Working Paper Series
ISSN 1177-777X

COMPUTER GRAPHICS TECHNIQUES FOR MODELING PAGE TURNING

Veronica Liesaputra and Ian H. Witten

Working Paper: 08/2007
October 2007

Department of Computer Science
The University of Waikato
Private Bag 3105
Hamilton, New Zealand

Computer Graphics Techniques for Modeling Page Turning

Veronica Liesaputra, Ian H. Witten

Department of Computer Science, University of Waikato, Hamilton, New Zealand
e-mail: {vl6, ihw}@cs.waikato.ac.nz

December 13, 2007

Abstract. Turning the page is a mechanical part of the cognitive act of reading that we do literally unthinkingly. Interest in realistic book models for digital libraries and other online documents is growing. Yet actually producing a computer graphics implementation for modeling page turning is a challenging undertaking. There are many possible foundations: two-dimensional models that use reflection and rotation; geometrical models using cylinders or cones; mass-spring models that simulate the mechanical properties of paper at varying degrees of fidelity; finite-element models that directly compute the actual forces within a piece of paper. Even the simplest methods are not trivial, and the more sophisticated ones involve detailed physical and mathematical models. The variety, intricacy and complexity of possible ways of simulating this fundamental act of reading is virtually unknown.

This paper surveys computer graphics models for page turning. It combines a tutorial introduction that covers the range of possibilities and complexities with a mathematical synopsis of each model in sufficient detail to serve as a basis for implementation. Illustrations are included that are generated by our implementations of each model. The techniques presented include geometric methods (both two- and three-dimensional), mass-spring models with varying degrees of accuracy and complexity, and finite-element models. We include a detailed comparison of experimentally-determined computation time and subjective visual fidelity for all methods discussed. The simpler techniques support convincing real-time implementations on ordinary workstations.

1 Introduction

Digital libraries invariably present their documents in a manner that is rather bland. Much electronic text is formatted for the screen in a way that is crude compared with typeset book pages. Some e-book designers do pay attention to look and feel, with crisp text, clearly formatted and attractively laid out (Henke, 2001). Many digital library collections offer page images rather than electronic text, and although these can be rather beautiful they are presented in a flat, two-dimensional manner.

In recent years there has been increased interest in modeling the act of turning a page. The British National Library's "Turning the pages" (British Library, 2006) is a pioneering project that aims to provide a reading experience closely resembling a real book. Readers view a screen showing a double-page spread of what looks like a physical rather than an electronic book. By wiping a finger across the touch-sensitive screen they metaphorically pick up a page and turn it. Pages look three-dimensional; the book's binding moves slightly in sympathy as a page is turned; page edges to right and left indicate how far through the book you are.

To accomplish this, photographs have been taken at several intermediate points during each page turn—so that what is displayed is a stored photo, not an artifact computed from a model of the book. There are many images—for example, one version of the system consumes 300 Mb for only twenty book pages (this includes zoomed-in versions of each page, and accompanying audio). The system is constructed using Macromedia Director. The simulation is compelling, and users rapidly become absorbed in the book itself, turning pages unthinkingly. In Coleridge's words, they "willingly suspend disbelief" in these "shadows of imagination." But the main drawback is that a slow animation of every page-turn must be painstakingly photographed in advance, for each book. This is not feasible as a delivery mechanism for online digital libraries.

Several more practical techniques for modeling the act of turning a page have been proposed (Card et al., 2006; Provot, 1995; Choi & Ko, 2002; Gotoda, 2000;

Bathe, 1995). These include a simple two-dimensional effect that allows the lower left-hand corner to be moved and shows the paper as though it were creased flat, the position of the crease being adjusted to follow the motion; three-dimensional geometric techniques that wrap the paper around a cylinder or cone, physical simulations based on mass-spring models of the paper; and finite-element models that divide the paper into small elements and consider the physical constraints on each one, including its interactions with its neighbors. The aim of this paper is to provide a detailed survey and explanation of all these methods.

Once the page-turning act has been simulated, many more details must be attended to before arriving at a compelling model of a realistic book (Chu et al., 2004). These include modeling the binding of the book, the motion of its covers, the interaction between the pages and the covers, and the effects of metadata such as usage and sectionalization. Although these details are crucial to the implementation of a full and realistic physical book metaphor, they are not discussed in this paper. It turns out that describing different techniques for page turning is a sufficiently ambitious goal.

There is a basic distinction between geometric and physical simulations. Ignoring the physical properties of the paper, the former define appearance by a set of geometric equations, each of which relates to a particular user action. For example, one action represents turning the page from its lower corner, while another simulates the paper being folded. Of course, there is an infinitude of possible user actions and it is impossible to know them all in advance: in practice, one picks a limited set and restricts users accordingly. In this paper we only consider the case when a user picks the bottom corner of the page and turns it from the right to left side of the book; the algorithms are readily adjusted for the more general case where the user grasps the top left corner or (less usually) any point down the right-hand edge or even the top or bottom edge. We also assume that initially the page lies flat on the XY plane. We describe one two-dimensional geometric model and two three-dimensional geometric models.

Physical models are more realistic than geometric ones but considerably more demanding to compute. We describe three that use mass-spring models of the interactions between the particles comprising the paper. The masses and springs form a mesh, and the forces are summed over the mesh points and then integrated over time to obtain the velocity and acceleration of these points. The first model copes only with paper that has cloth-like flexibility; the second allows the paper to bend in more natural curves but still suffers from a unnatural twisting effect; while the third uses a more sophisticated model to rectify this twisting problem.

A more comprehensive physical modeling technique is the finite-element method, which calculates the force, velocity, and acceleration for each small element of the paper. This is a rather complex operation, though for paper it is simpler than a full finite-element analysis because the material forms a thin shell that need not be split into elements in the Z direction (perpendicular to the paper surface).

Descriptions of these algorithms are scattered around the literature, and vary greatly in style and prerequisite knowledge. The aim of this article is to describe them all in a unified way, in sufficient detail to allow them to be understood, compared, debated, implemented, and tested.

2 Two-dimensional geometric model

Imagine turning over the lower right-hand corner of a page and creasing it flat to reveal a triangular-shaped region of the page beneath—a "dog-ear"—with a corresponding triangular region that shows the text on the obverse side of the page. Imagine creating a sequence of successively larger dog-ears. This would be difficult physically (and would make a creased mess of the page), but is trivial in a computer model—and not messy at all. As the motion continues the triangle grows and becomes a quadrilateral when it eventually subsumes the top right-hand corner of the page.

Figure 1 shows this peeling effect.Although the underlying model is entirely two-dimensional, visual details have been added to simulate the effect of a smooth bend rather than a sharp crease: some shading on the bend and some shadowing just beneath it. It is easy to perform the computation in real time as the page is turned. The method was proposed by Beaudouin-Lafon (2001) to handle overlapping windows, and has been used in another page-turning project (Bhangal, 2004).

Although this simple model does not look terribly realistic in the static pictures of Figure 1, it is surprisingly effective in practice. The reader defines the path of the corner of the page as it turns: in effect they gesture with the mouse or touch-screen and the corner of the page follows instantly—whether the motion is straight across to the left, or directly upwards, or even up and to the right. There is complete freedom to move the corner of the page (within the physical constraint imposed by not tearing the paper), and the crease and visual shading details follow instantly. The simulation is satisfyingly reactive.

Fig. 1. Page turning using a two-dimensional geometric model

This technique involves partitioning the page into three sections: the visible portion of the page being turned,

part of the obverse side of the page that the turn has made visible, and the part of the following page that has been revealed. These regions are shaded differently in Figure 2. The region formed by the crease (dark gray) can either be triangular, as in Figure 2(a), or quadrilateral, as in Figure 2(b). The area revealed (light gray) has exactly the same shape reflected in the axis formed by the crease. In these figures the page's initial position is the rectangle $ABCD$, and the act of turning has moved the lower right corner C to position P. The creased region is either the triangle PRS in Figure 2(a) or the quadrilateral $PQRS$ in Figure 2(b). The location of points Q (if applicable), R and S are calculated from the position of point P.

Figures 2(c) and 2(d) show the geometry of the two situations. The crease is the perpendicular bisector of the line PC. The triangular configuration of Figure 2(c) occurs when this bisector intersects the right-hand edge BC of the original page; the quadrilateral configuration of Figure 2(d) occurs when it intersects the top edge AB. This allows points R and S to be calculated. In Figure 2(d) we also need to calculate Q, which is obtained by drawing a line from corner B parallel to CP, noting where it intersects the crease line (point M), and producing it for an equal distance to locate point Q—effectively ensuring that the crease line is a perpendicular bisector of QB as well as of PC.

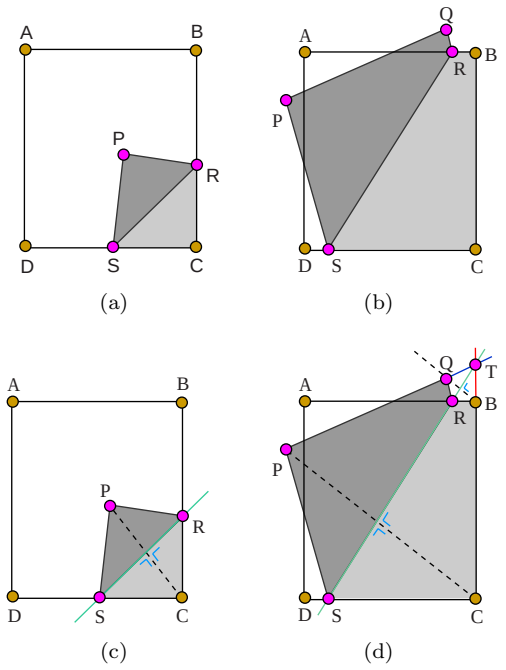


Fig. 2. The appearance of the page: (a) when R intersect BC (b) when R intersect AB. (c) and (d) shows the geometry of the page at situation (a) and (b), respectively.

The paper imposes a physical constraint on where the page corner P can be, for the distance PS cannot exceed the length of CD. This constrains P to lie within the circle shown in Figure 3; otherwise the page would be torn from the spine. If the user moves the mouse outside the circle (say to position P_m) it should be silently mapped to a point P on the arc of the circle. This feels perfectly natural because the system provides instant feedback by drawing the crease and filling in the contents of the region. Note that although we have described the technique using a portrait rather than a landscape layout, and turning from the lower right corner of the page instead of the upper right corner, it is straightforward to extend it to cover more general cases, including ones where the user picks up the page along an edge rather than at a corner.

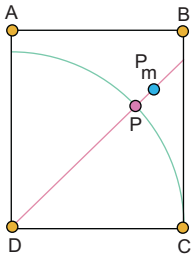


Fig. 3. Mapped position of the mouse

The computational complexity of this technique is constant, independent of any grid size, and in practice the system responds instantaneously to user actions. The method can be implemented entirely in Macromedia Flash. Three images are maintained: the facing and reverse side of the page being turned, and the facing side of the page beneath. To show the creased region, the page's obverse image is rotated to position its lower corner at P and upper corner at Q (if applicable), and a mask whose shape corresponds to the creased region is applied. To reveal the facing image of the page beneath, a mask shaped like the triangle RCS or quadrilateral $RBCS$ is applied.

A shadow effect can be created using a transparent bitmap image that includes appropriate shading for the revealed part: it is rotated to place its centerline along SR and masked appropriately. Shading can be applied on both sides of the crease, not just to the page beneath. In addition, a small shadow can be applied to the top of the visible edge of the page being turned. These subtle effects, which can be seen by examining Figure 1 closely, enhance the perception that the page is being turned in three dimensions. In addtion, pages can easily be made slightly transparent so that readers see a hint of the page underneath.

A visual shortcoming of the peeling effect is that the creased area PRS is *exactly* the same as the revealed area CRS (or, in the quadrilateral case, $PQRS$ is exactly the same as $CBRS$). In a true three-dimensional page-turning situation the extension of the paper in the Z direction will cause the turned area to be smaller than

the revealed area. The three-dimensional bending of the page can be simulated by adding a Z value to each point P, Q, R and S, and utilizing a spline function to make the bend look curvaceous instead of sharply creased. This makes the turned area smaller than the revealed area. The z-value of P can be assigned using a simple heuristic, and suitable values for Q, R and S calculated from it. In practice this necessitates segmenting the paper into a regular grid of size $n \times m$ (n and m differ in proportion to the paper's aspect ratio). The use of a spline function raises the complexity of the technique from constant to $O(n^2)$.

3 Three-dimensional geometric model

In two dimensions the paper is creased and visual effects incorporated to make the crease look reasonably realistic. An alternative simulation of how the paper bends can be created in three dimensions by shaping the turned page as a cylinder or cone. There are many possible ways of doing this. One is to wrap the page around an imaginary cylinder or cone that rests on the XY plane, touching it along the Y axis (which corresponds to the spine of the book). The radius of the cylinder, or the angle of the cone and the position of its apex on the negative Y axis, are varied to provide different degrees of curling.

As in the previous technique, the user controls the page turn by moving the point P that corresponds to its lower left-hand corner. In the cylinder model only the horizontal position (x-coordinate) of P can be controlled, and the page has exactly the same curvature all the way along the Y axis. In the conical model both the x and y value of P can be controlled, and the position of the cone's apex is adjusted to fit. In both cases the height (z-value) of P is calculated according to a simple heuristic. Of course, if a three-dimensional input device were available the user could control the height of P.

The first stage of the computation is to determine the parameters of the cylinder or cone from the position of P. In the second stage, the page is modeled as a uniform mesh of $m \times n$ points and as it is turned it is necessary to calculate where each point maps to on the cylinder or cone.

3.1 Cylindrical model

Imagine wrapping the paper around a cylinder of radius r. At the beginning, the radius is effectively infinite and the page lies flat on the XY plane. The cylinder axis is located along the Y axis. As the turn proceeds, the cylinder's radius is gradually reduced until the midpoint is reached. This causes the page to curl around the cylinder, lifting its right-hand edge from the plane. From the midpoint onward, the radius gradually increases and the cylinder axis moves down and to the left, uncurling the page, until the radius becomes infinite when fully turned and the page lies flat on the left-hand side of the book.

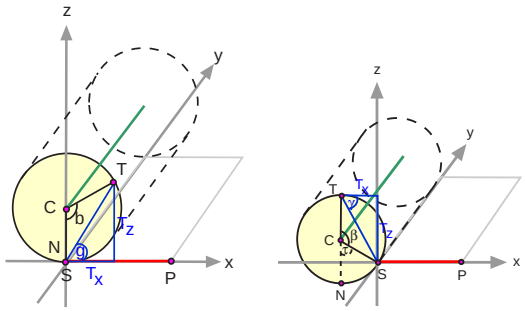


Fig. 4. Mapping a point to a cylinder

Let us assume that a user has moved the bottom right corner of the page $\boldsymbol{P}$ to position $\boldsymbol{T}$. Based on the position of $\boldsymbol{T}$, define a cylinder of radius r whose circular base passes point $\boldsymbol{T}$ and intersects the Y axis at point $\boldsymbol{S}$, where the length of the arc $\boldsymbol{ST}$ equals the line $\boldsymbol{SP}$, as shown in Figure 4. If $\boldsymbol{T}$ has not passed the midpoint, the angle between $\boldsymbol{CN}$ and $\boldsymbol{CS}$ is 0. Otherwise, the angle between $\boldsymbol{CN}$ and $\boldsymbol{CS}$ is $180° - \beta$. Point $\boldsymbol{C}$ is the centre of the circle that lies on the cylinder's axis. The x-value of each mesh point is mapped to the cylinder, so $O(n)$ operations are involved.

The result of this wrapping is rather stilted, and the page appears unnaturally rigid. The turning path parallels the X axis, which is not usually appropriate for single pages but can be useful when an entire multi-page section of a book is turned at once.

3.2 Conical model

Readers seldom turn a page with a path that is parallel to the X axis. They generally lift the corner of the page and move it diagonally upward until it reaches the page midpoint, then diagonally downward from the midpoint onwards. Thus, only the grabbed corner should be lifted ahead of the mesh. To achieve this, Card et al. (2006) use a conical model instead of a cylindrical one. The page turning mechanism is similar to the cylindrical model except that it uses a cone instead of a cylinder. The shape of the cone is defined by its angle θ and apex location $\boldsymbol{A} = (0, A_y, 0)$. Different page trajectories can be obtained by varying these values.

Assume the reader has moved the bottom right corner of the page $\boldsymbol{P}$ to position $\boldsymbol{T}$. Define a cone of radius r whose circular base passes through $\boldsymbol{T}$ and intersects the Y axis at the point $\boldsymbol{S}$ where the length of the arc $\boldsymbol{ST}$ equals the line $\boldsymbol{SP}$, as shown in Figure 5. As in the cylindrical model, if $\boldsymbol{T}$ has not passed the midpoint, the angle between $\boldsymbol{CN}$ and $\boldsymbol{CS}$ is 0; otherwise it is $180° - \beta$. Point $\boldsymbol{C}$ is the centre of the circle that lies on the cone's axis. Given the angle β between point $\boldsymbol{T}$, the centre of the cross section, and point $\boldsymbol{S}$, $\boldsymbol{T}$ can be obtained by first rotating $\boldsymbol{S}$ around the line $X = 0, Z = r$ through

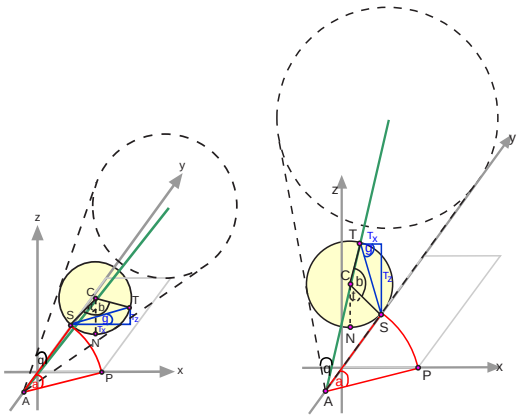


Fig. 5. Wrapping a page around a cone

an angle of β to the new position $\boldsymbol{S}' = (S'_x, S'_y, S'_z)$, and then rotating it around the line $Y = S_y, Z = 0$ through an angle of θ. The radius r of the circle at each cross section varies, so the deformed page is defined by mapping each mesh point onto the cone. The computational complexity of the algorithm is $O(n^2)$.

4 Physical simulation using particle models

In particle models, the paper is simulated by dividing the material into a grid of small elements and modeling each element as though it were a single "particle." The accuracy of the simulation is determined by the size of the grid: the result is more accurate with more elements, but needs more computation. We divide the sheet of paper into $m \times n$ small patches on a two-dimensional rectangular mesh. In reality the patches flex, which is what gives paper its mechanical properties, but here we ignore this and treat each patch as though it were atomic.

In this model, the mechanical properties of the paper are simulated by a lattice of springs rather than the elasticity of the material itself. These springs, which are mass-less, model the interactions between the particles that comprise the paper. The force that the springs exert on each particle is calculated: this is called the "internal" force. "External" forces include the user's page-turning action, constraints imposed by the book's spine, and the force of gravity. Both types of force are summed and the result is used to determine the deformation of the paper at each given point in time. This kind of physical simulation has three major components: mesh representation; forces, both internal and external; and time integration.

Three different particle models are considered in this section. The first is for paper that flexes in a way that resembles cloth more than paper, the second allows more natural curved bends to develop, while the third incorporates a more sophisticated model of bending based on dihedral angles between constituent planes. In each case we discuss the mesh representation, the forces applied and exerted, and how results are obtained by integrating the system of equations over time.

We denote by $\boldsymbol{p}_{ij}$ the particle at grid point i, j, where $i = 1, 2, ..., m$ and $j = 1, 2, ..., n$. These positions depend on the current time h, but to simplify the notation we do not show this dependence explicitly. The particle has a mass of m_{ij}. It is acted upon by the external forces of gravity and, depending on the particular values of i and j, the user's page-turning action and the spine constraints. The total external force is denoted by $\boldsymbol{F}_{ext}(\boldsymbol{p}_{ij})$:

$$\boldsymbol{F}_{ext}(\boldsymbol{p}_{ij}) = \boldsymbol{F}_{grav}(\boldsymbol{p}_{ij}) + \boldsymbol{F}_{user}(\boldsymbol{p}_{ij}) + \boldsymbol{F}_{spine}(\boldsymbol{p}_{ij}). \quad (1)$$

Two kinds of internal forces act on the particle: a damping force and the force exerted by all the springs attached to it. The internal force is designated by $\boldsymbol{F}_{int}(\boldsymbol{p}_{ij})$:

$$\boldsymbol{F}_{int}(\boldsymbol{p}_{ij}) = \boldsymbol{D}(\boldsymbol{p}_{ij}) + \boldsymbol{F}_{spring}(\boldsymbol{p}_{ij}). \quad (2)$$

The damping force $\boldsymbol{D}(\boldsymbol{p}_{ij})$ is calculated from the damping constant and the node's velocity, while the spring force $\boldsymbol{F}_{spring}(\boldsymbol{p}_{ij})$ is computed from the displacements of the node and those nodes to which it is attached, and each spring's stiffness constant.

The sum of the external and internal forces on the particle is

$$\boldsymbol{F}_{ij} = \boldsymbol{F}_{ext}(\boldsymbol{p}_{ij}) + \boldsymbol{F}_{int}(\boldsymbol{p}_{ij}). \quad (3)$$

The value of this at time h is used to determine the particle's position at time $h + \Delta h$. We do this operation on each particle in the grid, so $O(n^2)$ operations are involved.

4.1 Basic mass-spring model

In the simplest type of mass-spring model, developed by Provot (1995), nodes are connected with three types of mass-less springs as shown in Figure 6: stretch, shear and bend springs.

1. Stretch springs connect $\boldsymbol{p}_{ij}$ with $\boldsymbol{p}_{i\pm1,j}$ and $\boldsymbol{p}_{i,j\pm1}$.
2. Shear springs connect $\boldsymbol{p}_{ij}$ with $\boldsymbol{p}_{i\pm1,j\pm1}$.
3. Bend springs connect $\boldsymbol{p}_{ij}$ with $\boldsymbol{p}_{i\pm2,j}$, $\boldsymbol{p}_{i,j\pm2}$ and $\boldsymbol{p}_{i\pm2,j\pm2}$.

Springs produce a force that depends on their length and stiffness constant. The stiffness is the same for all springs of a given type.

These springs are responsible for simulating the paper's stretch, shear and bend resistance. Stretch springs define the resistance of the material to elongation or compression in the horizontal and vertical directions. Shear springs define the resistance in diagonal directions. When the paper is bent, particle $\boldsymbol{p}_{ij}$ moves closer to its next-nearest-neighbor particles, and the paper's bend resistance is calculated from the distance of $\boldsymbol{p}_{ij}$ to its eight next-nearest-neighbors.

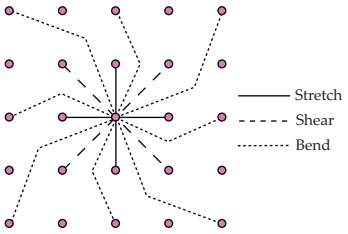


Fig. 6. Simple mass-spring model

4.1.1 Effect of forces

We denote the stiffness constant of the spring linking particles $\boldsymbol{p}_{ij}$ and $\boldsymbol{p}_{kl}$ by $K_{ij,kl}$, and the rest length of the spring, which is its initial length at time $h = 0$, by $\boldsymbol{l}^0_{ij,kl}$. The length of the spring at time h is $\boldsymbol{l}_{ij,kl} = \boldsymbol{p}_{kl} - \boldsymbol{p}_{ij}$. According to Hooke's Law, the total spring force on $\boldsymbol{p}_{ij}$ is given by

$$\boldsymbol{F}_{spring}(\boldsymbol{p}_{ij}) = \sum_{(k,l)\in R} -K_{ij,kl}(|\boldsymbol{l}_{ij,kl}| - |\boldsymbol{l}^0_{ij,kl}|)\frac{\boldsymbol{l}_{ij,kl}}{|\boldsymbol{l}_{ij,kl}|} \tag{4}$$

where the sum is over R, the set of nodes that are connected with $\boldsymbol{p}_{ij}$.

The damping force for node $\boldsymbol{p}_{ij}$ is

$$\boldsymbol{D}(\boldsymbol{p}_{ij}) = -K_d\dot{\boldsymbol{p}}_{ij}, \tag{5}$$

where K_d is the damping coefficient of the paper and $\dot{\boldsymbol{p}}_{ij}$ is the velocity of $\boldsymbol{p}_{ij}$.

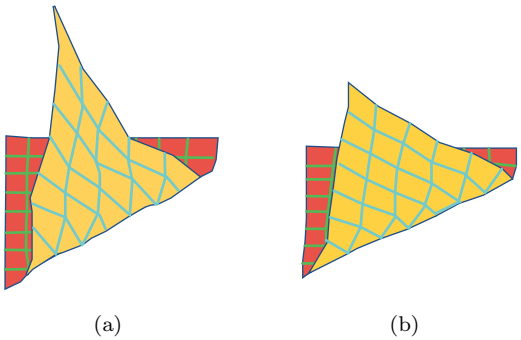
4.1.2 Limit strain

The elasticity of real materials is non-linear. Once a certain degree of elongation has been reached, they become very stiff extremely rapidly when stretched further. If the applied load is high the material will rupture; however, we will not attempt to model ruptures. Figure 7(a) shows the appearance of a piece of paper that has been stretched unreasonably, a problem that we will proceed to fix.

We define the amount of elongation as

$$\tau = \frac{|\boldsymbol{l}_{ij,kl}| - |\boldsymbol{l}^0_{ij,kl}|}{|\boldsymbol{l}^0_{ij,kl}|} \tag{6}$$

and limit it to a predetermined constant τ_c (normally set to 10%). If the elongation threatens to exceed τ_c the nodes are brought closer together to bring it back to this value. This is done by moving the both particles towards their mid-point if they are both loose, while if one is fixed—e.g., if it lies on to the book's spine—the other is moved closer to it. The result is shown in Figure 7(b), where each spring has been limited to a predefined maximum elongation.



(a) (b)

Fig. 7. The appearance of the page: (a) without limit straining (b) with limit straining.

4.1.3 Integrating over time

Given each node's position $\boldsymbol{p}_{ij}$, velocity $\dot{\boldsymbol{p}}_{ij}$ and total force $\boldsymbol{F}_{ij}$ at time h, our goal is to determine the position $\boldsymbol{p}'_{ij}$ and velocity $\dot{\boldsymbol{p}}'_{ij}$ at time $h + \Delta h$. According to Newton's law, the acceleration at time $h + \Delta h$ is

$$\ddot{\boldsymbol{p}}'_{\boldsymbol{ij}} = \frac{\boldsymbol{F}_{ij}}{m_{ij}}. \tag{7}$$

Using explicit Euler time integration,

$$\begin{aligned}\dot{\boldsymbol{p}}'_{ij} &= \dot{\boldsymbol{p}}_{ij} + \ddot{\boldsymbol{p}}'_{ij}\Delta h\\ \boldsymbol{p}'_{ij} &= \boldsymbol{p}_{ij} + \dot{\boldsymbol{p}}'_{ij}\Delta h.\end{aligned} \tag{8}$$

These equations are used to update the nodes' position and velocity at every time step.

4.2 Bending

We now describe a more advanced mass-spring model developed by Choi and Ko (2002) that can be bent into smooth curves. Paper is strongly resistant to stretching or shearing, but has little initial resistance to compression. However, when a certain degree of compression has been reached, it rapidly becomes stiffer under further compression. To simulate this a more sophisticated damping, bending and compression model is needed.

Figure 8(a) shows how unnatural bending can look with the previous model. The page behaves in a manner more akin to cloth than paper, with insufficient resistance to the bending force—it crumples rather than bending stiffly and smoothly as paper does. In contrast Figure 8(b) shows a smoothly contoured bend that was produced by the model we develop in this section.

While it would be perfectly possible to use the same explicit time integration that was developed in the last section for this new model, it would be much slower. In order to make the system respond quickly to the user's interaction, semi-implicit time integration is used instead. This is more stable than the explicit integration procedure, so large time steps can be used to reach the equilibrium state quickly.

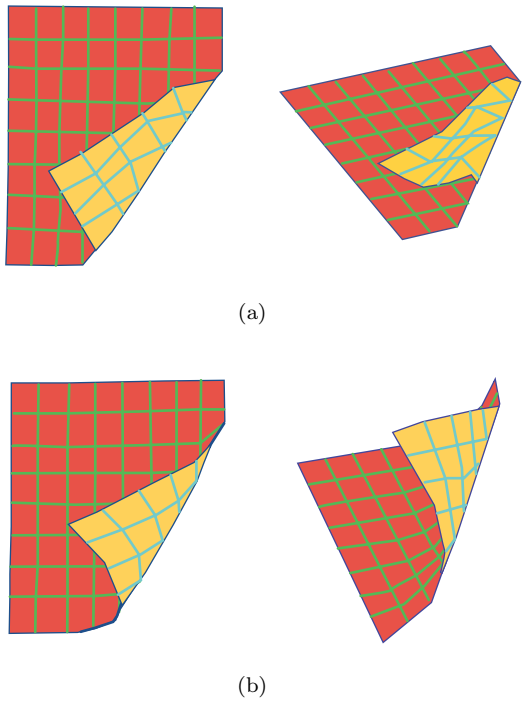


Fig. 8. The appearance of the page: (a) using the simple mass-spring model of Section 4.1; (b) using the improved model of Section 4.2

Each particle is connected with others in the mesh in the same manner as before, and again all springs of the same type have the same stiffness constant. However, different formulae are used to calculate the spring forces. Stretch and shear springs only affect the paper's resistance to elongation in the vertical, horizontal and diagonal directions, while bend springs affect its resistance to bending and compression in these directions.

4.2.1 Effect of forces

Let $K_{ij,kl}$ denote the stiffness constant of a spring connecting $\boldsymbol{p}_{ij}$ to $\boldsymbol{p}_{kl}$, and $\boldsymbol{l}^0_{ij,kl}$ the spring's initial length at time $h = 0$. The length of the spring at time h is $\boldsymbol{l}_{ij,kl} = \boldsymbol{p}_{kl} - \boldsymbol{p}_{ij}$. The total spring force on $\boldsymbol{p}_{ij}$ is given by

$$\boldsymbol{F}_{spring}(\boldsymbol{p}_{ij}) = \sum_{(k,l)\in R_s} \boldsymbol{s}_{ij,kl} + \sum_{(k,l)\in R_b} \boldsymbol{b}_{ij,kl} \qquad (9)$$

where the sums are over R_s, the set of nodes connected to $\boldsymbol{p}_{ij}$ by stretch and shear springs, and R_b, the set of particles connected to $\boldsymbol{p}_{ij}$ by bend springs.

The force exerted by the stretch and shear springs is the same as before (equation (4)) for elongation but zero for compression:

$$\boldsymbol{s}_{ij,kl} = \begin{cases} -K_{ij,kl}(|\boldsymbol{l}_{ij,kl}| - |\boldsymbol{l}^0_{ij,kl}|)\frac{\boldsymbol{l}_{ij,kl}}{|\boldsymbol{l}_{ij,kl}|} & : |\boldsymbol{l}_{ij,kl}| \geq |\boldsymbol{l}^0_{ij,kl}| \\ 0 & : |\boldsymbol{l}_{ij,kl}| < |\boldsymbol{l}^0_{ij,kl}| \end{cases} \qquad (10)$$

The force exerted by the bend springs is

$$\boldsymbol{b}_{ij,kl} = \begin{cases} 0 & : |\boldsymbol{l}_{ij,kl}| \geq |\boldsymbol{l}^0_{ij,kl}| \\ -K_{ij,kl} f_b(\frac{|\boldsymbol{l}_{ij,kl}|}{|\boldsymbol{l}^0_{ij,kl}|})\frac{\boldsymbol{l}_{ij,kl}}{|\boldsymbol{l}_{ij,kl}|} & : |\boldsymbol{l}_{ij,kl}| < |\boldsymbol{l}^0_{ij,kl}| \end{cases} \qquad (11)$$

This equation models the non-linear nature of bend resistance. The shape of buckled paper is close to a circular arc. We approximate the arc's length by the spring's rest length $|\boldsymbol{l}^0_{ij,kl}|$ and express the curvature as a function of the distance between the particles. Choi and Ko (2002) approximated the curvature function f_b as a fourth-order polynomial,

$$f_b(s) = -a_4 s^4 + a_3 s^3 - a_2 s^2 + a_1 s - a_0$$

where positive numeric values for the coefficients a_4, a_3, a_2, a_1 and a_0 were derived from the physical deformation constants that characterize the material in question.

In the previous model we used the same damping constant for stretch, shear and bend (equation (5)). However, paper resists stretching and shearing much more than it resists bending, and this requires higher values for the these two damping constants than for the bend damping constant. This was not done in the previous model because bending looked unrealistic anyway. Now we would like to reduce the bend damping constant—but this causes unrealistic in-plane oscillations. To inhibit these requires a more sophisticated damping model than the one developed in the previous section. If $K_d ij, kl$ is the damping constant of the spring linking particles $\boldsymbol{p}_{ij}$ and $\boldsymbol{p}_{kl}$, the total damping force on $\boldsymbol{p}_{ij}$ is

$$\boldsymbol{D}(\boldsymbol{p}_{ij}) = -\sum_{(k,l)\in R} K_d(\dot{\boldsymbol{p}}_{ij} - \dot{\boldsymbol{p}}_{kl}) \qquad (12)$$

where R is the list of nodes that are connected to $\boldsymbol{p}_{ij}$ by any type of spring.

4.2.2 Integrating over time

In this model, semi-implicit time integration with a second-order backward difference formula is used to calculate the new position of particle $\boldsymbol{p}_{ij}$ with mass m_{ij}. Let $\boldsymbol{P}_{ij}$ and $\dot{\boldsymbol{p}}_{ij}$ be the position and velocity at time $h - \Delta h$, and $\boldsymbol{p}'_{ij}$ and $\dot{\boldsymbol{p}}'_{ij}$ the position and velocity at time h. Given this information, the new position $\boldsymbol{p}''_{ij}$ and velocity $\dot{\boldsymbol{p}}''_{ij}$ at time $h + \Delta h$ can be calculated as follows:

$$\begin{aligned} \Delta\boldsymbol{p}''_{ij} &= \frac{1}{\Delta h}\left(\frac{3}{2}\boldsymbol{p}''_{ij} - 2\boldsymbol{p}'_{ij} + \frac{1}{2}\boldsymbol{p}_{ij}\right) \\ \Delta\dot{\boldsymbol{p}}''_{ij} &= \frac{1}{\Delta h}\left(\frac{3}{2}\dot{\boldsymbol{p}}''_{ij} - 2\dot{\boldsymbol{p}}'_{ij} + \frac{1}{2}\dot{\boldsymbol{p}}_{ij}\right) \end{aligned} \qquad (13)$$

These quantities must also satisfy the equation

$$\begin{pmatrix} \Delta\boldsymbol{p}''_{ij} \\ \Delta\dot{\boldsymbol{p}}''_{ij} \end{pmatrix} = \begin{pmatrix} \dot{\boldsymbol{p}}''_{ij} \\ \frac{\boldsymbol{F}''_{ij}}{m_{ij}} \end{pmatrix} \qquad (14)$$

and

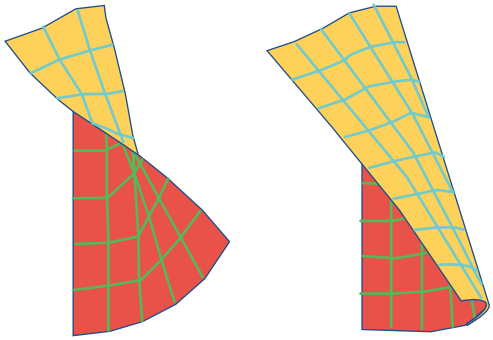
$$\boldsymbol{F}''_{ij} = \boldsymbol{F}'_{ij} + \sum_{(k,l)\in R} \left(\frac{\partial \boldsymbol{F}''_{ij}}{\partial \boldsymbol{p}''_{kl}} \Delta \boldsymbol{p}''_{ij} + \frac{\partial \boldsymbol{F}''_{ij}}{\partial \dot{\boldsymbol{p}}''_{kl}} \Delta \dot{\boldsymbol{p}}''_{ij}\right). \quad (15)$$

where R is the list of nodes that are connected to $\boldsymbol{p}_{ij}$ by any type of spring.

4.3 Bending without twisting

Previous sections modeled the resistance of the paper to bending using springs that connect each particle with its next-nearest-neighbors. This model is simplistic because these springs are insensitive changes in the dihedral angles between faces. We now take this into account by changing the way the bending force is calculated.

The effect can be seen in Figure 9, which shows a page using the model in the previous section and the model that will be developed in this section. Clearly, the latter looks more realistic, because the twist in the edge nearest to the viewer has been eliminated.



(a) (b)

Fig. 9. The appearance of the page: (a) using model of Section 4.2; (b) using improved bending model of Section 4.3

In this model, the paper is discretized into a set of right-angled triangles, as shown in Figure 10. Initially the paper is flat, and $z = 0$ for each particle. In order to calculate the internal forces on the triangles instead of on each particle, slightly different formulae are used to determine the material's resistance to stretching, shearing and bending.

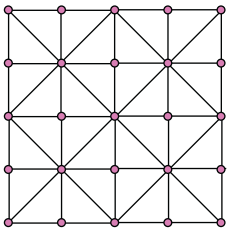


Fig. 10. Mesh representation

4.3.1 Effect of forces

The improved bending model is better expressed in terms of energies rather than forces as we did above; we can calculate the forces as the spatial derivative of energy. As before, three types of energy—stretch, shear and bend—are used to characterize the difference between the undeformed configuration and the deformed one. The internal energy working on each particle $\boldsymbol{p}_{ij}$ is

$$\boldsymbol{E}_{spring} = \sum_{x\in T} \boldsymbol{E}_{stretch}(x) + \sum_{x\in L} \boldsymbol{E}_{shear}(x) + \sum_{x\in B} \boldsymbol{E}_{bend}(x)$$

where T is the set of all triangles containing particle $\boldsymbol{p}_{ij}$, L is the set of all edges that are connected to particle $\boldsymbol{p}_{ij}$, and B is the set of all edges that are connected to particle $\boldsymbol{p}_{ij}$ and shared by two triangles. The stretch component relates to the triangle's area, the shear component to the length of its edges, and the bend component to the dihedral angle with adjacent triangles; these three components are discussed further below. The total internal force at particle $\boldsymbol{p}_{ij}$ is obtained by differentiating the energy:

$$\boldsymbol{F}_{spring}(\boldsymbol{p}_{ij}) = \frac{\partial \boldsymbol{E}_{spring}}{\partial \boldsymbol{p}_{ij}}.$$

Stretch energy is produced by the triangle's resistance to stretching or compression, which corresponds to changes in the triangle's area. Let A^0 be its initial area ($h = 0$) and A^h be its area at time h. The stretch energy for the triangle is

$$\boldsymbol{E}_{stretch}(x) = -K_{stretch} A^0 (1 - \frac{A^h}{A^0})^2, \quad (16)$$

where $K_{stretch}$ is the stretch stiffness constant.

Shear energy is produced by the triangle's resistance to shearing, which corresponds to changing the length of its edges but not its area. The shear energy for the triangle is the sum of the shear energies for each of its edges. Consider the edge e. Let $\boldsymbol{l}^0$ be its initial length (at time $h = 0$) and $\boldsymbol{l}^h$ its length at time h. The shear energy that works on the edge is

$$\boldsymbol{E}_{shear}(x) = -K_{shear} |\boldsymbol{l}^0| (1 - \frac{|\boldsymbol{l}^h|}{|\boldsymbol{l}^0|})^2 \quad (17)$$

where K_{shear} is the shear stiffness constant.

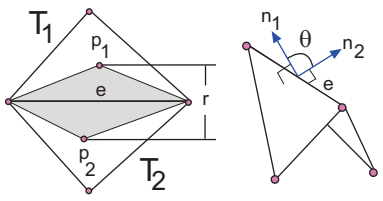


Fig. 11. Components to calculate the mean curvature

Bending energy is produced by altering the mean curvature between two triangles that share the same edge. The bending energy for the triangle is the sum of the bending energies for each edge that borders a neighboring triangle. Figure 11 shows two triangles T_1 and T_2 that share a common edge e. Their initial (flat) position is shown unshaded; the shaded triangles are leaning upwards, towards the viewer. The angle between the triangles at time h is

$$\theta^h = \tan^{-1}\left(\frac{\sin\theta^h}{\cos\theta^h}\right) \tag{18}$$

The sin and cosine can be expressed in terms of $\boldsymbol{n}_1^h$ and $\boldsymbol{n}_2^h$, the unit normal vectors to T_1 and T_2 at time h, and the length $\boldsymbol{l}^h$ of the edge e at time h:

$$\sin\theta^h = (\boldsymbol{n}_1^h \times \boldsymbol{n}_2^h) \cdot \frac{\boldsymbol{l}^h}{|\boldsymbol{l}^h|}$$
$$\cos\theta^h = \boldsymbol{n}_1^h \cdot \boldsymbol{n}_2^h$$

Denote the heights of triangles T_1 and T_2, measured from the shared edge e to the opposite vertex, by t_1 and t_2. The initial curvature span is the distance between the two triangles' centers of gravity in the initial position, which is $r = \frac{1}{6}(t_1 + t_2)$. The bending energy of edge e is

$$\boldsymbol{E}_{bend} = -K_{bend}\frac{|\boldsymbol{l}^0|}{r}(\theta^h - \theta^0)^2 \tag{19}$$

We use the same formulae as before, equation (12) above, to calculate the damping force exerted on particle $\boldsymbol{p}_{ij}$ through its interaction with particle $\boldsymbol{p}_{kl}$.

4.3.2 Integrating over time

Newmark implicit time integration is used for this model. This method gives the new position and velocity of particle $\boldsymbol{p}_{ij}$ at time $h + \Delta h$ as

$$\boldsymbol{p}'_{ij} = \boldsymbol{p}_{ij} + \Delta h\dot{\boldsymbol{p}}_{ij} + \frac{\Delta h^2}{4}(\ddot{\boldsymbol{p}}_{ij} + \ddot{\boldsymbol{p}}'_{ij})$$
$$\dot{\boldsymbol{p}}'_{ij} = \dot{\boldsymbol{p}}_{ij} + \frac{\Delta h}{2}(\ddot{\boldsymbol{p}}_{ij} + \ddot{\boldsymbol{p}}'_{ij})$$

The acceleration at time $h + \Delta h$ is given by

$$\ddot{\boldsymbol{p}}'_{ij} = \frac{\boldsymbol{F}_{ij}}{m_{ij}} \tag{20}$$

where m_{ij} is $\boldsymbol{p}_{ij}$'s mass.

5 Physical simulation using the finite element method

The finite element method is a general technique for simulating any physical material. The material is divided into a regular grid of small elements in three-dimensional space whose size determines the accuracy of the simulation. When simulating page turning we can take advantage of the quasi-two-dimensional nature of paper by using a two-dimensional rectangular mesh rather than a full three-dimensional grid, dividing a sheet of paper into $m \times n$ small patches. However, in the finite-element method patches are not modeled as particles as in they were in the previous section: instead, their dimensions and shape are taken into account. The forces on a patch—including gravity, forces exerted by neighboring patches, and the force the reader applies to turn the page—produce stress which deforms the patch, and the finite element method calculates the amount of deformation. Note that although the grid is two-dimensional, the thickness of the paper is not ignored: each patch does have thickness, which varies across the patch.

Modeling paper as a thin shell, represented by a two-dimensional rectangular mesh of patches whose thickness varies, is a special case of the general finite element model of a three-dimensional solid, and is slightly easier to tackle than the general case. The first task is to establish a geometric mapping between the patches themselves and the global coordinate system (Section 5.1). Ultimately we need to consider the physical forces on each patch—the stress—and work out the distortion they cause—the strain. Strain is defined in terms of the spatial derivatives of a point's displacement, and calculating the stress-strain relationship involves differential equations. This requires a smooth, differentiable model of the patch; obtaining this is the second task (Section 5.2). It is necessary to model the patches not just as two-dimensional surfaces but as having thickness; the third task (Section 5.3). Then we need to establish a three-dimensional coordinate system at each reference point of the patches (Section 5.4), and express the displacement of the reference points in terms of these coordinates (Section 5.5).

In terms of local coordinates the relationship between stress (Section 5.6) and strain (Section 5.7) is simple: it is given by a deformation matrix (Section 5.8) whose components depend on the Young's modulus, Poisson's ratio, and shear correction factor of the material under consideration, and it is easy to obtain reasonable estimates of these for paper. The relationship, which involves first derivatives of displacement, must be mapped back into global coordinates in order for it to be solved. We need to consider all the forces on each element, and finally integrate the differential equations that have been obtained (Section 5.9).

5.1 Local and global coordinates

Figure 13 shows a general element. It is defined by eight reference points, four representing its corners and four representing the mid-points of each side. In global (x, y, z) coordinates these reference points are $\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3$ and $\boldsymbol{p}_4$

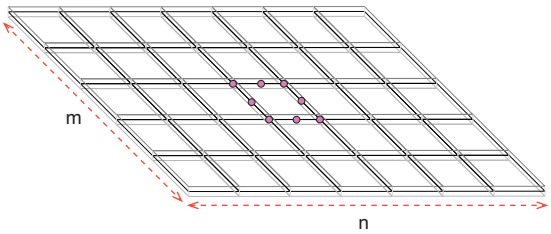


Fig. 12. Dividing a sheet of paper into $m \times n$ elements

for the first group and $\boldsymbol{p}_5, \boldsymbol{p}_6, \boldsymbol{p}_7$ and $\boldsymbol{p}_8$ for the second: each $\boldsymbol{p}_i$ has coordinates (x_i, y_i, z_i) in ordinary three-dimensional space. If thickness is modeled, the reference points are taken to be at the centre of the material and further parameters $t_1 \ldots t_8$ give the corresponding thicknesses, so that at each reference point the material of the paper extends from $-\frac{t_i}{2}$ to $+\frac{t_i}{2}$. The direction of this extension is normal to the paper at point $\boldsymbol{p}_i$, a direction that we call $\boldsymbol{n}_i$.

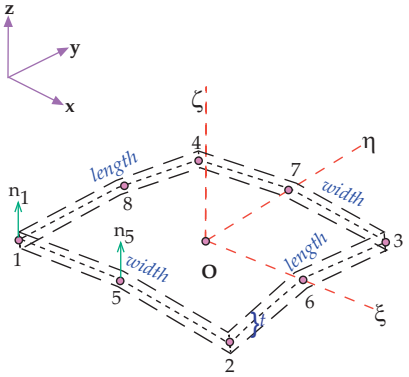


Fig. 13. An element of the shell

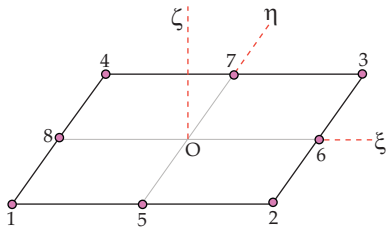


Fig. 14. Midsurface of the shell in local coordinate

Figure 14 shows the local coordinate system that is defined for each patch. Any point within the element is given by coordinates (ξ, η, ζ), where all coordinates lie between -1 and $+1$. ξ and η give the extent in directions corresponding to the width and height of the paper, respectively; while ζ is the direction perpendicular to the paper, and corresponds to its thickness. Reference point $\boldsymbol{p}_1$, which lies in the centre of the paper, has local coordinates $\boldsymbol{\pi}_1 = (-1, -1, 0)$, while point $\boldsymbol{p}_3$ has local coordinates $\boldsymbol{\pi}_3 = (1, 1, 0)$. The upper surface of the paper at $\boldsymbol{p}_3$ is $(1, 1, 1)$, while the lower surface is $(1, 1, -1)$. We use the Greek symbol $\boldsymbol{\pi}$ to denote the local (ξ, η, ζ) coordinates of the point whose global (x, y, z) coordinates are $\boldsymbol{p}$, for the element in question. Table 1 shows the local coordinates of the eight reference points.

i	(ξ_i , η_i , ζ_i)
1	(−1 , −1 , 0)
2	(1 , −1 , 0)
3	(1 , 1 , 0)
4	(−1 , 1 , 0)

i	(ξ_i , η_i , ζ_i)
5	(0 , −1 , 0)
6	(1 , 0 , 0)
7	(0 , 1 , 0)
8	(−1 , 0 , 0)

Each patch is fully defined by the position in space of its eight reference points, and the coherence of the assembly of patches is ensured by making the three points that define one side of a patch correspond to the three points that define the adjacent side of the neighboring patch in the obvious manner, all the way along both the height and the width of the paper.

5.2 Defining a smooth surface

We next address the problem of how to define from this grid of reference points a smooth and continuous surface for each patch. In fact, it is not necessary to fit a smooth surface in order to present the patch on the computer screen: for that purpose the discrete grid of reference points is entirely adequate. Rather, a smooth surface is needed because the analysis in subsequent subsections in terms of stress and strain involves spatial derivatives of the surface.

We restrict attention to a particular patch. Given a point on the midsurface of this patch whose local coordinates are $\boldsymbol{\pi} = (\xi, \eta, 0)$ ($\zeta = 0$ because it is on the midsurface), what is its global coordinate vector $\boldsymbol{p} = (x, y, z)$? The vector $\boldsymbol{p}$ will be expressed as a weighted sum of the vectors $\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_8$ that represent the 8 reference points of the patch, where the weights $w_1(\xi, \eta), w_2(\xi, \eta), \ldots, w_8(\xi, \eta)$ reflect the position of the point $\boldsymbol{\pi}$ in local coordinate space:

$$\boldsymbol{p} = \sum_{i=1}^{8} w_i(\xi, \eta)\boldsymbol{p}_i \tag{21}$$

For example, if $\boldsymbol{\pi}$ corresponds with one of the reference points, the weights will be 1 for that reference point and 0 for the other seven points.

If the patch were planar this would be a simple matter of linear interpolation. However, in general the points $\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_8$ do not lie on a plane in 3-space. To account for this, it is necessary to define eight non-linear interpolation functions $w_i(\xi, \eta)$ with the following properties:

1. $\sum_{i=1}^{8} w_i(\xi, \eta) = 1$ for any values ξ, $\eta \in [-1, 1]$

2. $w_i(\xi_i, \eta_i) = 1$ for each reference point, $i = 1, 2, \ldots, 8$
3. $w_i(\xi_j, \eta_j) = 0$ if $j \neq i$.

It turns out that while linear interpolation is inadequate to model a non-planar patch, quadratic weighting functions can be constructed that satisfy these properties. Figure 15(a) shows a suitable function $w_1(\xi, \eta)$ for reference point 1 of the patch: as can be seen, it satisfies properties 2 and 3 with $i = 1$. We construct this function by taking the component shown in Figure 15(b) and subtracting the components in Figure 15(c) and 15(d). Figure 15(b) is a bilinear surface that satisfies property 2 at reference point 1, and satisfies property 3 at points 2, 3, 4, 6 and 7 but but violates it at points 5 and 8; Figure 15(c) is a parabolic correction that restores property 3 at point 5; and Figure 15(d) does the same for point 8.

The bilinear function in Figure 15(b) has equation

$$\frac{1}{4}(1-\xi)(1-\eta).$$

This alters the function's value for reference point $\boldsymbol{\pi}_1 = (-1, -1, 0)$ as desired, and evaluates to 0 at any reference points with either $\xi = 1$ or $\eta = 1$—that is, points $\boldsymbol{\pi}_2, \boldsymbol{\pi}_6, \boldsymbol{\pi}_3, \boldsymbol{\pi}_7, \boldsymbol{\pi}_4$. It also has the undesired effect of evaluating to 0.5 at points $\boldsymbol{\pi}_5$ and $\boldsymbol{\pi}_8$, as shown. The parabolic functions in Figures 15(c) and 15(d) have equations

$$\frac{1}{4}(1-\xi^2)(1-\eta) \quad \text{and} \quad \frac{1}{4}(1-\xi)(1-\eta^2).$$

These both evaluate to 0 at points $\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \boldsymbol{\pi}_6, \boldsymbol{\pi}_3, \boldsymbol{\pi}_7$ and $\boldsymbol{\pi}_4$, and 0.5 at $\boldsymbol{\pi}_5$ and $\boldsymbol{\pi}_8$ respectively. Thus the overall effect of raising to 1 the weighting function's value at reference point $\boldsymbol{\pi}_1$, shown in Figure 15(a), is accomplished by,

$$\begin{aligned} w_1(\xi, \eta) &= \mathit{bilinear} - \mathit{parabolic}_1 - \mathit{parabolic}_2 \\ &= \frac{1}{4}(1-\xi)(1-\eta)(-1-\xi-\eta) \end{aligned}$$

Weighting functions for the other three corners $\boldsymbol{\pi}_2, \boldsymbol{\pi}_3$, and $\boldsymbol{\pi}_4$ are accomplished in like manner.

The weighting function for the mid-point of a side is simpler, requiring only a single parabolic model as shown in Figure 16 for $\boldsymbol{\pi}_5$.

In summary, we have established weighting functions that have the value of 1 at a reference point and 0 at the other seven reference points. These functions are

$$w_i(\xi, \eta) = \begin{cases} \frac{1}{4}(1+\xi\xi_i)(1+\eta\eta_i)(-1+\xi\xi_i+\eta\eta_i) & i \in [1,4] \\ \frac{1}{2}(1-\xi^2)(1+\eta\eta_i) & i \in \{5,7\} \\ \frac{1}{2}(1+\xi\xi_i)(1-\eta^2) & i \in \{6,8\} \end{cases} \tag{22}$$

Not only do they satisfy properties 2 and 3 above, but it can be shown algebraically that the eight functions sum to 1 for any values of ξ, η, thus satisfying property 1 too. Furthermore, the values of all eight interpolation functions along any edge are defined purely by the three reference points that constitute that edge; thus (x, y, z) coordinates of any point along an edge will be the same as for the neighboring patch, making the entire surface continuous. However, the first derivative across an edge is *not* continuous, which means that creases in the pa-

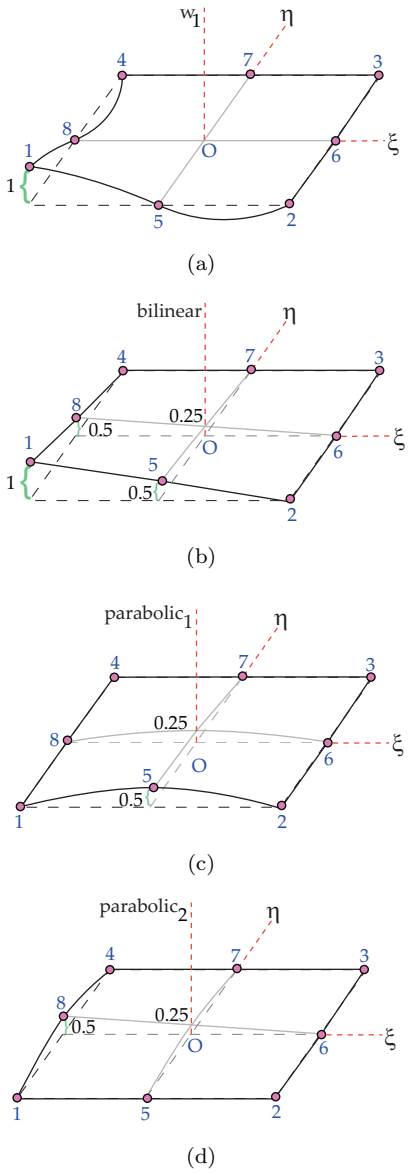


Fig. 15. Creating (a) the interpolation function w_1 by adding three functions: (b) *bilinear*, (c) $\mathit{parabolic}_1$, and (d) $\mathit{parabolic}_2$

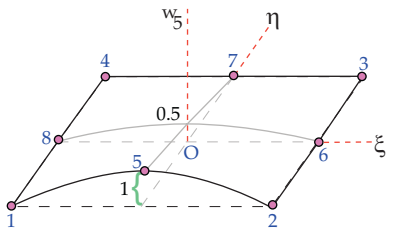


Fig. 16. Interpolation function w_5

per such as dogeared pages—or even a crumpled ball of paper—can be modeled.

5.3 Inside the shell

The discussion so far has focused on points on the midsurface of the shell. We will also need to deal with general points inside the shell, $\boldsymbol{\pi} = (\xi, \eta, \zeta)$ with $\zeta \neq 0$. For this we need to establish vectors normal to the shell at each of the reference points, as shown in Figure 17.

In the beginning, each element of the shell is represented by the coordinates of the top and bottom surfaces at each of the eight reference positions, rather than the reference points themselves. Then, at each reference position, the actual reference point is calculated by averaging the coordinates of the top and bottom surfaces, and the direction of the normal vector is determined by taking the difference between the top and bottom points and normalizing to a unit vector, as illustrated in Figure 17. As the configuration of the shell evolves in time the direction of each of the eight normal vectors is updated as described in the next subsection.

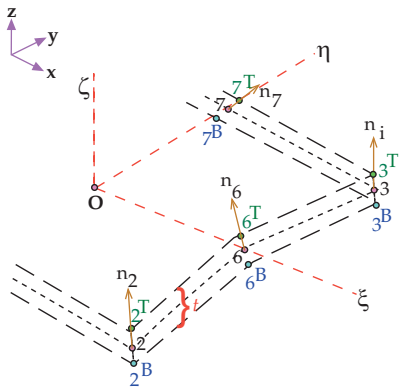


Fig. 17. Normal vector of the shell's midsurface

Given the normal vector, an arbitrary point $\boldsymbol{\pi} = (\xi, \eta, \zeta)$ inside the shell is transformed to $\boldsymbol{p}$ in (x, y, z) coordinate as follows:

$$\boldsymbol{p} = \sum_{i=1}^{8} w_i(\xi, \eta) \left(\boldsymbol{p}_i + \zeta \frac{t_i}{2} \boldsymbol{n}_i \right) \tag{23}$$

The first component inside the bracket, $\boldsymbol{p}_i$, corresponds to equation (21) above representing the midsurface of the shell. The second component, involving ζ, translates the displacement along the ζ axis back into (x, y, z) coordinates.

5.4 The normal-vector coordinate system

The normal vectors will play a crucial part in our model. For example, in Section 5.6 we assume that the material of the paper is completely flexible and offers no resistance to external forces in the direction perpendicular to it. Because of this, and to assist in updating the normal vectors from one configuration of the paper to the next, we establish a coordinate system at each reference point based on the direction of the normal at that point.

For this normal-vector coordinate system at reference point i we use the notation $\hat{x}_i, \hat{y}_i, \hat{z}_i$. The new z axis is in the direction of the normal to the surface at that reference point, $\boldsymbol{n}_i$. The new x axis is in a direction perpendicular to both this new z axis and the *old* y axis, that is, the y axis of the original global x, y, z coordinates. This direction is $\boldsymbol{e}^y \times \boldsymbol{e}_i^{\hat{z}}$, where $\boldsymbol{e}^y$ is a unit vector along the global y axis and $\boldsymbol{e}_i^{\hat{z}}$ is a unit vector along the new z axis, which is in the same direction as $\boldsymbol{n}_i$. (A degenerate case arises when the normal $\boldsymbol{n}_i$ coincides with the direction of the global y axis, in which case we set the new x axis to be in the direction of the old z axis.) Having established the new z and x axes, the new y axis is set to be perpendicular to them both, that is, in the direction $\boldsymbol{e}_i^{\hat{z}} \times \boldsymbol{e}_i^{\hat{x}}$.

We now consider how to update the normal vector $\boldsymbol{n}_i$ from one moment to the next. We assume that the new normal is the result of rotating $\boldsymbol{n}_i$ through an angle α_i around the $\hat{x}_i$ axis and an angle β_i around the $\hat{y}_i$ axis. Then the new normal vector is

$$\begin{aligned} \boldsymbol{n}_i' &= \boldsymbol{e}_i^{\hat{x}} \sin\beta_i + \boldsymbol{e}_i^{\hat{y}} \cos\beta_i \sin(-\alpha_i) + \boldsymbol{e}_i^{\hat{z}} \cos\beta_i \cos(-\alpha_i) \\ &= \beta_i \boldsymbol{e}_i^{\hat{x}} - \alpha_i \boldsymbol{e}_i^{\hat{y}} + \boldsymbol{e}_i^{\hat{z}} \end{aligned} \tag{24}$$

assuming that the rotations are small. $\boldsymbol{n}_i'$ will be the $\hat{z}$ direction of the normal-vector coordinate system that is used for the next step.

The angles α_i and β_i are central to the updating procedure, and will figure prominently in what follows.

5.5 Displacement

If the position of point $\boldsymbol{p}$ at time h is $\boldsymbol{p}$, and at time $h + \Delta h$ is $\boldsymbol{p}'$, its displacement from time h to time $h + \Delta h$ in the x, y and z directions is

$$\boldsymbol{d} = \boldsymbol{p}' - \boldsymbol{p}$$

by substituting the value of $\boldsymbol{p}$ according to equation (23),

$$\boldsymbol{d} = \sum_{i=1}^{8} w_i(\xi, \eta) \left\{ (\boldsymbol{p}_i' - \boldsymbol{p}_i) + \zeta \frac{t_i}{2} (\boldsymbol{n}_i' - \boldsymbol{n}_i) \right\}$$

We write $\boldsymbol{d} = (u, v, w)$. The entire displacement of node i from time h to $h + \Delta h$ is characterized by the following 5-vector:

$$\boldsymbol{q}_i = (u_i, v_i, w_i, \alpha_i, \beta_i)$$

where u_i, v_i, and w_i are the displacement of $\boldsymbol{p}_i$ in the x, y and z axis directions respectively, and α_i and β_i are the rotation angles around the $\hat{x}$ and $\hat{y}$ axes.

Given the displacement vectors $\boldsymbol{q}_i$ for each of the nodes of a surface patch, we now calculate the displacement vector $\boldsymbol{d}$ for a general point on the patch at position (ξ, η, ζ).

$$\begin{aligned}\boldsymbol{d} &= \sum_{i=1}^{8} w_i(\xi,\eta)\left\{(\boldsymbol{p}_i' - \boldsymbol{p}_i) + \zeta\frac{t_i}{2}(\boldsymbol{n}_i' - \boldsymbol{n}_i)\right\} \\ &= \sum_{i=1}^{8} w_i(\xi,\eta)\left\{\begin{bmatrix} u_i \\ v_i \\ w_i \end{bmatrix} + \zeta\frac{t_i}{2}(\beta_i \boldsymbol{e}_i^{\hat{x}} - \alpha_i \boldsymbol{e}_i^{\hat{y}})\right\} \\ &= \sum_{i=1}^{8} \boldsymbol{w}_i \boldsymbol{q}_i \end{aligned} \tag{25}$$

where the interpolation matrix for reference point $\boldsymbol{p}_i$ is

$$\boldsymbol{w}_i = w_i(\xi,\eta)\begin{bmatrix} 1 & 0 & 0 & -\zeta\frac{t_i}{2}l_i^{\hat{y}} & \zeta\frac{t_i}{2}l_i^{\hat{x}} \\ 0 & 1 & 0 & -\zeta\frac{t_i}{2}m_i^{\hat{y}} & \zeta\frac{t_i}{2}m_i^{\hat{x}} \\ 0 & 0 & 1 & -\zeta\frac{t_i}{2}n_i^{\hat{y}} & \zeta\frac{t_i}{2}n_i^{\hat{x}} \end{bmatrix}$$

and $\boldsymbol{e}_i^{\hat{x}} = (l_i^{\hat{x}}, m_i^{\hat{x}}, n_i^{\hat{x}})$, $\boldsymbol{e}_i^{\hat{y}} = (l_i^{\hat{y}}, m_i^{\hat{y}}, n_i^{\hat{y}})$.

We have now completed our analysis of the geometry of the surface element or patch. To summarize: each patch is situated in three-dimensional space by the coordinates of its eight reference points $\boldsymbol{p}_i$. A local coordinate system is defined for the patch which allows any point within it to be defined by coordinates (ξ, η, ζ). The entire volume of the patch is swept out as these coordinates range from -1 to 1. The third coordinate corresponds to the thickness of the patch, and $\zeta = 0$ defines the midsurface. Eight of the nine combinations of the values -1, 0, 1 for each of ξ and ζ define the eight reference points $\boldsymbol{p}_i$ (the ninth, $(0, 0)$, is the centre of the patch.) Any point $\boldsymbol{\pi} = (\xi, \eta, \zeta)$ within the patch can be expressed in terms of its global coordinates $\boldsymbol{p} = (x, y, z)$ using the transformation of equation (23). This transformation also involves the normal vector $\boldsymbol{n}_i$ to the patch at each reference point.

The normal vector $\boldsymbol{n}_i$ is used to define an coordinate system at reference point i which will be deployed in the analysis below. As the patch moves from one configuration to the next under the influence of forces discussed in subsequent sections, it is necessary to update the coordinates of each reference point and the orientation of the normal vector at each one. The key to the updating procedure is the displacement vector, a five-dimensional vector $\boldsymbol{q}_i$ which includes the displacement of the reference point in the x, y, and z directions as well as the angular displacements around the two axes of the normal-vector coordinate system that are perpendicular to the normal at the reference point. Given any point within the patch expressed as (ξ, η, ζ) in local coordinates, and the five-dimensional displacement vectors $\boldsymbol{q}_i$ for each reference point, expression (25) shows how to calculate the corresponding displacement of the point in global (x, y, z) coordinates. These displacements occur as the result of the forces that we discuss next.

5.6 Stress

Stress ($\boldsymbol{\sigma}$) measures the distribution of the internal forces in the shell, and determines the material's resistance to deformation under the pressure of external forces ($\boldsymbol{b}$). The stress at any point within the shell—and here we are considering a general point $\boldsymbol{p}$ within a surface element, not a reference point $\boldsymbol{p}_i$—is defined as the force per unit area over a small area (ΔA) of the body. Formally,

$$\boldsymbol{\sigma} = \lim_{\Delta A \to 0} \frac{\Delta \boldsymbol{b}}{\Delta A}$$

Stress can be resolved into normal stress and shear stress. The former measures the force perpendicular to the surface, while the latter measures the forces that act parallel to the surface. For example, σ_x is a stress perpendicular to the yz plane, while σ_{xy} and σ_{xz} are stresses in the yz plane that are parallel to the y and z axes respectively. A total of six stresses act at every point within the shell, and we express them as a column vector $\boldsymbol{\sigma} = [\sigma_x, \sigma_y, \sigma_z, \sigma_{xy}, \sigma_{xz}, \sigma_{yz}]^T$.

The effect of stress is to deform the body under consideration. The amount of deformation is called *strain*, and is expressed as the change in length per unit of the original length. Each of the six stresses at a point (more precisely, at a vanishingly small area) cause a corresponding amount of strain, where strain is a six-dimensional vector that corresponds to the spatial derivatives of the point's displacement $\boldsymbol{d} = \boldsymbol{p}' - \boldsymbol{p}$ discussed earlier:

$$\boldsymbol{\epsilon} = \begin{bmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_z \\ \epsilon_{xy} \\ \epsilon_{xz} \\ \epsilon_{yz} \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \end{bmatrix} \boldsymbol{d}$$

The relationship between the degree of stress and the amount of strain it causes is given by Hooke's law of elasticity, which states that the amount by which a body is deformed—the strain—is directly proportional to the stress that cause the deformation. The stress-strain relation can be expressed by the equation

$$\boldsymbol{\sigma} = \boldsymbol{D}\boldsymbol{\epsilon}, \tag{26}$$

where $\boldsymbol{D}$ is the deformation matrix, discussed below. First, however, we consider the problem of calculating the strain itself.

5.7 Calculating the strain

The strain is obtained by taking partial derivatives of $\boldsymbol{d}$ with respect to the global coordinates x, y and z, as shown above. Equation (25) expresses the displacement

at any point within a patch in terms of its local (ξ, η, ζ) coordinates and the eight vectors q_i that characterize the displacement of each reference point of the patch. To take the derivatives of this equation with respect to ξ, η and ζ is a simple matter. In order to calculate the strain, partial derivatives with respect to x, y, and z must be expressed in terms of partial derivatives with respect to ξ, η and ζ.

This is done by applying the chain rule in the following form:

$$\begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{bmatrix} = \boldsymbol{J}^{-1} \begin{bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \\ \frac{\partial}{\partial \zeta} \end{bmatrix} \tag{27}$$

where $\boldsymbol{J}^{-1}$ is the inverse of the Jacobian matrix that relates local coordinate derivatives to global ones:

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix} \tag{28}$$

$\boldsymbol{J}$ can be determined by taking derivatives of equation (23) above, which gives an explicit transformation from local coordinates to global ones. The details can be found in the Appendix.

The upshot is that the strain $\boldsymbol{\epsilon}$ at any point can be obtained from its local coordinates ξ, η and ζ and the displacement vector $\boldsymbol{q}_i$ at each of the patch's eight reference points. The operation can be summarized in matrix form as

$$\boldsymbol{\epsilon} = \sum_{i=1}^{8} \boldsymbol{B}_i(\xi, \eta, \zeta)\boldsymbol{q}_i \tag{29}$$

where $\boldsymbol{B}_i(\xi, \eta, \zeta)$, the *strain-displacement matrix*, can be obtained explicitly using the above techniques, for each reference point i.

5.8 The deformation matrix

If a body is compressed it tends to expand sideways; on the other hand, if it is pulled the material contracts laterally. These effects are quantified by the deformation matrix $\boldsymbol{D}$ that characterizes the relationship between stress and strain expressed by equation (26).

The deformation matrix depends on three constants that characterize the material under consideration: Young's modulus of elasticity E, Poisson's ratio ν, and the shear correction factor K. Young's modulus represents the stiffness of the material. Poisson's ratio captures the relation between deformation in the direction of the stress and deformations in directions perpendicular to it (we assume that the material is isotropic).

The shear correction factor is used to model a shear strain that is not uniform throughout the material, and allows the strain to alter linearly across its thickness. In our simulations, we typically use values of $E = 1,370\ N\ cm^{-1}$ $\nu = 0.33$ and $K = \frac{5}{6}$ for paper with mass density of $80\ m^{-3}$ (Bathe, 1995).

We assume that the shell does not resist any deformation in the $\hat{z}$ direction, so that the stress normal to its surface is zero. In the normal-vector coordinate system, the deformation matrix at point $\boldsymbol{\pi} = (\xi, \eta, \zeta)$ is

$$\widehat{\boldsymbol{D}} = \begin{bmatrix} \frac{E}{1-\nu^2} & \frac{E\nu}{1-\nu^2} & 0 & 0 & 0 & 0 \\ \frac{E\nu}{1-\nu^2} & \frac{E}{1-\nu^2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{E}{2(1+\nu)} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{EK}{2(1+\nu)} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{EK}{2(1+\nu)} \end{bmatrix} \tag{30}$$

This matrix must be transformed to give a deformation matrix $\boldsymbol{D}$ in the global coordinate system. The elements of the transformation matrix $\boldsymbol{Q}$ are obtained from the direction cosines of the $\hat{x}, \hat{y}, \hat{z}$ coordinate axes at point $\boldsymbol{p}$ measured in the x, y, z coordinate directions. $\boldsymbol{Q}$ transforms the strain vector in local coordinates $[\epsilon_\xi, \epsilon_\eta, \epsilon_\zeta, \epsilon_{\xi\eta}, \epsilon_{\xi\zeta}, \epsilon_{\eta\zeta}]^T$ into the strain vector in global coordinates $[\epsilon_x, \epsilon_y, \epsilon_z, \epsilon_{xy}, \epsilon_{xz}, \epsilon_{yz}]^T$. Again, the details are relegated to the Appendix.

5.9 Effect of forces

The deformation of any element is determined by the external and internal forces that work on it. By using the principle of conservation of energy, we can translate the forces into the displacement of the element at time $h + \Delta h$. This section explains the principles, the details are given in the Appendix.

When external forces are applied to a body, their potential energy (δE_T) is converted into kinetic energy (δE_K), corresponding to internal forces of inertia and damping, and potential energy (δE_P), corresponding to stress. Conservation of energy is expressed by equating internal and external energy:

$$\delta E_K + \delta E_P = \delta E_T. \tag{31}$$

We begin with the external forces to which each element of the shell is subjected. There are three different types. *Body forces* ($\boldsymbol{F}_B$)—such as the force of gravity—apply throughout the element. *Surface forces* ($\boldsymbol{F}_S$) may be applied at the surface of the shell—for example, the upper and lower surfaces of the left edge of the paper are held rigidly in place by forces exerted by the book's spine. *Nodal forces* ($\boldsymbol{R}_N$) are applied at individual reference points, for example by the user's thumb and forefinger on the corner of the page as it is turned. These three examples—gravity, the spine, and the user's effort in turning the page—are the only external forces in our simulation, although the model permits more general body, surface, and nodal forces to be applied.

Potential energy equals force times displacement. Each force must be integrated appropriately. The body force is integrated with respect to the element's volume, yielding $\boldsymbol{R}_B$. The surface force is integrated with respect to its surface area, yielding $\boldsymbol{R}_S$. Both these forces, along with $\boldsymbol{R}_N$, are multiplied by the displacement to compute the energy.

The potential energy done by external forces is calculated by multiplying each vector by the displacement vector and summing:

$$\delta E_T = (\boldsymbol{\delta q})^T(\boldsymbol{R}_B + \boldsymbol{R}_S + \boldsymbol{R}_N)$$

Now we turn to the internal forces of inertia, damping, and stress. *Inertial force* is caused by the element's reluctance to accelerate or deaccelerate. In accordance with Newton's second law, it is given by $\rho \boldsymbol{w}\ddot{\boldsymbol{q}}'$, where ρ is the element's mass density. *Damping force* results from friction within the deforming material, and is proportional to the element's velocity: $\kappa \boldsymbol{w}\dot{\boldsymbol{q}}'$ where κ is the element's damping coefficient. *Stress* measures the material's resistance to deformation due to the external forces and the element's initial configuration, and can be written as $\boldsymbol{DBq}' + \boldsymbol{\sigma}_0$. All three effects must be integrated over the volume of the element. This gives

$$\delta E_K + \delta E_P = (\boldsymbol{\delta q})^T(\boldsymbol{M}\ddot{\boldsymbol{q}}' + \boldsymbol{C}\dot{\boldsymbol{q}}' + \boldsymbol{K}\boldsymbol{q}' + \boldsymbol{S}_0)$$

The matrices $\boldsymbol{M}, \boldsymbol{C}$ and $\boldsymbol{K}$ are the element's mass, damping and stiffness matrices, $\boldsymbol{S}_0$ is the integrated initial stress $\boldsymbol{\sigma}_0$. If the initial position of the paper is flat, $\boldsymbol{\sigma}_0$ can be omitted altogether. For a more general case, where the paper might already be bent, $\boldsymbol{\sigma}_0$ is not zero and is calculated in the same way as $\boldsymbol{\sigma}$.

The law of conservation of energy is expressed by equating δE_T with the sum of δE_K and δE_P. Cancelling $(\boldsymbol{\delta q})^T$ and moving $\boldsymbol{S}_0$ to the other side yields the second-order differential equation

$$\boldsymbol{M}\ddot{\boldsymbol{q}}' + \boldsymbol{C}\dot{\boldsymbol{q}}' + \boldsymbol{K}\boldsymbol{q}' = \boldsymbol{R}_B + \boldsymbol{R}_S + \boldsymbol{R}_N - \boldsymbol{S}_0 \quad (32)$$

This is the fundamental equation that must be solved in order to determine how the position of the element evolves under the applied forces.

We solve this equation using Newmark implicit time integration (Bathe, 1995). Given the position $\boldsymbol{q}$, velocity $\dot{\boldsymbol{q}}$ and acceleration $\ddot{\boldsymbol{q}}$ of the element at time h, the new values $\dot{\boldsymbol{q}}', \ddot{\boldsymbol{q}}'$ at time $h + \Delta h$ are

$$\ddot{\boldsymbol{q}}' = \frac{4}{\Delta h^2}(\boldsymbol{q}' - \boldsymbol{q}) - \frac{4}{\Delta h}\dot{\boldsymbol{q}} - \ddot{\boldsymbol{q}}$$
$$\dot{\boldsymbol{q}}' = \dot{\boldsymbol{q}} + \int_h^{h+\Delta h} \ddot{\boldsymbol{q}}\, dh \quad (33)$$

To determine the nodal displacement at time $h + \Delta h$ based only on terms at time h, we substitute the value of $\dot{\boldsymbol{q}}', \ddot{\boldsymbol{q}}'$ from equation (33) into equation (32) and then convert this into an expression for $\boldsymbol{q}'$ by bringing all other terms to the right-hand side.

$$\boldsymbol{q}'\left[\frac{4}{\Delta h^2}\boldsymbol{M} + \frac{2}{\Delta h}\boldsymbol{C} + \boldsymbol{K}\right] = \boldsymbol{R}_B + \boldsymbol{R}_S + \boldsymbol{R}_N - \boldsymbol{S}_0 + \boldsymbol{M}\left(\frac{4}{\Delta h^2}\boldsymbol{q} + \frac{4}{\Delta h}\dot{\boldsymbol{q}} + \ddot{\boldsymbol{q}}\right) + \boldsymbol{C}\left(\frac{2}{\Delta h}\boldsymbol{q} + \dot{\boldsymbol{q}}\right) \quad (34)$$

To summarize the entire algorithm to simulate the behavior of the element:

1. Initialize $\boldsymbol{q}^0 = \dot{\boldsymbol{q}}^0 = \ddot{\boldsymbol{q}}^0 = 0$
2. Select the time step Δh
3. Form the matrices $\boldsymbol{M}, \boldsymbol{C}, \boldsymbol{K}, \boldsymbol{S}_0$.
4. For each time step:
   (a) Calculate the load $\boldsymbol{R}'_B, \boldsymbol{R}'_S$.
   (b) Calculate the displacement $\boldsymbol{q}'$ at time $h + \Delta h$ according to the equation just mentioned.
   (c) Calculate the acceleration $\dot{\boldsymbol{q}}'$ and velocity $\ddot{\boldsymbol{q}}'$ at time $h + \Delta h$ from equation (33).

6 Discussion of methods

Table 1 shows the essential features of the page turning techniques that we have surveyed. It contains entries for the two variants of the *peeling* technique, basic peeling and the use of a 3D correction to simulate foreshortening of the paper due to its extension in the z direction; two *geometric wrapping* techniques that involve wrapping the paper around a cylinder and cone respectively; the basic *mass-spring model* and two variants that attempt to simulate more natural bending and twisting respectively; and the *finite element* method, which models the paper as a 3D shell whose thickness is represented by a single element.

6.1 Computation time

The computation time required for each method is largely dictated by the size of the grid, where applicable. Most methods use rectangular grids. However, the first method needs no grid, and the third mass-spring variant uses a triangular grid in order to resist twisting by constraining dihedral angles between neighboring triangles. Grid elements in the finite-element method are three-dimensional, but are arranged in a two-dimensional array because paper is thin enough for a single layer of elements to adequately model its substance.

We simulated a complete page turn with each method, involving many intermediate postures between the beginning, where the page lies flat on the right-hand side, and the end, where it lies flat on the left-hand side. For the first four methods, which do not involve iteration, Table 1 shows the average time taken to compute the configuration of the paper at each posture. For the last

four methods, it shows the average number of iterations to reach a posture, and the average execution time per iteration. The execution time taken by these methods to compute the configuration of the paper at each posture is the product of these two numbers. The time taken to completely turn a page can be calculated by multiplying this product by the number of in-between postures—say 20 in-betweens for a fast $\frac{2}{3}$-second page turn animated at 30 frames/second.

In each case the lower right-hand corner of the page was constrained to move along the same arc. Each point is specified as a position on the grid. The number of possible positions with different x-values is twice the grid size. Times are shown for grids of size 10×10 (18 postures between the paper lying flat on the right side and flat on the left side) and 100×100 (198 in-between postures). The smaller grid gives a rough idea of what the page turn looks like, while the larger one produces a very good visual approximation of the page turn. In practice, one would not use a square grid but tailor it to the aspect ratio of the paper, using an 8×11 or 9×11 grid for $8\frac{1}{2}'' \times 11''$ paper. Our implementations are not optimized and could no doubt be improved by more careful attention to computational details. The code was written in Java, and timings were measured on a 2.8 GHz Pentium 4 processor.

The two-dimensional peeling technique does not use a grid and is essentially instantaneous; it takes 50 μs to compute a posture. The second method, which incorporates a correction for foreshortening in the z direction, does require a grid, and this increases computation time from $O(1)$ to $O(n^2)$ for an $n \times n$ grid. Table 1 gives values of 16 ms for the 10×10 grid and 5.96 s for the 100×100 grid, which are indeed on the order of 100 and 10,000 times the values for the $O(1)$ peeling technique (to within a factor of 3 and 12 respectively).

The time taken by geometric wrapping is $O(n)$ for the cylinder and $O(n^2)$ for the cone. The difference is because in the former case each element of the paper has the same position for all values of z, while in the latter the position depends on the z-value, as described in Section 3. The time taken for the cylinder is 1 ms for the 10×10 grid and 312 ms for the 100×100 grid; the increase is rather more than the factor of 10 that is expected for an $O(n)$ method but does include the time required to output the coordinates of all the points, which adds a small $O(n^2)$ component. The time taken for the cone is 16 ms for the 10×10 grid and 6.44 s for the 100×100 grid.

Now we turn to the physical simulation models in the lower half of the Table. Their stability depends on the value chosen for the integration time step. If this is too large the simulation will be unstable—the paper may vibrate, or, worse still, the computation may blow up and produce infinite values. Although the explicit time integration technique used for the first of these four physical models is the simplest method, it is less stable than the implicit methods that are used for the remaining three, and therefore requires smaller time steps. However, in order to compare the execution time based on the complexity of the model itself, we have chosen the same time step value of $\Delta h = 20$ ms, which works well for all physical models.

For each method, the number of iterations required for the 100×100 grid is far smaller than for the 10×10 one because the previous posture of the page, which is used as a starting point for iteration, is far closer to the current posture on the finer grid. Recall that we are computing 18 in-between postures for the coarse grid and 198 for the fine grid.

Table 1 gives the *average* number of iterations. In physical models the number of iterations required to reach a specified posture varies. More accurate models simulate the stiffness properties of the paper better, which means that there is little change from one iteration to the next. The largest change from one iteration to the next occurs when force is first applied to the page, and therefore only a small number of iterations are needed to reach the first given posture. When the page approaches its equilibrium posture near the end of the simulation, the page deformation reduces, the amount of change in each iteration decreases, and more iterations are required to reach a posture. In practice, the number of iterations ranges from about 200 steps below the average to 300 steps above it for the 10×10 grid, and from about 10 steps below to 20 steps above for the 100×100 grid.

In terms of the grid size, the complexity of the mass-spring models is $O(n^2)$. For all three of them, the time taken for a complete 20-frame page turn animation is about $400 \times 16 \times 20$ ms ≈ 2 min on a 10×10 grid, and about $30 \times 6.5 \times 20$ s ≈ 1 hour on a 100×100 one.

The finite element method is far more complex still, and grows with the fourth power of the grid size. A complete 20-frame page turn animation takes $500 \times 6.1 \times 20$ ms ≈ 17 hours even on the crude 10×10 grid, and we are clearly unable to perform the computation for the larger 100×100 grid because it would take on the order of 10^4 times as long.

Note that in a practical implementation many speed-ups are possible. The integration step size could be carefully tuned. Moreover, one could cache the results of a single page turn (or, perhaps, a selection of different page turns) and texture-map the actual contents of each page onto the cached geometric surface. We did in fact do this in an earlier project (Chu et al., 2003).

6.2 Visual fidelity

Figures 18 and 19 show the visual appearance of the page produced by each method when the lower right-hand corner of the page is placed at three different positions. To achieve this in a geometric simulation, the program is given the desired (x, y, z) coordinates of the page corner—in the case of the two-dimensional peeling

method, just the (x, y) coordinates. For physical simulations, it involves applying a sequence of forces to the lower right-hand corner of the page that are sufficient to bring it to the appropriate point.

Figure 18 shows representative samples of the visual effects produced by the geometric techniques of peeling and wrapping. Figure 18(a) is clearly inferior to 18(b) because, as explained in Section 2, the creased dog-ear has exactly the same area as the part of the page beneath that has been revealed, which is obviously unnatural. Nevertheless, when used in a reactive real-time page-turning simulation users do not, in practice, notice this deficiency. The geometric wrapping models in Figures 18(c) and 18(d) do not fare well. The cylindrical model (Figure 18(c)) looks rather unnatural. The first and third images of the conical model (Figure 18(d)) suffers from the fact that the cone's axis is constrained to lie on the negative y-axis, and the page is therefore hardly curled at all. No doubt different variants of this model could produce more realistic effects.

Some more radical differences can be seen in the images in Figure 19 produced by the physical methods—three mass-spring models of increasing complexity and the finite element method. Because the basic mass-spring method does not model the page's resistance to bending forces well, the page crumples in a way that resembles cloth rather than paper, weighed down by the force of gravity. This can be seen in Figure 19(a). In Figure 19(b) the page is twisted unnaturally, an effect which—although it looks plausible—is impossible to achieve with ordinary paper. The third model tries to correct this twist and overall the simulation looks reasonably natural, but a slight twist can still be discerned in the third image of Figure 19(c). This twisting behaviour does not appear in the finite element method (Figure 19(d)), which looks quite natural.

7 Conclusion

The simple act of turning a page is quite a complicated thing to simulate. This paper has described a variety of models with radically different geometric and physical underpinnings. We have included enough detail to allow readers to understand their basis of operation, and to implement them if desired. We have implemented each algorithm and compared them on the basis of execution speed and visual appearance.

Different degrees of visual accuracy can readily be observed when the simulation results are placed side by side as in Figures 18 and 19. However, these effects generally go unnoticed when the images are viewed individually, particularly when they are animated under the control of a user who is focused on the page content. For a book using ordinary rectangular flexible pages, the simplest model—two-dimensional peeling as in Figure 18(a)—seems adequate for most applications. Although it sacrifices 3D realism, the appearance can be improved by adding visual details to simulate the effect of a smooth bend rather than a sharp crease—some shading on the bend and some shadowing just beneath it—as illustrated in Figure 1.

The peeling model is inadequate when showing the visual effect of several pages being turned together; the cylindrical and conical models are more suitable for this. If the paper's rigidity and/or shape must be varied, or if users are allowed to make creases—for example, folding corners down into dog-ears—geometric models are inadequate and mass-spring models should be used. In a situation with unlimited computing resources where visual accuracy is more important than interactivity, the finite element method will produce the most realistic simulation for any type of paper under any condition. But a heavy price is paid in computation time.

In order to study how people interact with realistic page-turning models, we are experimenting with a lightweight implementation of the peeling method. It provides a quick, easy-to-use, and highly responsive page-turning mechanism and permits the inclusion of hyperlinks and animated media. Our scheme, which is based on Micromedia Flash, is usable, within a web browser, today. The result is appealing and has many advantages over viewing documents as standard HTML or PDF files, including very rapid startup. Examples can be seen in our book gallery at `http://www.nzdl.org/books`.

Table 1. Summary of page turning techniques

Technique	Method	Mesh	Time integration	Iteration/position		Time/iteration (secs)		Complexity
				10×10	100×100	10×10	100×100	
Peeling								
two-dimensional	produce dog-ear effect by simple geometric reflection	-	-	1	1	5×10^{-5}	5×10^{-5}	$O(1)$
with 3D correction	adjust dog-ear with heuristic z-value and spline correction	rectangle	-	1	1	0.016	5.96	$O(n^2)$
Geometric wrapping								
around cylinder	map grid to cylinder whose radius and axis vary	rectangle	-	1	1	0.001	0.31	$O(n)$
around cone	map grid to cone whose angle and apex vary	rectangle	-	1	1	0.016	6.44	$O(n^2)$
Mass-spring model								
basic	calculate spring lengths from Hooke's law and apply limit straining	rectangle	explicit Euler	349	27	0.016	6.38	$O(n^2)$
with bending	apply Hooke's law for elongation, and calculate curvature for compression	rectangle	semi-implicit backward difference	386	30	0.015	6.38	$O(n^2)$
bending, no twisting	take account of bending energy by calculating dihedral angle between neighboring triangles	triangle	implicit Newmark	450	33	0.017	6.94	$O(n^2)$
Finite element								
3D shell model	use stress-strain relationship to derive the deformation from applied forces	3D patch	implicit Newmark	500	-	6.10	-	$O(n^4)$

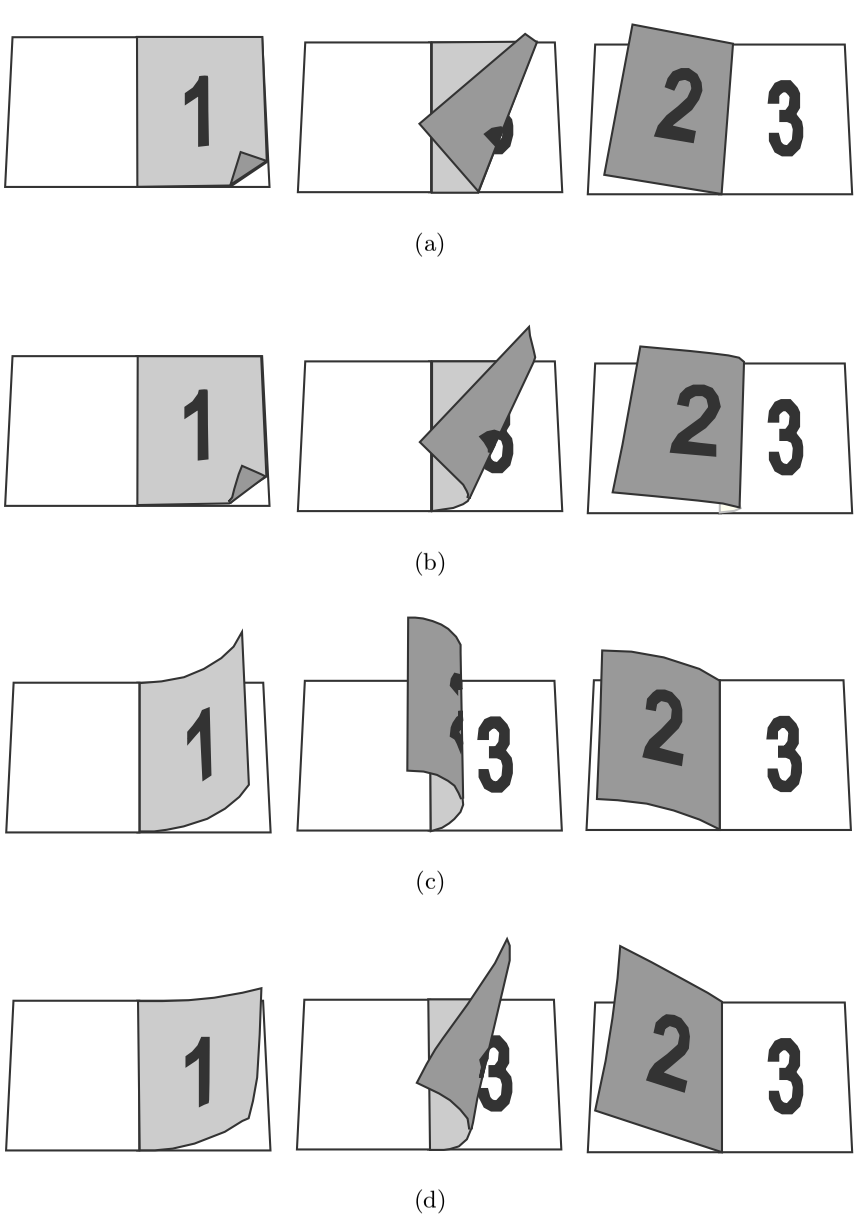


Fig. 18. Geometric page turning simulation using: (a) Peeling 2D, (b) Peeling 3D, (c) Cylindrical, and (d) Conical models

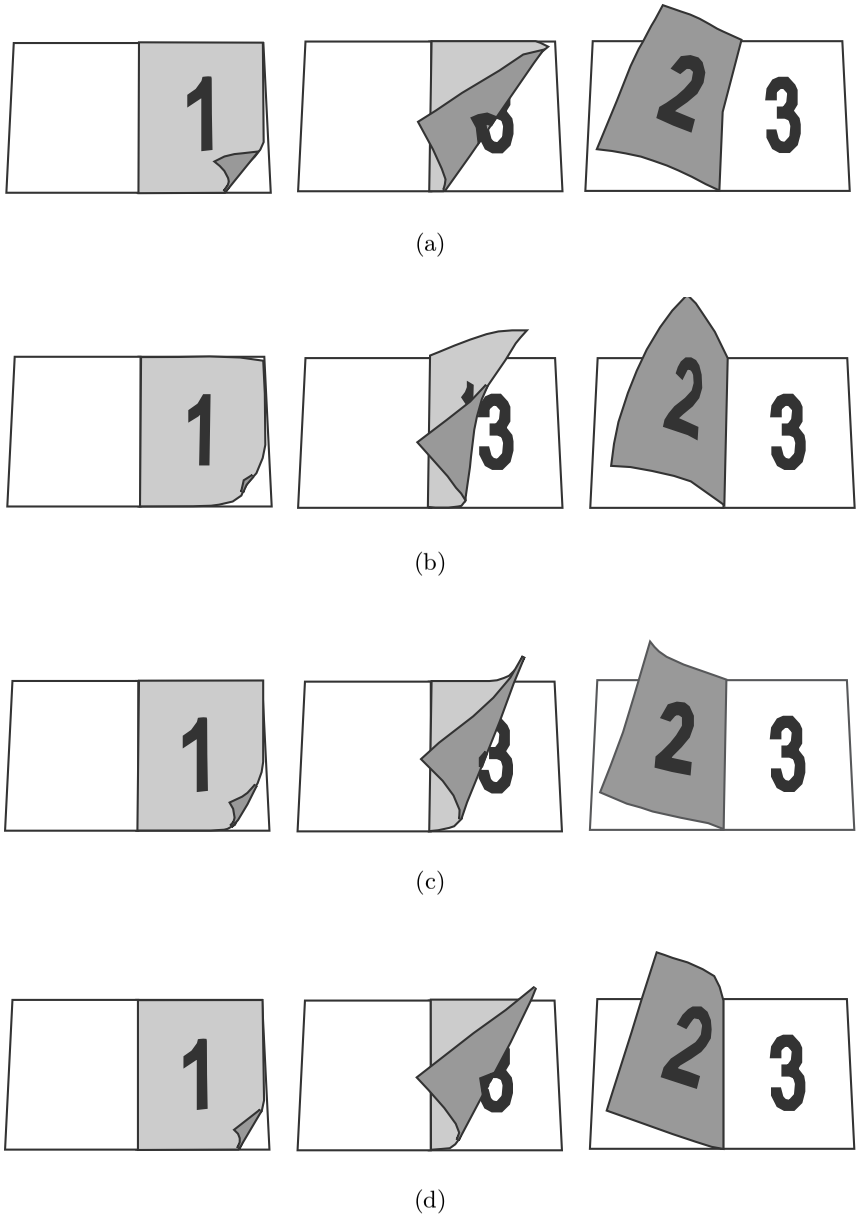


Fig. 19. Physical page turning simulation using: (a) Basic mass-spring, (b) Bending, (c) Bending without twisting, and (d) Finite element method models

Acknowledgments

We acknowledge the entire New Zealand Digital Library Project team for their unstinting work in providing an environment that makes this kind of research meaningful—and fun. This work is funded in part by Google.

References

Bathe, K. (1995). *Finite element procedures.* Prentice-Hall.

Beaudouin-Lafon, M. (2001). Novel interaction techniques for overlapping windows. *UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology*, 153–154.

Bhangal, S. (2004). *The page turn effect in flash mx.* `http://www.oreillynet.com/pub/a/javascript/2004/09/03/flashhac%ks.html`.

Card, S. K., Hong, L., & Chen, J. D. (2006). Turning pages of 3d electronic books. *2006 IEEE Symposium on 3D User Interfaces*, 159-165.

Choi, K. J., & Ko, H. S. (2002). Stable but responsive cloth. *SIGGRAPH 2002 Conference Proceedings: ACM Transactions on Graphics*, *21*(3), 604–611.

Chu, Y.-C., Bainbridge, D., Jones, M., & Witten, I. H. (2004). Realistic books: A bizarre homage to an obsolete medium? *JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital Libraries*, 78–86.

Chu, Y.-C., Witten, I. H., Lobb, R., & Bainbridge, D. (2003). How to turn the page. *JCDL '03: Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital Libraries*, 186–188.

Gotoda, H. (2000). Moving finite element for simulation creasing phenomena of nearly unstretchable sheet materials. *IEEE Computer Graphics and Application.*

Henke, H. (2001). *Electronic books and epublising: a practical guide for authors.* New York: Springer Verlag.

British Library. (2006). *Turning the pages.* `http://www.bl.uk/onlinegallery/ttp/ttpbooks.html`.

Provot, X. (1995). Deformation constraints in a mass-spring model to describe rigid cloth behaviour. *In Proceedings of Graphic Interface'95*, 147–154.

8 Appendix

This Appendix contains technical details of the finite element method calculations that were omitted from Section 5 of the main text. The critical steps concern calculating the strain, obtaining the deformation matrix, and turning the forces into a system of equations that can be integrated to determine the spatial evolution of every finite element at successive time steps.

List of symbols

x, y, z	Global coordinate system
ξ, η, ζ	Local coordinate system
$\hat{x}, \hat{y}, \hat{z}$	Normal-vector coordinate system
$\boldsymbol{p}$	Point in global coordinate
$\boldsymbol{p}_i$	Reference point i in global coordinate
$\boldsymbol{\pi}$	Point in local coordinate
$\boldsymbol{\pi}_i$	Reference point i in local coordinate
$\boldsymbol{n}$	Vector normal to the shell
$\boldsymbol{n}_i$	Vector normal to the shell at reference point i
$\boldsymbol{d}$	Displacement vector
$\boldsymbol{q}_i$	Displacement vector for reference point i
$w_i(\xi, \eta)$	Interpolation function for reference point i
t	Element's thickness
E	Young's modulus
ν	Poisson's ratio
K	Shear correction factor

8.1 Calculating the strain

The strain is determined by obtaining the derivatives of the displacement $\boldsymbol{d}$ with respect to the ξ, η and ζ axes and multiplying these by the inverse of the Jacobian matrix.

The Jacobian is defined in equation (28), and can be calculated by taking derivatives of point $\boldsymbol{p} = (x, y, z)$ from equation (23) with respect to the ξ, η and ζ axes:

$$\frac{\partial \boldsymbol{p}}{\partial \xi} = \sum_{i=1}^{8} \frac{\partial w_i(\xi,\eta)}{\partial \xi}(\boldsymbol{p}_i + \zeta \frac{t_i}{2}\boldsymbol{n}_i)$$

$$\frac{\partial \boldsymbol{p}}{\partial \eta} = \sum_{i=1}^{8} \frac{\partial w_i(\xi,\eta)}{\partial \eta}(\boldsymbol{p}_i + \zeta \frac{t_i}{2}\boldsymbol{n}_i)$$

$$\frac{\partial \boldsymbol{p}}{\partial \zeta} = \sum_{i=1}^{8} w_i(\xi,\eta)\frac{t_i}{2}\boldsymbol{n}_i$$

This yields the value of the Jacobian matrix $\boldsymbol{J}$.

The strain is the derivative of the displacement $\boldsymbol{d}$ with respect to the global coordinates x, y and z. The displacement is given in terms of the local ξ, η and ζ coordinate system by equation (25), and its derivatives with respect to those coordinates are:

$$\frac{\partial \boldsymbol{d}}{\partial \xi} = \sum_{i=1}^{8} \frac{\partial w_i(\xi,\eta)}{\partial \xi}\left\{ \begin{bmatrix} u_i \\ v_i \\ w_i \end{bmatrix} + \zeta \frac{t_i}{2}(\beta_i \boldsymbol{e}_i^{\hat{x}} - \alpha_i \boldsymbol{e}_i^{\hat{y}}) \right\}$$

$$\frac{\partial \boldsymbol{d}}{\partial \eta} = \sum_{i=1}^{8} \frac{\partial w_i(\xi,\eta)}{\partial \eta}\left\{ \begin{bmatrix} u_i \\ v_i \\ w_i \end{bmatrix} + \zeta \frac{t_i}{2}(\beta_i \boldsymbol{e}_i^{\hat{x}} - \alpha_i \boldsymbol{e}_i^{\hat{y}}) \right\}$$

$$\frac{\partial \boldsymbol{d}}{\partial \zeta} = \sum_{i=1}^{8} w_i(\xi,\eta)\frac{t_i}{2}(\beta_i \boldsymbol{e}_i^{\hat{x}} - \alpha_i \boldsymbol{e}_i^{\hat{y}})$$

The strain at time h is obtained by multiplying the above derivatives by $\boldsymbol{J}^{-1}$ according to the chain rule, as

given in equation (27):

$$\boldsymbol{\epsilon} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \end{bmatrix} \boldsymbol{d}$$

$$= \sum_{i=1}^{8} \left\{ \begin{bmatrix} a_i & 0 & 0 \\ 0 & b_i & 0 \\ 0 & 0 & c_i \\ b_i & a_i & 0 \\ c_i & 0 & a_i \\ 0 & c_i & b_i \end{bmatrix} \begin{bmatrix} u_i \\ v_i \\ w_i \end{bmatrix} + \begin{bmatrix} f_i & 0 & 0 \\ 0 & g_i & 0 \\ 0 & 0 & h_i \\ g_i & f_i & 0 \\ h_i & 0 & f_i \\ 0 & h_i & g_i \end{bmatrix} (\beta_i \boldsymbol{e}_i^{\hat{x}} - \alpha_i \boldsymbol{e}_i^{\hat{y}}) \right\}$$

$$= \sum_{i=1}^{8} \boldsymbol{B}_i \boldsymbol{q}_i$$

The strain-displacement matrix for $\boldsymbol{p}_i$ is

$$\boldsymbol{B}_i = \begin{bmatrix} a_i & 0 & 0 & -f_i l_i^{\hat{y}} & f_i l_i^{\hat{x}} \\ 0 & b_i & 0 & -g_i m_i^{\hat{y}} & g_i m_i^{\hat{x}} \\ 0 & 0 & c_i & -h_i n_i^{\hat{y}} & h_i n_i^{\hat{x}} \\ b_i & a_i & 0 & -g_i l_i^{\hat{y}} - f_i m_i^{\hat{y}} & g_i l_i^{\hat{x}} + f_i m_i^{\hat{x}} \\ c_i & 0 & a_i & -f_i n_i^{\hat{y}} - h_i l_i^{\hat{y}} & f_i n_i^{\hat{x}} + h_i l_i^{\hat{x}} \\ 0 & c_i & b_i & -h_i m_i^{\hat{y}} - g_i n_i^{\hat{y}} & h_i m_i^{\hat{x}} + g_i n_i^{\hat{x}} \end{bmatrix}$$

where

$$a_i = J_{11}^{-1} \frac{\partial w_i(\xi,\eta)}{\partial \xi} + J_{12}^{-1} \frac{\partial w_i(\xi,\eta)}{\partial \eta}$$
$$b_i = J_{21}^{-1} \frac{\partial w_i(\xi,\eta)}{\partial \xi} + J_{22}^{-1} \frac{\partial w_i(\xi,\eta)}{\partial \eta}$$
$$c_i = J_{31}^{-1} \frac{\partial w_i(\xi,\eta)}{\partial \xi} + J_{32}^{-1} \frac{\partial w_i(\xi,\eta)}{\partial \eta}$$
$$f_i = \frac{t_i}{2}(a_i \zeta + J_{13}^{-1} w_i(\xi,\eta))$$
$$g_i = \frac{t_i}{2}(b_i \zeta + J_{23}^{-1} w_i(\xi,\eta))$$
$$h_i = \frac{t_i}{2}(c_i \zeta + J_{33}^{-1} w_i(\xi,\eta))$$

Here, J_{ij}^{-1} refers to the component of $\boldsymbol{J}^{-1}$ at row i, column j. The derivatives of the interpolation function $w_i(\xi,\eta)$ with respect to the ξ and η axis are given in the table below.

i	$\frac{\partial w_i(\xi,\eta)}{\partial \xi}$	$\frac{\partial w_i(\xi,\eta)}{\partial \eta}$
1	$\frac{1}{4}(2\xi+\eta)(1-\eta)$	$\frac{1}{4}(1-\xi)(2\eta+\xi)$
2	$\frac{1}{4}(2\xi-\eta)(1-\eta)$	$\frac{1}{4}(1+\xi)(2\eta-\xi)$
3	$\frac{1}{4}(2\xi+\eta)(1+\eta)$	$\frac{1}{4}(1+\xi)(2\eta+\xi)$
4	$\frac{1}{4}(2\xi-\eta)(1+\eta)$	$\frac{1}{4}(1-\xi)(2\eta-\xi)$
5	$-\xi(1-\eta)$	$-\frac{1}{2}(1-\xi^2)$
6	$\frac{1}{2}(1-\eta^2)$	$-\eta(1+\xi)$
7	$-\xi(1+\eta)$	$-\frac{1}{2}(1-\xi^2)$
8	$\frac{1}{2}(1-\eta^2)$	$-\eta(1-\xi)$

8.2 Calculating the deformation matrix

As explained in Section 5.8, the deformation matrix $\widehat{\boldsymbol{D}}$ quantifies the relationship between stress and strain at point $\boldsymbol{p}$ in the normal-vector coordinate system. The transformation matrix $\boldsymbol{Q}$ is used to map $\widehat{\boldsymbol{D}}$ to the global coordinate system. The elements of $\boldsymbol{Q}$ are obtained from the direction cosines of the $\hat{x}, \hat{y}, \hat{z}$ coordinate axes at point $\boldsymbol{p}$ measured in the x, y, z coordinate system.

The $\hat{x}$ axis is in a direction perpendicular to the ξ axis at $\boldsymbol{p}$, namely $\boldsymbol{e}^{\hat{x}} = \frac{J_1}{|J_1|}$ where the vector J_k is the k^{th} row of the Jacobian matrix. The $\hat{z}$ axis is in a direction normal to the shell's surface at $\boldsymbol{p}$, and can be obtained by taking the cross product of two different vectors tangential to this surface. It can therefore be expressed as $\boldsymbol{e}^{\hat{z}} = \frac{J_1 \times J_2}{|J_1 \times J_2|}$. Once the directions of the $\hat{x}$ and $\hat{z}$ axes are established, the $\hat{y}$ axis is chosen to be perpendicular to them both; the unit vector along this axis is $\boldsymbol{e}^{\hat{y}} = \boldsymbol{e}^{\hat{x}} \times \boldsymbol{e}^{\hat{z}}$.

The components of the matrix $\boldsymbol{T}$ are the direction cosines of the normal-vector coordinate system at point $\boldsymbol{p}$, measured in the x, y, z coordinate system:

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{e}^{\hat{x}} \\ \boldsymbol{e}^{\hat{y}} \\ \boldsymbol{e}^{\hat{z}} \end{bmatrix} = \begin{bmatrix} l^{\hat{x}} & m^{\hat{x}} & n^{\hat{x}} \\ l^{\hat{y}} & m^{\hat{y}} & n^{\hat{y}} \\ l^{\hat{z}} & m^{\hat{z}} & n^{\hat{z}} \end{bmatrix}$$

The matrices $\widehat{\boldsymbol{D}}, \boldsymbol{D}$, and $\boldsymbol{Q}$ are all of the form

$$\begin{bmatrix} A_{1111} & A_{1122} & A_{1133} & A_{1112} & A_{1113} & A_{1123} \\ A_{2211} & A_{2222} & A_{2233} & A_{2212} & A_{2213} & A_{2223} \\ A_{3311} & A_{3322} & A_{3333} & A_{3312} & A_{3313} & A_{3323} \\ A_{1211} & A_{1222} & A_{1233} & A_{1212} & A_{1213} & A_{1223} \\ A_{1311} & A_{1322} & A_{1333} & A_{1312} & A_{1313} & A_{1323} \\ A_{2311} & A_{2322} & A_{2133} & A_{2312} & A_{2313} & A_{2323} \end{bmatrix}$$

The suffixes have a slightly different interpretation in each of the three cases. An element in the local deformation matrix $\widehat{\boldsymbol{D}}$ is $\widehat{D}_{rstu}$, where $r, s, t, u \in \{1, 2, 3\}$. The numbers 1, 2 and 3 signify the $\hat{x}, \hat{y}$ and $\hat{z}$ axes respectively. The global deformation matrix $\boldsymbol{D}$ has elements D_{ijkl}, with $i, j, k, l \in \{1, 2, 3\}$—and here the numbers 1, 2 and 3 signify the x, y and z axes. The relationship between $\widehat{D}$ and D is given by the fourth-order tensor transformation

$$D_{ijkl} = \sum_{r=1}^{3} \sum_{s=1}^{3} \sum_{t=1}^{3} \sum_{u=1}^{3} T_{ir} T_{js} \widehat{D}_{rstu} T_{kt} T_{lu},$$

where T_{ab} refers to the component of $\boldsymbol{T}$ at row a and column b. This transformation can be written in matrix form:

$$\boldsymbol{D} = \boldsymbol{Q}^T \widehat{\boldsymbol{D}} \boldsymbol{Q}$$

Finally, $\boldsymbol{Q}$ is an orthogonal matrix with elements Q_{mnop}, where $m, n \in \{1, 2, 3\}$ signify the axes of the global coordinate system and $o, p \in \{1, 2, 3\}$ the axes of the normal-vector coordinate system:

$$Q_{mnop} = \begin{cases} T_{mo} T_{np} + T_{mp} T_{no} & : o \neq p \\ T_{mo} T_{np} & : o = p \end{cases}$$

8.3 Effect of forces

The element's internal energy in Section 5.9 is

$$\begin{aligned}\delta E_K + \delta E_P &= \int_V (\boldsymbol{\delta d})^T (\boldsymbol{F}_I + \boldsymbol{F}_D)\, dV + \int_V (\boldsymbol{\delta \epsilon})^T \boldsymbol{\sigma}\, dV \\ &= \int_V (\boldsymbol{w\delta q})^T (\rho \boldsymbol{w\ddot{q}'} + \kappa \boldsymbol{w\dot{q}'})\, dV + \\ &\quad \int_V (\boldsymbol{B\delta q})^T (\boldsymbol{DBq'} + \boldsymbol{\sigma}_0)\, dV \\ &= (\boldsymbol{\delta q})^T (\boldsymbol{M\ddot{q}'} + \boldsymbol{C\dot{q}'} + \boldsymbol{Kq'} + \boldsymbol{S}_0)\end{aligned}$$

where

$$\boldsymbol{M} = \int_V \rho \boldsymbol{w}^T \boldsymbol{w}\, dV \tag{35}$$

$$\boldsymbol{C} = \int_V \kappa \boldsymbol{w}^T \boldsymbol{w}\, dV \tag{36}$$

$$\boldsymbol{K} = \int_V \boldsymbol{B}^T \boldsymbol{DB}\, dV \tag{37}$$

$$\boldsymbol{S}_0 = \int_V \boldsymbol{B}^T \boldsymbol{\sigma}_0\, dV \tag{38}$$

As noted in Equation (31), conservation of energy implies that this internal energy equates to the element's external energy, which is

$$\begin{aligned}\delta E_T &= \int_V (\boldsymbol{\delta d})^T \boldsymbol{F}_B\, dV + \int_A (\boldsymbol{\delta d}^A)^T \boldsymbol{F}_S\, dA + (\boldsymbol{\delta q})^T \boldsymbol{R}_N \\ &= \int_V (\boldsymbol{w\delta q})^T \boldsymbol{F}_B\, dV + \int_A (\boldsymbol{w}^A \boldsymbol{\delta q})^T \boldsymbol{F}_S\, dA + (\boldsymbol{\delta q})^T \boldsymbol{R}_N \\ &= (\boldsymbol{\delta q})^T (\boldsymbol{R}_b + \boldsymbol{R}_s + \boldsymbol{R}_N)\end{aligned}$$

where

$$\boldsymbol{R}_B = \int_V \boldsymbol{w}^T \boldsymbol{F}_B\, dV \tag{39}$$

$$\boldsymbol{R}_S = \int_A (\boldsymbol{w}^A)^T \boldsymbol{F}_S\, dA \tag{40}$$

Here, $\boldsymbol{w}^A$ is obtained from the matrix $\boldsymbol{w}$ by substituting the appropriate ξ and η values.

To solve for the values of $\boldsymbol{M}, \boldsymbol{C}, \boldsymbol{K}, \boldsymbol{R}_B, \boldsymbol{R}_S$ and $\boldsymbol{S}_0$, three-point Gaussian quadrature integration is used:

$$\int_V f(V)\, dV = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} f(\xi,\eta,\zeta)|\boldsymbol{J}|\, d\xi\, d\eta\, d\zeta$$

$$\int_A f(A)\, dA = \int_{-1}^{1}\int_{-1}^{1} f(\xi,\eta)|\boldsymbol{J}|\, d\xi\, d\eta$$

where $|\boldsymbol{J}|$ is the determinant of the Jacobian matrix. In terms of the grid size, the computational complexity of $f(V)$ and $f(A)$ is $O(n^2)$. The complexity of the finite element method after the integration is $O(n^4)$.