
case cmdPosBot2: Energy2 = 2; break;
case cmdPosTop2: Energy2 = 3; break;
default: inherited::DoCommand(theCommand);
}
}

void CSolution::SolutionDial(short DialogueId)
{
    CDLOGDirector *theInfo; CDialog *aDialog = (CDialog*) itsWindow;

theInfo = new (CDLOGDirector); theInfo->IDLOGDirector(DialogueId, this); theInfo->
DoModalDialog(cmdOK); theInfo->EndDialog(cmdOK, TRUE); theInfo->Close(TRUE); aDialog->
Update(); }

void CSolution::WhichCheckRB(void)
{
    CRadioGroupPane *radioGroupSet;
    radioGroupSet = (CRadioGroupPane *)itsWindow->FindViewByID(kRadioSolutionID);
switch (radioGroupSet -> GetStationID()) {
case kSlidingSolution: ModalDialSolution("\p", "\ph = 2.5 R", "\pANSWER CHECK", "\pSLIDING
BALL", 128); break;
case kRollingSolution: ModalDialSolution("\p", "\ph = 2.7 R", "\pANSWER CHECK",
"\pROLLING BALL", 128); break;
default: ModalDialSolution("\pYou have not selected Sliding or Rolling.", "\pClick the appropriate
choice to obtain the final expression.", "\pANSWER CHECK", "\pMAKE A SELECTION", 128);
break; }
}

void CSolution::ModalDialSolution(Str255 theText0, Str255 theText1, Str255 theText2, Str255
theText3, short IdNum)
{ ParamText(theText0, theText1, theText2, theText3); PositionDialog('ALRT', IdNum); InitCursor;
NoteAlert(IdNum, NULL); return; }

void CSolution::EditWindSolution(short dialogItemNo)
{
    CDialog *theDialog = (CDialog*) itsWindow; tGopherInfo gopherInfo; CView
    *newGopher;
do {
    gopherInfo.currentGopher = (CView*) gGopher; theDialog->FindGophers(&gopherInfo); newGopher
    = gopherInfo.nextGopher; newGopher->BecomeGopher(TRUE);
} while(newGopher->ID != dialogItemNo);
StrLength = ((CEditText*) gGopher)->GetLength(); textEntry = ((CEditText*) gGopher)->
CopyTextRange(0, ((CEditText*) gGopher)->GetLength()); }

void CSolution::ForceSolutionCheck(void)
{
    int i, j, m, grav, v2T, Rad, down, cV, sign;
    i = j = m = grav = v2T = Rad = down = cV = sign = 0;
if (Energy1 == 0) {
    ModalDialSolution("\p", "\pTo commence force equation check, make a selection from \"Position 1\"
for the point at which the equation should be written", "\pPROBLEM SOLUTION", "\pFORCE
EQUATION CHECK", 128); return; }
if (Energy1 != 3) { SolutionDial(WhyNormalTop); return; }
EditWindSolution(ForceEquationSlot);
if ((*textEntry) == 0 || (*textEntry) == '0') && Energy1 == 3){
    ModalDialSolution("\pTo commence, type the force equation for the top into the \"Force equation\"
slot.", "\pNOTE: This section performs minimal check of your equation.\rIf you need help to set up
the force equation, go to the \"Start Panel\" and select \"Write force equations\".", "\pPROBLEM
SOLUTION", "\pFORCE EQUATION CHECK", 128); return; }
for ( i = 0; i <= StrLength - 1; i++)
if ((*textEntry + i) == '=')

```

```

j++;
if (!j){
ModalDialSolution("\p", "\pThere is no \"=\\" symbol in your equation.", "\pPROBLEM SOLUTION",
"\pFORCE EQUATION CHECK", 128); return; }
for ( i = 0; i <= StrLength - 1; i++) {
switch ((*textEntry + i)) {
case '0': case '1': case '2': case '3': case '4': case '5': case '6': case '7':case '8': case '9':
ModalDialSolution("\pYour equation has a number in it.", "\pThe energy equation should not have a
number in it.\rFor more help, go to the \"Start Panel\" and select \"Write force equations\".",
"\pPROBLEM SOLUTION", "\pFORCE EQUATION CHECK", 128); return; break;
case '.': case '*': case '=': break;
case '+': case '-': sign++; break;
case '/': down++; break;
case 'm': ModalDialSolution("\pUse capital M for the mass.", "\p", "\pPROBLEM SOLUTION",
"\pFORCE EQUATION CHECK", 128); return; break;
case -127: v2T++; break;
case 's': if ((*textEntry + i + 1) == 'i' && (*textEntry + i + 2) == 'n') {
ModalDialSolution("\pThe trigonometry function should be cosine not sine", "\pThe component of
the gravitational force in the radial direction is Mgcoss.", "\pPROBLEM SOLUTION", "\pFORCE
EQUATION CHECK", 128); return; } break;
case 'c': if ((*textEntry + i + 1) == 'o' && (*textEntry + i + 2) == 's'){ cV++; i = i + 2; } break;
case 'B': ModalDialSolution("\pAt the top of the loop  $\beta = 180$  degrees, and  $\cos\beta = -1$ ",
"\pIf the ball is in contact with the track at the top of the loop then it will successfully perform the
loop. \rThis is the reason for writing the force equation for the top of the loop.", "\pPROBLEM
SOLUTION", "\pFORCE EQUATION CHECK", 128); return; break;
case -81: ModalDialSolution("\pUse  $\beta$ , not  $\emptyset$ ", "\p $\emptyset$  is the angle of the incline.",
"\pPROBLEM SOLUTION", "\pFORCE EQUATION CHECK", 128); return; break;
case -125: case -124: case -110: case -126: case -109: case 'v': ModalDialSolution("\pWrong velocity
symbol. Use the velocity symbol for the top of the loop.", "\pIf the ball is in contact with the track at
the top of the loop then it will successfully perform the loop. \rThis is the reason for writing the force
equation for the top of the loop.", "\pPROBLEM SOLUTION", "\pFORCE EQUATION CHECK",
128); return; break;
case -123: case -120: case -119: case -118: ModalDialSolution("\pWrong angular velocity symbol.
Use the angular velocity symbol for the top of the loop.", "\pIf the ball is in contact with the track at
the top of the loop then it will successfully perform the loop. \r This is the reason for writing the force
equation for the top of the loop.", "\pPROBLEM SOLUTION", "\pFORCE EQUATION CHECK",
128); return; break;
case 'r': ModalDialSolution("\pUse R instead of r.", "\pR is the radius of the loop performed.",
"\pPROBLEM SOLUTION", "\pFORCE EQUATION CHECK", 128); return; break;
case -128: ModalDialSolution("\pIt should be velocity squared.", "\pSquare the velocity symbol.",
"\pPROBLEM SOLUTION", "\pFORCE EQUATION CHECK", 128); return; break;
case -122: ModalDialSolution("\pIt should be angular velocity squared.", "\pSquare the angular
velocity symbol.", "\pPROBLEM SOLUTION", "\pFORCE EQUATION CHECK", 128); return;
break;
case -121: ModalDialSolution("\pUse  $v = wR$ .", "\pReplace w.", "\pPROBLEM SOLUTION",
"\pFORCE EQUATION CHECK", 128); return; break;
case 'R': Rad++; break;
case 'N': ModalDialSolution("\pIf the ball is in contact with the track then,  $N \geq 0$ .",
"\pAt the top of the loop when N just becomes zero give the minimum starting height, h. \r\r
Substitute  $N = 0$  into your equation.", "\pPROBLEM SOLUTION", "\pFORCE EQUATION
CHECK", 128); return; break;
case 'M': m++; break;
case 'g': grav++; break;
default: ModalDialSolution("\pYou have used a symbol which is either inapplicable or needs to be
substituted.", "\pFor help to check your equation go to \"Start Panel\" and select \"Write force
equation\".", "\pPROBLEM SOLUTION", "\pFORCE EQUATION CHECK", 128); return; break; }
}
if (sign) {
ModalDialSolution("\pThe equation can be further simplified.", "\pThere should be no plus or minus
sign in the simplified form of the equation. \rFor more help go to \"Start Panel\" and select \"Write
force equation\" to perform a check", "\pPROBLEM SOLUTION", "\pFORCE EQUATION CHECK",
128); return; }
if (cV) { ModalDialSolution("\pAt the top of the loop  $\beta = 180$  degrees, and  $\cos\beta = -1$ ", "\pIf the ball
is in contact with the track at the top of the loop then it will successfully perform the loop. \rThis is
the reason for writing the force equation for the top of the loop.", "\pPROBLEM SOLUTION",
"\pFORCE EQUATION CHECK", 128); return; }
else if (m) {

```



```

ModalDialogSolution("\pM should cancel from your force equation.", "\pFor help to check your
equation go to \"Start Panel\" and select \"Write force equation\".", "\pPROBLEM SOLUTION",
"\pFORCE EQUATION CHECK", 128); return; }
else if (!grav) {
ModalDialogSolution("\p\"g\", the acceleration due to gravity should be in your equation.", "\pFor help
to check your equation go to \"Start Panel\" and select \"Write force equation\".", "\pPROBLEM
SOLUTION", "\pFORCE EQUATION CHECK", 128); return; }
else if (!Rad) {
ModalDialogSolution("\p\"R\", the radius of the loop should be in your equation.", "\pFor help to check
your equation go to \"Start Panel\" and select \"Write force equation\".", "\pPROBLEM SOLUTION",
"\pFORCE EQUATION CHECK", 128); return; }
else if (!v2T) {
ModalDialogSolution("\pThe velocity at the top of the loop should be in your equation.", "\pFor help
to check your equation go to \"Start Panel\" and select \"Write force equation\".", "\pPROBLEM
SOLUTION", "\pFORCE EQUATION CHECK", 128); return; }
else if (v2T == 1 && grav == 1 && Rad == 1) {
ModalDialogSolution("\p", "\p", "\pPROBLEM SOLUTION", "\pFORCE EQUATION CHECK
COMPLETED", 2009); return; }
else {
ModalDialogSolution("\pYou have used a symbol which is either inapplicable or needs to be
substituted.", "\pFor help to correct your equation go to \"Start Panel\" and select \"Write force
equation\".", "\pPROBLEM SOLUTION", "\pFORCE EQUATION CHECK", 128); return; }
}

void CSolution::EnergySolutionCheck(void)
{
short theModifiers;
if (!Energy1 && !Energy2) {
ModalDialogSolution("\pSelect a set of positions from \"Position 1\" and \"Position 2\".",
"\pSelect a different position for each of the positions. \pThe selected positions should be those at
which you wish to equate the energies.", "\pSELECT POSITIONS", "\pENERGY EXPRESSION
CHECK", 128); return; }
if (Energy1 == Energy2 && Energy1) {
ModalDialogSolution("\pThe two positions are equal.", "\pOne has to be changed.", "\pSIMILAR
POSITIONS", "\pENERGY EXPRESSION CHECK", 128); return; }
if ((Energy1 && !Energy2) || (!Energy1 && Energy2)) {
ModalDialogSolution("\pA position should be selected for both \"Position 1\" and \"Position 2\".",
"\pA single position is sufficient if you wish to write the energy expression at a point. \pTo check the
expression, select \"Write energy expressions\" in \"Start Panel\".", "\pSELECT POSITIONS FOR
BOTH", "\pENERGY EXPRESSION CHECK", 128); return; }
if ((Energy1 == 1 && Energy2 != 3) || (Energy1 != 3 && Energy2 == 1) || (Energy1 != 1 &&
Energy2 == 3) || (Energy1 == 3 && Energy2 != 1)) { SolutionDialog(EnergyClues); return; }
}
if ((Energy1 == 1 && Energy2 == 3) || (Energy1 == 3 && Energy2 == 1)){
ModalDialogSolution("\pThe two selected positions are correct.", "\pSelect \"Write energy expressions\"
from \"Start Panel\" to write the energy expression.", "\pSELECT START AND TOP", "\pENERGY
EXPRESSION CHECK", 128); return; }
}

```

REFERENCES

- Adams, R. S. and Biddle, B. J. (1970) *Realities of Teaching: Explorations with a videotape*, Holt, Rinehart and Winston, New York.
- Anderson, J. R. (1983) *The Architecture of Cognition*, Harvard University Press, Cambridge, Massachusetts.
- Anderson, J. R. (1985) *Cognitive Psychology and its Implications*, Second Edition, Freeman, New York.
- Anderson, J. R., Boyle, C. F., Corbett, A. T. and Lewis, M. W. (1990) Cognitive Modelling and Intelligent Tutoring, *Artificial Intelligence*, 42(1), 7-49.
- Ausubel, D. P. (1968) *Educational Psychology: A Cognitive View*, Holt, Rinehart and Winston, New York.
- Bailey, K.D. (1978) *Methods of Social Research*, Collier-Macmillan, London.
- Barr, A. and Feigenbaum, E. A. (Eds.) (1982) *The Handbook of Artificial Intelligence Volume 1*, William Kaufmann, Los Altos, California.
- Bartlett, F. C. (1932) *Remembering. A Study in Experimental and Social Psychology*, Cambridge University Press, Cambridge.
- Barzilay, A. (1985) SPIRIT: A Flexible Tutoring Style in an Intelligent Tutoring System. In C. R. Weisbin (Ed.), *Proceedings of the Second Conference on Artificial Intelligence Applications. The Engineering of Knowledge-Based Systems*, Miami Beach, Florida, IEEE Press, North-Holland.
- Beiser, A. (1962) *The Mainstream of Physics*, Addison-Wesley, Reading, Massachusetts.
- Bigge, M. L. (1982) *Learning Theories for Teachers*, Harper and Row, New York.
- Bobrow, D. G. and Norman, D. A. (1975) Some Principles of Memory Schemata. In D. G. Bobrow and A. Collins (Eds.), *Representation and Understanding, Studies in Cognitive Science*, Academic Press, New York.
- Bork, A. M. and Robson, J. (1972) A Computer Simulation for the Study of Waves, *American Journal of Physics*, 40, 1288-1294.
- Brown, D. E. (1992) Using Examples and Analogies to Remediate Misconceptions in Physics: Factors Influencing Conceptual Change, *Journal of Research in Science Teaching*, 29(1), 17-34.
- Brown, J. S. and Burton, R. R. (1975) Multiple Representations of Knowledge for Tutorial Reasoning. In D. G. Bobrow and A. Collins (Eds.), *Representation and Understanding: Studies in Cognitive Science*, Academic Press, New York.
- Brown, J. S. and Burton, R. R. (1978) Diagnostic Models for Procedural Bugs in Basic Mathematical Skills, *Cognitive Science*, 2, 155-192.

- Brown, J. S., Burton, R. R. and Bell, A. G. (1975) SOPHIE: A Step Toward Creating a Reactive Learning Environment, *International Journal of Man-Machine Studies*, 7, 675-696.
- Brown, J. S., Burton, R. R. and deKleer, J. (1982) Pedagogical, Natural Language and Knowledge-Engineering Techniques in Sophie I, II and III. In D. Sleeman and J. S. Brown (Eds.), *Intelligent Tutoring Systems*, Academic Press, New York.
- Brownston, L., Farrell, R., Kant, E. and Martin, N. (1986) *Programming Expert Systems in OPS5: An Introduction to Rule Based Programming*, Addison-Wesley, Reading Massachusetts.
- Bruner, J. (1985) Models of the Learner, *Educational Researcher*, 14(6),5-8.
- Buchanan, B. G., Barstow, D., Bechtal, R., Bennett, J., Clancey, W., Kulikowski, C., Mitchell, T. and Waterman, D. A. (1983) Constructing an Expert system. In F. Hayes-Roth, D. A. Waterman and D. B. Lenat (Eds.), *Building Expert Systems Volume I*, Addison- Wesley, Reading , Massachusetts.
- Bumbaca, F. (1988) Intelligent Computer-Assisted Instruction: A theoretical framework, *International Journal of Man-Machine Studies*, 29(3), 227-255.
- Bundy, A., Byrd, L., Luger, G., Mellish, C. and Palmer M. (1979) Solving Mechanics Problems Using Meta-Level Inference. In D. Michie (Ed.), *Expert Systems in the Micro-electronic Age*, Edinburgh University Press, Edinburgh, Scotland.
- Burton, R. R. and Brown, J. S. (1982) An Investigation of Computer Coaching for Informal Learning Activities. In D. Sleeman and J. S. Brown (Eds.), *Intelligent Tutoring Systems*, Academic Press, New York.
- Butts, W. and Smith, R. (1987) H.S.C. chemistry students' understandings of the structure and properties of molecular structure and ionic compounds, *Research in Science Education*, 17, 192-201.
- Cannell, C. F. and Kahn, R. L. (1968) Interviewing. In D. Lindzey and E. Aronson (Ed.), *The Handbook of Social Psychology, volume 2, Research Methods*, Addison-Wesley, New York.
- Carbonell, J. G. (1986) Derivational Analogy: A Theory of Reconstructive Problem Solving and Expertise Acquisition. In R. S. Michalski, J. G. Carbonell and T. M. Mitchell (Eds.), *Machine Learning: An Artificial Intelligence Approach Volume 2*, Tioga Press, Palo Alto, California.
- Carbonell, J. G., Michalski, R. S. and Mitchell, T. M. (1986) An Overview of Machine Learning. In R. S. Michalski, J. G. Carbonell and T. M. Mitchell (Eds.), *Machine Learning: An Artificial Intelligence Approach*, Tioga Publishing, Palo Alto, California.
- Carbonell, J. R. (1970) AI in CAI: An Artificial Intelligence Approach to Computer-Assisted Instruction, *IEEE Transactions on Man-Machine Systems*, 11, 190-202.

- Carr, M (1991) Methods for Studying Personal Constructions. In J. Northfield and D. Symington (Ed.), *Learning in Science Viewed as Personal Constructions. An Australasian Perspective*, Key Centre for School Science and Mathematics, Curtin University of Technology, Australia.
- Champagne, A. B., Gunstone, R. F. and Klopfer, L. E. (1985) Effecting Changes in Cognitive Structures Among Physics Students. In L. H. T. West and A. L. Pines (Eds.), *Cognitive Structure and Conceptual Change*, Educational Psychology Series, Academic Press, Orlando, Florida.
- Chi, M. T. H., Bassok, M., Lewis, M. W., Reimann, P. and Glaser, R. (1989) Self-Explanations: How Students Study and Use Examples in Learning to Solve Problems, *Cognitive Science*, 13, 145-182.
- Chi, M., Feltovich, P. and Glaser, R. (1981) Categorisation and Representation of Physics Problems by Experts and Novices, *Cognitive Science*, 5, 121-152.
- Clancey, W. J. (1987a) *Knowledge-Based Tutoring: The GUIDON Programme*, MIT Press, Cambridge, Massachusetts.
- Clancey, W. J. (1987b) Methodology for Building an Intelligent Tutoring System. In G. P. Kearsley (Ed.), *Artificial Intelligence and Instruction: Applications and Methods*, Addison-Wesley, Reading, Massachusetts.
- Claxton, G. L. (1985) Teaching and Acquiring Scientific Knowledge, Unpublished Paper, *Centre for Science and Mathematics Education*, University of London, Chelsea College, London.
- Cohen, L. and Manion, C. (1989) *Research Methods in Education*, Routledge, London.
- Collins, A. (1976) Processes in Acquiring Knowledge. In R. C. Anderson, R. J. Spiro and W. E. Montague (Eds.), *Schooling and the Acquisition of Knowledge*, Lawrence Erlbaum, Hillsdale, New Jersey.
- Coombs, A. W. and Snygg, D. (1959) *Individual Behaviour*, Revised Edition, Harper and Row, New York.
- Davis, R. (1989) Expert Systems: How far can they go, Part one, *AI Magazine*, 10(1), 61-67.
- de Kleer, J. (1977) Multiple Representations of Knowledge in a Mechanics Problem-Solver. In *Proceedings of the Fifth International Joint Conference on Artificial Intelligence Volume I*, Massachusetts Institute of Technology, Cambridge, Massachusetts, William Kaufmann, Los Altos, California.
- Driver, R. and Oldham, V. (1986) A Constructivist Approach to Curriculum Development in Science, *Studies in Science Education*, 13, 105-122.
- Duchastel, P. C. (1988) designing Intelligent Learning Environments. In H. Mathias, N. Rushby and R. Budgett (Eds.), *Aspects of Educational Technology Volume*

- XXI: Designing New Systems and Technologies for Learning*, Kogan Page, London.
- Ericsson, K.A. and Simon, H.A. (1985) *Protocol Analysis: Verbal Reports As Data*, MIT Press, Cambridge, Massachusetts.
- Fischer, H. E. and Aufschnaiter, S. V. (1993) Development of Meaning During Physics Instruction: Case Studies in View of the Paradigm of Constructivism, *Science Education*, 77(2), 153-168.
- Gagne, E. D. (1985) *The Cognitive Psychology of School Learning*, Little, Brown and Company, Boston.
- Gagne, R. M. (1977) *The Conditions of Learning*, Holt, Rinehart and Winston, New York.
- Gamow, G. and Cleveland, J. M. (1976) *Physics: Foundations and Frontiers*, Prentice Hall, Englewood Cliffs, N.J.
- Gardner, H. (1985) *The Mind's Science: A History of the Cognitive Revolution*, Basic Books, New York.
- Garg-Janardan, C. and Salvendy, G. (1988) A Structured Knowledge Elicitation Methodology for Building Expert Systems, *International Journal of Man-Machine Studies*, 29(4), 377-406.
- Garrett, R. M. (1986) Problem-Solving in Science Education, *Studies in Science Education*, 13, 70-95.
- Genesereth, M. R. (1982) The Role of Plans in Intelligent Teaching Systems. In D. Sleeman and J. S. Brown (Eds.), *Intelligent Tutoring Systems*, Academic Press, New York.
- George, F. H. (1970) *Models of Thinking*, George Allen and Unwin, London.
- Glasson, G. E. and Lalik, R. V. (1993) Reinterpreting the Learning Cycle from a Social Constructivist Perspective: A Qualitative Study of Teacher's Beliefs and Practices, *Journal of Research in Science Teaching*, 30(2), 187-207.
- Goldstein, I. P. (1979) The Genetic Epistemology of Rule Systems, *International Journal of Man-Machine Studies*, 11, 51-77.
- Goldstein, I. P. (1982) The Genetic Graph: A Representation for the Evolution of Procedural Knowledge. In D. Sleeman and J. S. Brown (Eds.), *Intelligent Tutoring Systems*, Academic Press, New York.
- Grimellini-Tomasini, N. and Pecori-Balandi, B. (1993) Understanding Conservation Laws in Mechanics: Students' Conceptual Change in Learning about Collisions, *Science Education*, 77(2), 169-189.
- Gunstone, R. F. (1984) Circular Motion: Some pre-instruction alternative frameworks, *Research in Science Education*, 14, 125-136.
- Gunstone, R. F. (1991) Constructivism and metacognition: Theoretical issues and classroom studies. In R. Duit, F. Goldberg and H. Niedderer (Eds.), *Research in*

- Physics Learning: Theoretical Issues and Empirical Studies*, Proceedings of an International Workshop, University of Bremen, Germany, Institute of Science Education, 129 - 140.
- Harmon, H. H. (1960) *Modern Factor Analysis*, University of Chicago Press, Chicago.
- Hartley, J. R. (1988) Learning from Computer-Based Learning in Science, *Studies in Science Education*, 15, 55-76.
- Hayes-Roth, F., Waterman, D. A. and Lenat, D. B. (1983) An Overview of Expert Systems. In F. Hayes-Roth, D. A. Waterman and D. B. Lenat (Eds.), *Building Expert Systems*, Addison-Wesley, Reading, Massachusetts.
- Heller, J. I. and Reif, F. (1984) Prescribing Effective Human Problem-Solving Processes: Problem Descriptions in Physics, *Cognition and Instruction*, 1, 177-216.
- Hoinville, G. and Jowell, R. (1978) *Survey Research Practice*, Heinemann Educational Books, London.
- Johnson, W. L. (1990) Understanding and Debugging Novice Programs, *Artificial Intelligence*, 42(1), 51-97.
- Kearsley, G. P. (Ed.) (1987) *Artificial Intelligence and Instruction. Applications and Methods*, Addison-Wesley, Reading, Massachusetts.
- Kelly, G. A. (1955) *The Psychology of Personal Constructs*, Norton, New York.
- Kimball, R. (1982) A Self-Improving Tutor for Symbolic Integration. In D. Sleeman and J. S. Brown (Eds.), *Intelligent Tutoring Systems*, Academic Press, New York.
- Kinnear, J. F. and Simmons, P. E. (1990) "Expert" and "Novice" Subjects' Approaches to Genetic Problem Solving in a Computer-Based Setting, *Research in Science Education*, 20, 171-180.
- Kinnear, J. F. and Simmons, P. E. (1990) "Expert" and "Novice" Subjects' Approaches to Genetic Problem Solving in a Computer-Based Setting, *Research in Science Education*, 20, 171-180.
- Koballa Jr., T. R., Crawley, F. E. and Shrigley, R. L. (1990) A summary of Research in Science Education - 1988, *Science Education*, 74(3), 334-351.
- Kolodner, J. L. (1984) *Retrieval and Organisational Strategies in Conceptual Memory: a Computer Model*, Lawrence Erlbaum, Hillsdale, New Jersey.
- Krutetskii, V. A. (1976) *The Psychology of Mathematical Problem Solving*, University of Chicago Press, Illinois.
- Langley, P. (1979) Rediscovering Physics with BACON.3. In *Proceedings of the Sixth International Joint Conference on Artificial Intelligence Volume I*, Tokyo, Japan.
- Larkin, J. H. (1980) Teaching Problem Solving in Physics: The Psychological Laboratory and the Practical Classroom. In D. T. Tuma and F. Reif (Eds.),

- Problem Solving and Instruction: Issues in Teaching and Research*, Lawrence Erlbaum, Hillsdale, New Jersey.
- Larkin, J. H. (1981) Enriching Formal Knowledge: A Model for Learning to Solve Textbook Physics Problems. In J. R. Anderson (Ed.), *Cognitive Skills and Their Acquisition*, Lawrence Erlbaum, Hillsdale, New Jersey.
- Larkin, J.H. and Rainard, B. (1984) A Research Mehtodology For Studying How People Think, *Journal of Research in Science Teaching*, 21(3), 235-254.
- Lindsay, S. (1988) *Practical Applications of Expert Systems*, QED Information Sciences, Wellesley, Massachusetts.
- Lofland, J. (1971) *Analysing Social Settings*, Wadsworth, Belmont, California.
- Maslow, A. H. (1987) *Motivation and Personality*, Revised Third Edition, Harper and Row, New York.
- McCloskey, M. (1983) Intuitive Physics, *Scientific American*, 248(4).
- McDermott, J. (Ed.) (1987) *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, Milan, August 23-28, Morgan Kaufmann, Los Altos, California.
- Merton, R. K. and Kendall, P.L. (1946) The focused interview, *American Journal of Sociology*, 51, 541-557
- Mestre, J. and Touger, J. (1989) Cognitive Research - What is in it for Physics Teachers ?, *The Physics Teacher*, 27(6), 447-456.
- Michalski, R. S. (1986) A Theory and Methodology of Inductive Learning. In R. S. Michalski, J. G. Carbonell and T. M. Mitchell (Eds.), *Machine Learning: An Artificial Intelligence Approach*, Tioga Publishing, Palo Alto, California.
- Miller, M. L. (1982) A Structured Planning and Debugging Environment for Elementary Programming. In D. Sleeman and J. S. Brown (Eds.), *Intelligent Tutoring Systems*, Academic Press, New York.
- Millward, R. B. (1980) Models of Concept Formation. In R. E. Snow, P. Frederico and W. E. Montague (Eds.), *Aptitiude, Learning and Instruction Volume 2: Cognitive Process Analyses of Learning and Problem Solving*, Lawrence Erlbaum, Hillsdale, New Jersey.
- Minsky, M. (1975) A Framework for Representing Knowledge. In P. H. Winston (Ed.), *The Psychology of Computer Vision*, McGraw-Hill, New York.
- Minsky, M. (Ed.)(1968) *Semantics Information Processing*, MIT Press, Cambridge, Massachusetts.
- Nawrocki, L. H. (1987) Artificial Intelligence Applications to Maintenance Training. In G. P. Kearsley (Ed.), *Artificial Intelligence and Instruction: Applications and Methods*, Addison-Wesley, Reading, Massachusetts.
- Newell, A. and Simon, H. A. (1972) *Human Problem Solving*, Prentice-Hall, Englewood Cliffs, New Jersey.

- Novak Jr., J. S. (1977) Representations of Knowledge in a Program for Solving Physics Problems. In *Proceedings of the Fifth International Joint Conference on Artificial Intelligence Volume I*, Massachusetts Institute of Technology, Cambridge, Massachusetts, William Kaufmann, Los Altos, California.
- Novak, J. D. (1988) Learning Science and the Science of Learning, *Studies in Science Education*, 15, 77-101.
- O'Shea, T. (1982) A Self-Improving Quadratic Tutor. In D. Sleeman and J. S. Brown (Eds.), *Intelligent Tutoring Systems*, Academic Press, New York.
- Ogborn, J. (1987) Computational Modelling in Science. In R. E. Lewis and E. D. Togg (Eds.), *Trends in Computer-Assisted Education*, Blackwell Scientific Publications, England.
- Osborn, A. F. (1963) *Applied Imagination*, Third Edition, Scribner's, New York.
- Osborne, R. and Freyberg, P. (1985) *Learning in Science: The implications of children's science*, Heinemann, Auckland, New Zealand.
- Osborne, R. and Wittrock, M. (1985) The Generative Learning Model and its Implications for Science Education, *Studies in Science Education*, 12, 59-87.
- Park, O., Perez, R. S. and Seidel, R. J. (1987) Intelligent ICAI: Old Wine in New Bottles, or a New Vintage? In G. P. Kearsley (Ed.), *Artificial Intelligence and Instruction. Applications and Methods*, Addison-Wesley, Reading, Massachusetts.
- Pinard, A. and Laurendeau, M. (1969) "Stage" in Piaget's Cognitive-Developmental Theory: Exegesis of a Concept. In D. Elkind and J. H. Flavell (Eds.), *Studies in Cognitive Development: Essays in Honor of Jean Piaget*, Oxford University Press, New York.
- Pizzini, E. L., Shepardson, D. P. and Abell, S. K. (1989) A Rationale for and the Development of a Problem-Solving Model of Instruction in Science Education, *Science Education*, 73(5), 523-534.
- Polya, G. (1971) *How to Solve it; A New Aspect of Mathematical Method*, Princeton University Press, Princeton, New Jersey.
- Putnam, H. (1988) Much Ado About Not Very Much. In S. R. Graubard (Ed.), *The Artificial Intelligence Debate. False Starts, Real Foundations*, MIT Press, Cambridge, Massachusetts.
- Reif, F. (1981) Teaching Problem Solving - A Scientific Approach, *The Physics Teacher*, 19, 310-316.
- Reif, F. (1983) Understanding and Teaching Problem Solving in Physics. In La Launde Les Maures (Ed.), *Research on Physics Education: Proceedings of the First International Workshop. Editions du Centre National de la Recherche Scientifique*, Paris, France.

- Reif, F. (1985) Acquiring an Effective Understanding of Scientific Concepts. In L.H. T. West and A. L. Pines (Eds.), *Cognitive Structure and Conceptual Change*, Educational Psychology Series, Academic Press, Orlando, Florida.
- Reif, F. and Heller, J. (1982) Knowledge Structure and Problem Solving in Physics, *Educational Psychologist*, 17(2), 102-127.
- Resnick, R. and Halliday, D. (1988) *Physics*, Third Edition, Wiley and Sons, New York.
- Rolston, D.W. (1988) *Principles of Artificial Intelligence and Expert Systems Development*, McGraw Hill, New York.
- Rubinstein, M. F. (1975) *Patterns of Problem Solving*, Prentice-Hall, Englewood Cliffs, New Jersey.
- Rumelhart, D. E. and Norman, D. A. (1981) Analogical Processes in Learning. In J. R. Anderson (Ed.), *Cognitive Skills and their Acquisition*, Lawrence Erlbaum, Hillsdale, New Jersey.
- Schank, R. C. and Abelson, R. P. (1977) *Scripts, Plans, Goals and Understanding. An Inquiry into Human Knowledge Structures*, Lawrence Erlbaum, Hillsdale, New Jersey.
- Simmons, G. L. (1984) *Introducing Artificial Intelligence*, National Computing Centre Publications, Manchester, England.
- Simon, H. A. (1980) Problem Solving and Education. In D. Tuma and F. Reif (Eds.), *Problem Solving and Education: Issues in Teaching and Research*, Lawrence Erlbaum, Hillsdale, New Jersey.
- Singh, A. (1989) *Knowledge Structures and Physics Teaching*, Unpublished master's thesis, University of Canterbury, New Zealand.
- Skinner, B. F. (1969) *Contingencies of Reinforcement*, Appleton-Century-Crofts, New York.
- Sleeman, D. (1982) Assessing Aspects of Competence in Basic Algebra. In D. Sleeman and J. S. Brown (Eds.), *Intelligent Tutoring Systems*, Academic Press, New York.
- Sleeman, D. (1987) Micro Search: A "Shell" for Building Systems to Help Students Solve Non Deterministic Tasks. In G. P. Kearsley (Ed.), *Artificial Intelligence and Instruction: Applications and Methods*, Addison-Wesley, Reading, Massachusetts.
- Sleeman, D. and Brown, J. S. (Eds.) (1982) *Intelligent Tutoring Systems*, Academic Press, New York.
- Sleeman, D. H. and Hendley, R. J. (1982) ACE: A System Which Analyses Complex Explanations. In D. Sleeman and J. S. Brown (Eds.), *Intelligent Tutoring Systems*, Academic Press, New York.
- Smith, R. L., Graves, W. H., Blaine, L. H, and Marinov, V. G. (1975) Computer Assisted Axiomatic Mathematics: Informal Rigor. In O'Lacarme and R. Lewis (Eds.),

- Computers in Education, Proceedings of the International Federation of Information Processing*, American Elsevier, Amsterdam, North Holland, 803-809.
- Sokolov, A. N. (1972) *Inner Speech and Thought*, Plenum Press, New York.
- Stake, R.E. (1978) The case study method in social inquiry, *Educational Researcher*, February, 5-8.
- Stanford, A. L. and Tanner, J. M. (1985) *Physics for Students of Science and Engineering*, Academic Press, Orlando, Florida.
- Staver, J. R., Enochs, L. G., Koeppe, O. J., McGrath, D., McLellan, H., Oliver, J. S., Scharmann, L. C. and Wright, E. L. (1989) A summary of Research in Science Education - 1987, *Science Education*, 73(3), 248-249 & 307-313.
- Stevens, A. A. and Collins, A. (1980) Multiple Conceptual Models of a Complex System. In R. E. Snow, P. Frederico and W. E. Montague (Eds.), *Aptitude, Learning and Instruction Volume 2: Cognitive Process Analyses of Learning and Problem Solving*, Lawrence Erlbaum, Hillsdale, New Jersey.
- Stevens, A. L., Collins, A. and Goldin, S. (1979) Misconceptions in Students Understanding, *International Journal of Man-Machine Studies*, 11, 145-156.
- Stevens, A. L., Collins, A. and Goldin, S. (1982) Misconceptions in Students Understanding, In D. Sleeman and J. S. Brown (Eds.), *Intelligent Tutoring Systems*, Academic Press, New York.
- Stewart, J., Streibel, M., Collins, A. and Jungck, J. (1989) Computers as Tutors: MENDEL as an Example, *Science Education*, 73(2), 225-242.
- Tait, K. and Hughes, I. E. (1984) *Some Experiences in Using a Computer-Based Learning System as an Aid to Self-Teaching and Self-Assessment*, *Computer Education*, 8(3), 271-284.
- Thorndike, E.L. (1949) *Selected Writings from a Connectionist's Psychology*, Appleton-Century-Crofts, New York.
- Tipler, P. A. (1991) *Physics*, Worth Publishers, New York.
- Tuckman, B.W. (1972) *Conducting Educational Research*, Harcourt Brace Jovanovich, New York.
- van Weeren (1983) Problem Solving. In La Launde Les Maures(Ed.), *Research on Physics Education: Proceedings of the First International Workshop. Editions du Centre National de la Recherche Scientifique*, Paris, France.
- Vitale, B. M. (1983) *Unicorns are Real*, Jalmar Press, Rolling Hill Estates, California.
- Waterman, D. A. (1986) *A Guide to Expert Systems*, Addison-Wesley, Reading, Massachusetts.
- Webb, G. I. (1988) A Knowledge-Based Approach to Computer-Aided Learning, *International Journal of Man-Machine Studies*, 29(3), 257-285.

- Weidner, R. T. and Sells, R. L. (1966) Elementary Classical Physics Volume 1, Allyn and Bacon, Boston, U.S.A.
- Wheatley, G. H. (1991) Constructivist perspectives on science and mathematics learning, *Science Education*, 75, 9-21.
- White, B. Y. and Frederiksen, J. R. (1990) Causal Model Progressions as a Foundation for Intelligent Learning Environments, *Artificial Intelligence*, 42(1), 99-157.
- White, R. and Gunstone, R. (1992) Probing Understanding, Falmer Press, London.
- Wilson, J. D. (1981) *Student Learning in Higher Education*, John Wiley and Sons, New York.
- Wolfram, S. (1991) *Mathematica: A System for Doing Mathematics by Computer*, Second edition, Addison-Wesley, Redwood, California.
- Woolf, B. P. (1987) Theoretical Frontiers in Building a Machine Tutor. In G. P. Kearsley (Ed.), *Artificial Intelligence and Instruction: Applications and Methods*, Addison-Wesley, Reading, Massachusetts.
- Yasdi, R. and Ziarko, W. (1988) An Expert System for Conceptual Schema Design: A Machine Learning Approach, *International Journal of Man-Machine Studies*, 29(4), 351-376.