3D Shape Measurement of Moving Objects for Industrial Applications

A thesis
submitted in fulfillment
of the requirements for the degree
of
Master of Engineering
at
The University of Waikato
by
Sean Andrew Charleston

THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato

2015

Abstract

Three dimensional (3D) cameras provide distance measurements to objects, allowing computers and instruments to interact with their environment. The applications are wide-ranging, from human gesture control to industrial processing. Time-of-flight cameras measure the distance to the scene by measuring the flight time of a modulated light source. Sequential captures are required to produce the depth map, hence time-of-flight cameras are vulnerable to depth errors from motion blur in dynamic scenes. This is a major hindrance for industrial applications, where accurate results are required when reconstructing objects. The fruit grading industry is of particular interest for this work, where significant advancements can be made using 3D cameras. The produce moves at a constant velocity, providing an ideal case for initial work into industrial motion correction.

The SR4000 from Mesa Imaging is an industrial grade time-of-flight camera with a high quality factory calibration, and is used throughout this work. When applying custom algorithms (such as motion correction), the camera is run in 'raw mode' where the sequential captures can be individually manipulated, however the factory calibration set is lost. The first part of this work investigates calibrations in time-of-flight cameras, where the factory calibration set in the SR4000 is extracted from the camera to be used on the 'raw mode' data in custom algorithms. The factory calibrated data is compared to both the 'raw mode' data, as well as data acquired using the extracted calibration set. The key results show a root mean squared error (RMSE) of 62.4 mm for 'raw mode' data, while using the extracted calibrations shows an RMSE of 6.1 mm.

The effects of motion blur on time-of-flight cameras are then investigated. The technique from Hussmann *et al.* (2011) provides a good first attempt at motion correction, however fails to implement a number of calibrations. The improvements presented in this thesis on the motion correction technique manipulates the demodulation of time-of-flight cameras so that these additional calibrations are incorporated, resulting in a more robust motion correction algorithm. To test these improvements, a controlled experiment is setup to image a moving spherical object, and a stationary reference image of the same object is captured for comparison. Without motion correction the RMSE is 75.9 mm. Using the naive correction technique from Hussmann *et al.* (2011) gives an RMSE of 58.7 mm, and finally applying the suggested improvements reduces the RMSE to 4.3 mm.

Acknowledgements

Firstly, I would like to give my gratitude to my supervisor, Dr. Michael Cree, for support and guidance throughout this Master's degree. Your writing expertise has been invaluable for completing this thesis, as well the conference paper.

To Dr. Lee Streeter, although you weren't my supervisor, your help and encouragement has been consistent and extraordinarily constructive throughout this work. Even after the many discussions on significance testing and statistics you still answered all of my repeated questions with enthusiasm.

My initial motivation and inspiration into the field time-of-flight cameras came through Dr. Adrian Dorrington, back while I was completing my Bachelor's degree. I thank you for being so passionate, as well as providing mentoring and guidance throughout the time I've known you.

To the Chronoptics research group, I thank you for providing an outstanding learning and working environment, with a lot of great people. I am sure that I have made a number of life friends in the group and hope that we continue to work together and share Friday afternoon beers. I am also very gracious for the use of equipment and technial support provided by the group.

I would like to thank the Dick and Mary Earle Foundation, as well as the University of Waikato, for their financial support in making this work possible. I would additionally like to thank Compac Sorting Equipment, for their support in-kind, providing me with the initial motivation and insight into bringing time-of-flight cameras closer to industry.

Finally to all of my friends and family, who have always provided encouragement throughout my life and university career. I would especially like to thank my parents, Andrew and Stella, who have given so much in raising me and supporting me throughout my life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Machine vision looks to bridge the gap between computation and reality. Humans are equipped with two eyes, which allow scenes to be interpreted in three dimensions (3D) through binocular vision, as well as with the use of visual queues (Howard and Rogers, 1995). Traditional cameras use image sensors to capture incident light. This light is then used to reproduce a scene in two dimensions (2D).

Three dimensional cameras attempt to reproduce scenes, giving the additional dimension of depth compared to 2D cameras. Some applications of 3D cameras include human interaction (such as gesture control or interactive games), or allowing instruments to interact with their environments (such as self-driving cars or industrial robots). The main motivation for the work presented in this thesis arises from 3D imaging in industrial environments, where a 3D camera is used to image a scene, obtaining spatial information about objects of interest. In particular, conveyor type systems are investigated, where objects of interest are moving along a conveyor system, and a 3D camera images the objects. A specific example is the fruit and vegetable grading industry, where produce moving at some constant velocity is imaged, and the 3D shape is reconstructed for analysis.

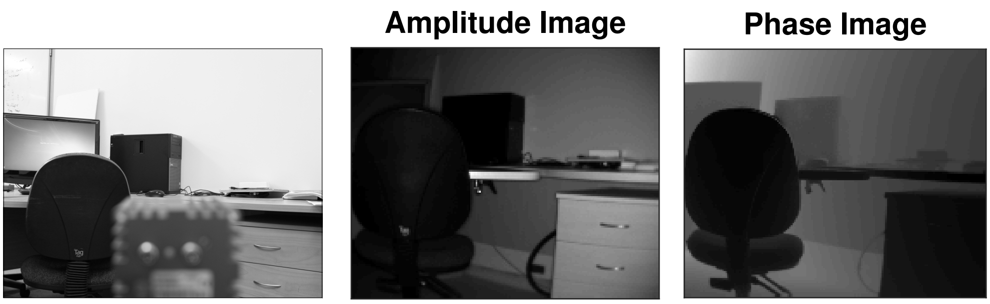
Imaging in three dimensions can be achieved in a number of ways. Stereophotogrammetry uses a principle based on binocular vision, where a scene is captured from two viewpoints (Marr and Poggio, 1976), although multiple viewpoints can be used. Distance to the scene is then calculated through knowledge of the distance and angle between cameras, as well as the correspondence of points/features between the images. The structured light imaging technique (Scharstein and Szeliski, 2003) builds on these ideas, introducing a projector to the imaging system. A pattern is projected onto the scene, and the imaging cameras detect distortion of the pattern to aid in the 3D reconstruction of the scene.

LIDAR (**Li**ght **d**etection **a**nd **r**anging) (Wehr and Lohr, 1999) is a long distance ranging technique, where pulses of light are emitted from a source, and the reflected light is analysed to determine the distance to the object based on the speed of light. Some LIDAR systems utilise a scanning system, where the emitted pulses of light are swept across a scene to develop a complete image, resulting in long acquisition times. Time-of-flight cameras combine a complete image sensor with a light source, producing near simultaneous depth maps of an entire scene at high frame rates (up to 30 fps).

Time-of-flight cameras are based on the principle that the precise knowledge of the speed of light can be used to determine the distance (d) to an object (Lange, 2000), via,

$$d = \frac{ToF \cdot c}{2}, \tag{1.1}$$

where ToF is the flight time of the light travelling to the object and back to the sensor, and c is the speed of light. Time-of-flight cameras are generally used to image close range scenes ($< 15\,\mathrm{m}$). In practice, the distance is often measured indirectly, using a technique called *Amplitude Modulated Continuous Wave* (AMCW), due to the high electronic complexity and cost required to directly measure the flight time of close range scenes (Büttgen *et al.*, 2005). In AMCW systems, the amplitude of the transmitted light is modulated, and the phase change of the received signal is measured to infer distance (Dorrington *et al.*, 2009). These cameras require at least three captures (known as phase steps), in order to generate a single phase and amplitude image recreation of the scene (Figure 1.1). The amplitude image represents the amount of reflected light at each pixel, while the phase image directly relates to the object distance at each pixel.



(a) Conventional 2D camera image. (b) Time-of-flight camera amplitude image. (c) Time-of-flight camera phase image.

Figure 1.1: Computer and desk scene captured with a time-of-flight camera. The conventional 2D camera image shows the time-of-flight camera in red in the foreground. The amplitude image shows the amount of signal retuning to the camera, where lighter colour represents more signal. The phase image relates to the depth of each pixel, where lighter pixels are farther from the camera.

As with traditional 2D photography, 3D cameras require a wide range of calibrations to obtain precise and repeatable results. On top the standard 2D camera calibrations of focal length, pixel pitch, lens calibrations, and fixed pattern noise, time-of-flight cameras introduce a range of additional sources of error such as propagation delays, where signal propagation time within the electronics causes apparent changes in depth, as well as issues from imperfect modulation signal generation.

In general, all imaging systems require an exposure time, in order for electronic sensors to obtain enough light to correctly reproduce a scene. The scene is then able to be digitally manipulated, extracting and analysing useful information. It follows that any motion in the scene or from the camera itself during the exposure time causes motion blur. Parameters can be manipulated to reduce the effects of motion (such as exposure time), however there is an often a trade-off (such as less captured light, resulting in a noisier image). Time-of-flight cameras suffer from additional motion blur, due to the fact that they acquire multiple sequential captures of a scene. As well as the motion during the exposure time (intra-frame motion blur), there is also motion between captures, while data is being read-out from the sensor (inter-frame motion blur).

Mesa Imaging (recently integrated into Heptagon$^{\mathrm{TM}}$ Enterprise Systems `http://enterprise.hptg.com/`) produce a time-of-flight camera known as the SR4000, which is a highly regarded industrial time-of-flight camera. The SR4000 comes fully calibrated in its off-the-shelf state, and is regarded to have a high quality calibration (Chiabrando *et al.*, 2010; Piatti and Rinaudo, 2012). A number of applications, including the work on motion blur presented in this thesis, operate the camera in 'raw mode', where only the phase steps are acquired. In 'raw mode', the factory calibrations are lost, hence it is desirable to reverse engineer the factory calibrations, such that they can be applied even when operating the camera in 'raw mode'.

In the first part of this thesis, a novel technique is presented for acquiring data to be used in calibration, using a translation stage and a retro-reflector. This technique produces planar waves on the camera's sensor, providing a robust calibration dataset. This data is then used to reverse engineer the factory calibrations, which can be applied at will to data acquired in 'raw mode'.

The second part of this thesis provides a thorough investigation into motion blur, which is one of the main hindrances in using time-of-flight cameras in industrial applications. A specific motion correction technique presented by Hussmann *et al.* (2011) is then investigated for its effectiveness in reproducing objects affected by motion. Novel improvements to this motion correction technique are then presented, which looks to alter the demodulation technique used in time-of-flight cameras. These improvements implement the calibrations acquired in the previous section, for a more robust motion correction algorithm.

One of the key applications for the motion correction algorithm looks to investigate the fruit and vegetable grading industry. The produce moves along a conveyor type system at a speed of $1\,\mathrm{m\,s^{-1}}$, which creates a major deterrent when using time-of-flight cameras without motion correction (see Appendix A for further industrial specifications). Compac Sorting Equipment (`http://www.compacsort.com/`) is a global company which is New Zealand owned. They are the global market leader in high-tech systems for sorting and grading produce. These systems include the grading lines themselves, as well as large machines that investigate each fruit for defects and other issues. One of the key issues existing in these systems is in accurately tracking each fruit's position and orientation, to provide additional information. This can aid in reducing false positives, where a piece of fruit is incorrectly rejected due to a normal part of the fruit (such as the stem) being classed as a defect. While it is difficult to directly quantify the financial effects of improving the detection of defects in produce, horticulture is a major industry in New Zealand, providing approximately $2.2 billion in exports (`http://www.mpi.govt.nz/agriculture/horticulture/fruits`). The work in this thesis is supported in-kind by Compac Sorting Equipment (see Appendix A for a letter of support), where the improvement of motion artefacts in time-of-flight cameras could greatly improve their systems. The motion correction techniques presented in this thesis could also provide advances in a wide range of industries, where there is apparent motion as well as a need to acquire 3D images of a scene. The work could be further improved by designing application-specific cameras to meet required specifications.

The work in this thesis is made possible by the Chronoptics research group at the University of Waikato (`http://chronoptics.com`), who specialise in time-of-flight imaging. The group has supplied vital knowledge and equipment for use in completing the work herein.

1.1 Thesis structure

The work in this thesis is broken down into two main segments, firstly analysing and extracting the factory calibration set the SR4000 time-of-flight camera. Secondly, motion blur in time-of-flight cameras is investigated, implementing advancements on existing motion correction techniques for more robust object construction.

Background information is initially provided, on the principles of time-of flight cameras, calibration of the cameras, and finally motion blur. The following chapters then go into detail of the proposed work with calibration extraction and motion correction. Finally, the work is concluded and linked back to the industrial applications of motion correction, specifically investigating the improvements of an apple moving on a conveyor at industrial specifications.

Chapter 2 presents background aspects relating to all parts of the thesis. A general outline of the principles of time-of-flight imaging is first presented. The SR4000 camera is then discussed in depth, outlining its operation and limitations. Following this, a general outline of the noise and error sources in time-of-flight cameras is presented, which includes errors specific to 3D imaging, as well as general sensor calibrations suffered by all cameras, allowing images to projected from 2D to Cartesian 3D coordinates. An overview of previous work in deriving these noise and error sources is discussed, along with how previous authors apply these calibrations. Finally, an overview of the motion problems in time-of-flight cameras is presented, where different sources of motion blur are isolated, and is concluded with a review of previous work in correcting each form of motion blur.

Chapter 3 presents a novel data acquisition technique, where a retro-reflective surface is imaged on a precision linear translation stage. This aims to produce planar waves on the image sensor, providing a more robust calibration image set. Each of the calibrations from the SR4000 are then discussed and extracted through use of the calibration data set. The benefit of extracting these calibrations is in having the freedom to acquire data in 'raw mode', and apply said calibrations when required. This chapter is concluded with a comparison between the off-the-shelf calibrations and the extracted calibrations, using an independent data set.

Chapter 4 covers the aspects of correcting the inter-frame motion blur in time-of-flight cameras. The motion correction algorithm presented by Hussmann *et al.* (2011) is first investigated thoroughly. A novel demodulation technique is then presented, which allows the phase calibrations to be applied to the motion correction algorithm, even with motion aretefacts present. Other time-of-flight calibrations are then discussed and applied, providing a fully calibrated and robust technique for correcting inter-frame motion blur. A statistical analysis is then performed on the motion corrected data, with comparisons to stationary images of the imaged objects. The motion correction is then run on a full speed system, with similar restrictions to that of an industrial conveyor system. An apple is imaged under these industrial specifications, to analyse the accuracy of the motion correction for the specific application described above. In order to assist in the analysis and development of the motion correction technique, a software simulation of a time-of-flight camera is generated, which allows the decomposition of the various motion correction stages. This simulation is discussed and compared to the practically obtained results.

Chapter 5 provides a conclusion, as well an overview of the remaining issues and limitations of the motion-correction technique. An outlook on the current state, and future work required to produce a fully-functional industrial solution is then presented.

1.2 Publications arising from this thesis

The following publication was accepted and presented at the SPIE conference 'Optical Metrology', in Munich, Germany, June 2015 (Charleston *et al.*, 2015), based on the work presented in Chapter 3.

Charleston, S. A., Dorrington, A. A., Streeter, L., & Cree, M. J. 2015. Extracting the MESA SR4000 calibrations. *Pages 95280S-1–95280S-9 of: SPIE Optical Metrology, Videometrics, Range Imaging, and Applications.* Proc. SPIE, vol. 9528.

Chapter 2

Background

2.1 Time-of-flight principles

Time-of-flight range imaging is a metrological technique for measuring distances to objects in a scene. Time-of-flight cameras have two main components: a light source and an image sensor. A modulation source drives the light source, which emits amplitude modulated light toward a scene. A portion of this light is reflected back toward the image sensor on the camera, which is driven by the modulation source at the same frequency as the light source (Figure 2.1). The phase change of the modulation waveform is then used to infer the distance to the object.

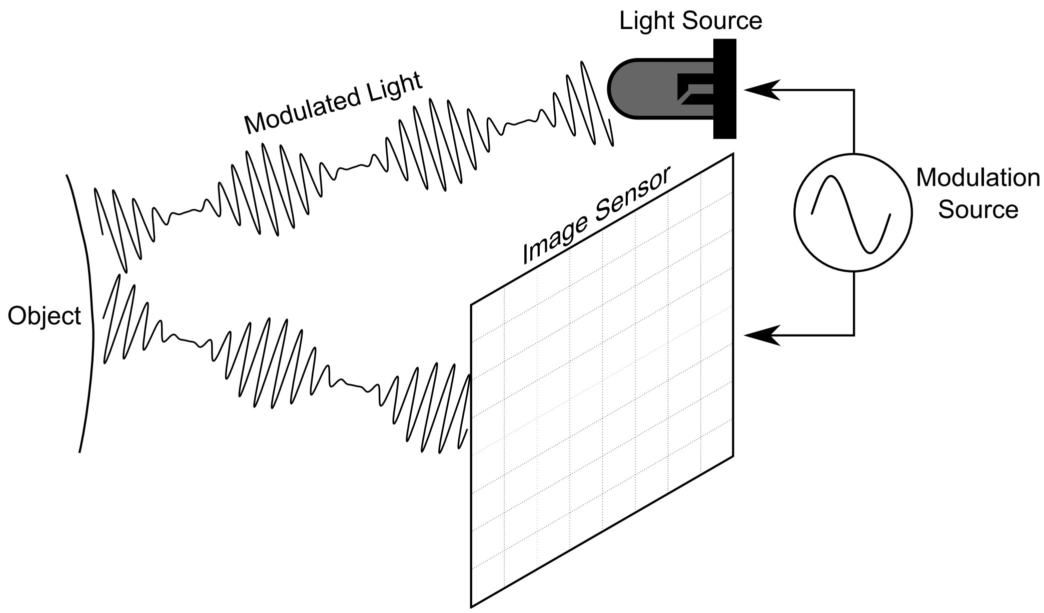


Figure 2.1: Time-of-flight principle. A light source and an imaging sensor are modulated by a source. The transmitted light reflects off an object back toward the image sensor.

Time-of-flight imaging sensors are based on so called 'smart pixels' (Xu *et al.*, 1998). Each 'smart pixel' on the sensor has a modulation window, which allows light to enter one of two charge collecting wells. Time-of-flight cameras using the *Amplitude Modulated Continuous Wave* (AMCW) technique use demodulation to determine the phase change (ϕ), amplitude (α), and background light level (B) of the returning modulation waveform by correlation with the original modulation waveform (Lange, 2000), which is assumed to be sinusoidal (Figure 2.2).

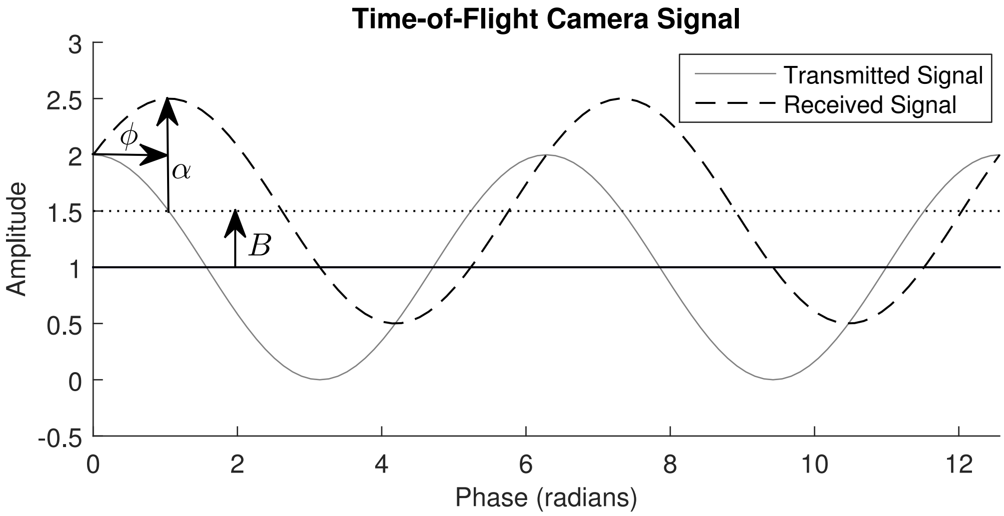


Figure 2.2: Time-of-flight camera signals can be illustrated as a transmitted and received cosine wave. The received signal gives the phase change (ϕ), amplitude (α), and background light level (B) (Source: Charleston *et al.* Proc. SPIE, vol. 9528. "Extracting the MESA SR4000 Calibrations": 2015. `http://dx.doi.org/10.1117/12.2183654`. Used with permission).

A series of captures (phase steps) are taken with the camera, where N consecutive phase steps (τ) are acquired with equally spaced phase offsets (θ) added, from 0 to 2π,

$$\theta_n = 2\pi \frac{n-1}{N}, \tag{2.1}$$

for the nth phase step.

At least three phase steps are required to resolve the phase (ϕ), amplitude (α), and background level (B) of the returning modulation waveform (Payne *et al.*, 2010). Under the assumption that the transmitted wave is sinusoidal, each of the phase steps can be represented as,

$$\tau_n = A\cos(\phi + \theta_n) + B, \tag{2.2}$$

where A is each phase step's amplitude.

The general demodulation technique computes an N-point discrete Fourier transform (Plaue, 2006; Streeter and Dorrington, 2014; Rapp, 2007), returning a complex phasor,

$$P_{jk} = \sum_{n=1}^{N} \tau_{jkn} e^{-i\theta_n}, \tag{2.3}$$

where P is the complex phasor for a particular pixel in row j and column k of the pixel array, and $i = \sqrt{-1}$.

Four phase steps $(\tau_1, ..., \tau_4)$ are typically used, as it simplifies the numerical calculations for hardware implementation (Payne *et al.*, 2008; Cree *et al.*, 2013). For the four phase step case, Equation 2.3 simplifies to the following (Lange and Seitz, 2001; Hsu *et al.*, 2006),

$$\phi = \tan^{-1}\left(\frac{\tau_1 - \tau_3}{\tau_2 - \tau_4}\right), \tag{2.4}$$

$$\alpha = \frac{\sqrt{(\tau_1 - \tau_3)^2 + (\tau_2 - \tau_4)^2}}{2}, \tag{2.5}$$

and

$$B = \frac{\tau_1 + \tau_2 + \tau_3 + \tau_4}{4}. \tag{2.6}$$

Time-of-flight camera signals are modulated at a particular modulation frequency (f_m). This modulated light must travel from the camera, to the object, and back to the camera's sensor. Because the signal is periodic, if the total light travel distance is greater than the wavelength, the signal will wrap and the camera will report an incorrect range. The maximum range that can be measured is known as the ambiguity distance (d_a), which is given by

$$d_a = \frac{c}{2f_m}, \tag{2.7}$$

where c is the speed of light.

The radial distance (d_r) to each pixel can then be calculated, based on the phase change and ambiguity range, by

$$d_r = \phi \frac{d_a}{2\pi}. \tag{2.8}$$

Time-of-flight cameras measure data radially, acting as a point source emitting light in a partial sphere, hence a flat wall appears farther with distance from the camera's central (optical) axis (Figure 2.3). This radial data can be converted into the Cartesian coordinate system, giving an x, y, and z coordinate for each pixel, with respect to the front face of the camera (further discussed in Section 2.2).

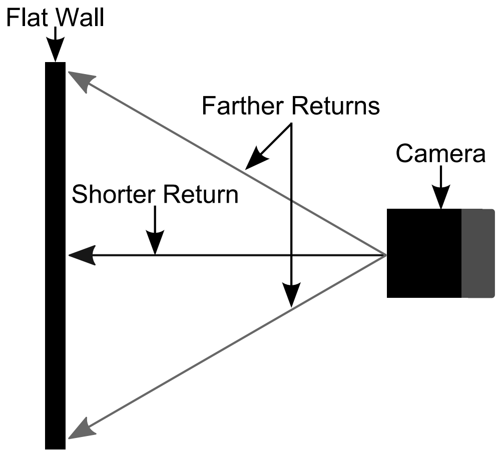


Figure 2.3: Time-of-flight camera radial measurement example. Measuring a flat wall gives farther radial distances with increasing distance from the optical axis.

2.2 Mesa Imaging SR4000

Mesa Imaging produce the SR4000 camera (Mesa Imaging, 2011) (Figure 2.4). The SR4000 operates at a nominal modulation frequency of 30 MHz, corresponding to an ambiguity distance of 5 m, and has an illumination wavelength of 850 nm. The camera has a pixel array size of 144×176, with a field of view of $43.6° \times 34.6°$.

Figure 2.4: Mesa Imaging SR4000 time-of-flight camera.

In standard operation, the SR4000 outputs distance data, amplitude data, a confidence map, and Cartesian data in x, y, and z coordinates (Mesa Imaging, 2010). The distance data is given as the phase angle (ϕ, derived from Equation 2.4) stored in an array of 16 bit data, however only 14 bits are significant. This phase data can be converted to distance in metres, by

$$d_r = \phi \frac{d_a}{2^{14}}, \tag{2.9}$$

giving a distance resolution of

$$\frac{5\,\mathrm{m}}{2^{14}} = 0.31\,\mathrm{mm}. \tag{2.10}$$

Similarly, the amplitude data is output as a 14 bit value, proportional to the amplitude of the received light return. When amplitude saturation occurs, the highest 14 bit value of 0x3FFF (hexadecimal) is reported.

The confidence map outputs a numerical value related to the measurement quality for each pixel, based on a combination of the distance and amplitude measurements, along with time dependent variations. The method for calculating the confidence map is not provided by Mesa Imaging. Low confidence can be caused by low signal reflections or movement in the scene (Mesa Imaging, 2010). The confidence map is output in a 16 bit word, where increasing values represent increasing confidence.

Cartesian (x, y, z) data is given through a coordinate transform function, converting the 14 bit radial data to Cartesian coordinates. Performing this transform also compensates for lens distortion of the optical configuration (Mesa Imaging, 2010). The Cartesian coordinate system has its origin located at the front face on the lens of the camera on the optical axis (Figure 2.5). Increasing z values correspond to increase of perpendicular distance to the camera's lens. Further information on how the transform is performed, and how the lens distortion is corrected is discussed in Section 2.3.

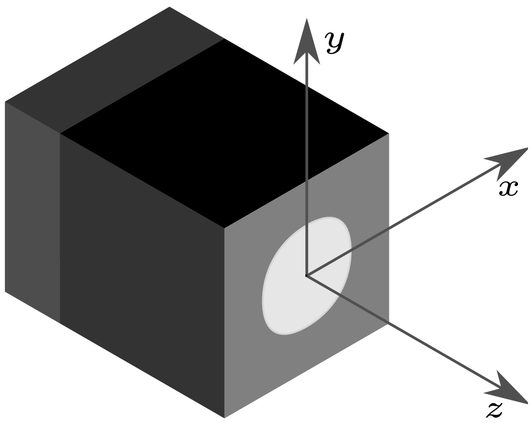


Figure 2.5: Cartesian coordinate system of the SR4000.

The integration time of the SR4000 is able to be manually set by the user, or automatically modified in software with the intention of obtaining a high quality image without saturating the sensor.

The integration time in the SR4000 is set with an 8 bit value ($intTime$), where the values of 0 and 255 correspond to the minimum and maximum integration times respectively. The minimum integration time is 0.3 ms, incrementing in steps of 0.1 ms up to the maximum of 25.8 ms,

$$IT = 0.3\,\text{ms} + 0.1\,\text{ms}(intTime), \tag{2.11}$$

where IT is the total integration time for each phase step. Post integration there a read out time ($RO \approx 4.6\,\text{ms}$) for each phase step, resulting in a total acquisition time (AT) of

$$AT = 4(IT + RO). \tag{2.12}$$

The frame rate is simply given by the inverse of the acquisition time. Using the maximum and minimum integration times, frame rate extremes of 8 FPS and 51 FPS are achieved respectively.

The SR4000 offers a range of filters and calibrations that can be automatically applied to the acquired data. A 'Convert grey mode' filter, and an 'Adaptive neighbourhood filter' are applied by default. The convert grey filter attempts to make the amplitude image (Equation 2.5) closer to that of a standard grey-scale image sensor. This is achieved firstly by compensating the amplitude based on depth, where each amplitude value is multiplied by a factor (γ_ϕ) proportional to its measured distance squared, normalised to the ambiguity distance,

$$\gamma_\phi \propto \frac{\phi^2}{d_a}. \tag{2.13}$$

Secondly, the amplitude image is compensated for illumination irregularities where the sensor is not evenly illuminated, caused by a drop in illumination with distance from the centre of the field of view, and is represented as a multiplicative factor (σ).

The adaptive neighbourhood filter is a hardware implemented 5×5 filter, which combines amplitude and distance information, attempting to reduce noise and preserve detail (Oggier *et al.*, 2012).

There are also enhancements which are not applied by default. These are the 'median filter', which is a hard-coded 3×3 filter, reducing noise using the median of the kernel. An option is also provided for the camera to output the confidence map discussed above.

The SR4000 provides the option to change the modulation frequency. The SR4000 is able to be modulated at 15 MHz, however this frequency is not calibrated for this particular camera. This means that the phase and amplitude data (Equations 2.4 and 2.5), require manual calibration before being processed.

Alternatively to outputting the factory calibrated range (ϕ) and amplitude (α), the four raw phase steps ($\tau_1, \ldots, \tau_4$) are able to be output from the camera. Outputting these phase steps is known as 'raw mode', and is useful when implementing algorithms without the factory calibrations applied, for example the mixed pixel restoration algorithm (Dorrington *et al.*, 2011). 'Raw mode' is also useful when running custom algorithms on the raw phase step data, such as the motion correction technique presented in this thesis.

Because the radial to Cartesian coordinate transform is unavailable whilst the camera is set to acquire data in 'raw mode', Dorrington *et al.* (2011) present a method of retrospectively applying this calibration (along with the associated lens correction). A unit pointing vector $\hat{c}_{jk}$ is derived and stored for each pixel of the array. A 3D point cloud of a scene is obtained in the camera's Cartesian coordinate system, after which $\hat{c}_{jk}$ is populated as

$$\hat{c}_{jk} = \frac{\mathbf{v}_{jk}}{|\mathbf{v}_{jk}|}, \tag{2.14}$$

where $\mathbf{v}_{jk}$ is the vector from the origin to the measured 3D pixel, and $|\mathbf{v}_{jk}|$ is the Euclidean distance from the origin to the measured 3D pixel. The Cartesian coordinate for each radial pixel can then be found by multiplying the radial distance at each pixel with the calibration unit pointing vector

$$\mathbf{v}'_{jk} = \hat{c}_{jk} d_{rjk}, \tag{2.15}$$

where $\mathbf{v}'_{jk}$ is the vector for the Cartesian coordinate, and d_{rjk} is the radial distance acquired for each pixel.

Along with custom software for viewing and manipulating the data acquired by the SR4000, Mesa Imaging supply an application programming interface (API), allowing the camera to be operated in numerous programming languages (C++, C#, and Python) and software packages. The work presented in this thesis processes the data from the SR4000 using a commercial software package (MATLAB 2014b, The MathWorks Inc., Natick, MA), unless otherwise stated.

2.3 Noise and error sources in time-of-flight cameras

Time-of-flight cameras are subject to multiple error and noise sources. As well as requiring calibrations common to traditional two dimensional cameras (photogrammetric calibration (Wiedemann *et al.*, 2008) and lens distortion (Ho, 2013)), the process of acquiring depth measurements adds additional errors and noise in itself (Foix *et al.*, 2011). Depth measurement errors and noise can be both systematic and non-systematic in nature. Systematic errors are able to be calibrated to improve the image, while non-systematic errors are artefacts occurring due to physical fluctuations of the actual measurement process, hence appear as random variation, usually improved by filtering (Foix *et al.*, 2011) or averaging multiple images (Karel *et al.*, 2010). In order to compensate for the errors associated with time of flight cameras, a calibration model must be developed that expands the ideal phase step model (Equation 2.2).

2.3.1 Phase errors

The errors associated with causing offsets in the depth measurements are enumerated. The first error source, fixed in time and independent of range, is a global phase offset (GPO). This constant phase offset is due to fixed propagation delays in the electronics, resulting in a fixed phase offset for all pixels (Figure 2.6), appearing as a non-zero phase axis intercept (that is, $0\,\mathrm{rad} \neq 0\,\mathrm{mm}$). This offset is corrected by applying a constant phase correction to all pixels (Schiller *et al.*, 2008). Secondly, there is a gradual phase offset across the sensor, caused by clocking propagation delays (Fuchs and Hirzinger, 2008; May *et al.*, 2009). This gradual phase offset is usually grouped with the global phase offset, for a constant per-pixel propagation offset. Finally, there is a fixed pattern phase offset, due to additional fixed per-pixel delays, including different material properties in the CMOS gates (Fuchs and May, 2007; Foix *et al.*, 2011). Example data points are illustrated in Figure 2.7, where the three sources of phase error arise as offsets to the target plane. These three fixed phase offsets can be combined into one phase correction, δ_{jk} per pixel.

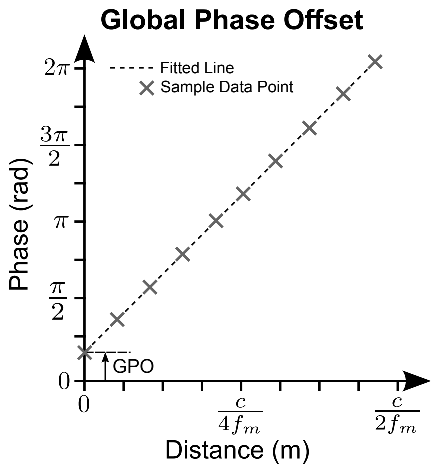


Figure 2.6: The global phase offset is found by comparing the phase output by the camera, to the corresponding distance. The phase axis intercept gives the offset (Source: Charleston *et al.* Proc. SPIE, vol. 9528. "Extracting the MESA SR4000 Calibrations": 2015. `http://dx.doi.org/10.1117/12.2183654`. Used with permission).

Phase offsets that are dependent on the range to the scene (ρ_ϕ) are caused by harmonic distortion, namely harmonics (odd harmonics only if four phase steps are used) of the modulation signal (due to square wave modulation and/or non-linearities) aliased to the fundamental (Lange, 2000; Payne *et al.*, 2008). The error presents itself as a calibratable oscillation on the undistorted phase measurement (Figure 2.8).

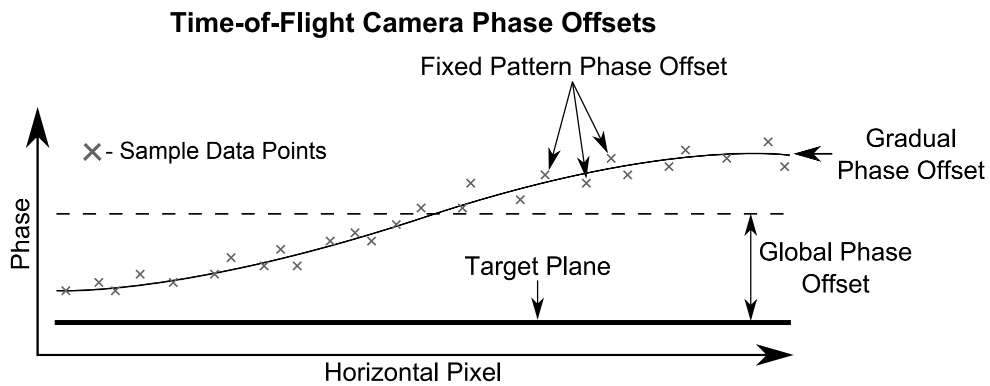


Figure 2.7: A row of example data points across an image that might be acquired by a time-of-flight camera imaging a target plane. The crosses show sample output data points, along with an illustrative curve representing the gradual phase offset. The fixed pattern phase offset is also shown as phase variation from the gradual phase offset. The global phase offset is represented by a fixed offset for all pixels (Source: Charleston *et al.* Proc. SPIE, vol. 9528. "Extracting the MESA SR4000 Calibrations": 2015. `http://dx.doi.org/10.1117/12.2183654`. Used with permission).

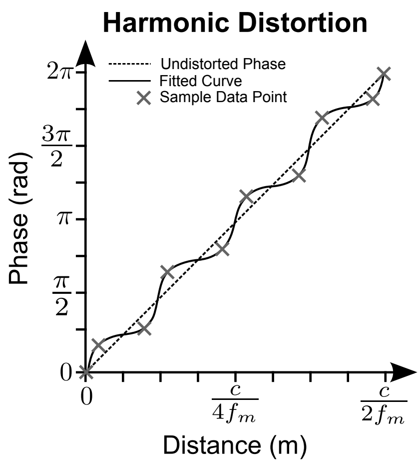


Figure 2.8: Harmonic distortion presents itself as a distance based offset, oscillating about the undistorted phase (Source: Charleston *et al.* Proc. SPIE, vol. 9528. "Extracting the MESA SR4000 Calibrations": 2015. `http://dx.doi.org/10.1117/12.2183654`. Used with permission).

2.3.2 Amplitude errors

There are also three sources of amplitude error. The first is a fixed pattern gain factor (η_{jk}), due to variation in the quantum efficiency of each pixel, as well as differences in integration storage capacitance (i.e., the same number of photoelectrons produce different voltages at each pixel). The result is a multiplicative error dependent on the signal amplitude (Hussmann and Edeler, 2010a). The second source of amplitude error, due to uneven illumination on the sensor (σ_{jk}) is also multiplicative in nature. The spatial output profile of the light source and lens vignetting (Schuon *et al.*, 2008) results in the apparent illumination being strongest in the centre of the image (Kim *et al.*, 2008).

Finally, as mentioned in Section 2.2, the SR4000 includes a distance dependent amplitude correction (γ_ϕ), which looks to correct the inverse square drop in intensity with distance. This is also a multiplier, which is based on the measured phase, and is calculated after signal demodulation. The function of this amplitude correction is to make the amplitude image closer to that of a typical greyscale imaging sensor, hence is not always necessary. Because this is not an undesired artefact of the measured signal, it is not included in the general overview and equation for time-of-flight camera noise and error sources.

2.3.3 Background errors

There are also errors associated with the background light (Equation 2.6). There is a well understood dark current offset per pixel (β_{jk}). This offset is reported by the sensor in the absence of light (Hussmann and Edeler, 2010a), and is caused by small currents produced even without photons incident on the sensor. With increasing integration time, the amount of signal generated from dark current is also integrated, meaning that the correction of dark current is a function of integration time (Büttgen *et al.*, 2005). Additionally, there is an amplitude dependent background offset ($\psi_{\alpha jk}$), where the amount of background signal is dependent on the incident amplitude level for each pixel (Büttgen *et al.*, 2005). The background B from Equation 2.6 can be rewritten as,

$$B = \beta_{jk} + \psi_{\alpha jk}. \tag{2.16}$$

The background part of returning modulation signal (Figure 2.2) can be modified to incorporate the individual background components (Figure 2.9).

The phase step received at a particular pixel including all systematic error sources considered, is finally given by

$$\tau_{jkn} = \eta_{jk}\sigma_{jk}A_{jk}\cos(\phi_{jk} + \theta_n + \delta_{jk} + \rho_\phi) + (\psi_{\alpha jk} + \beta_{jk}). \tag{2.17}$$

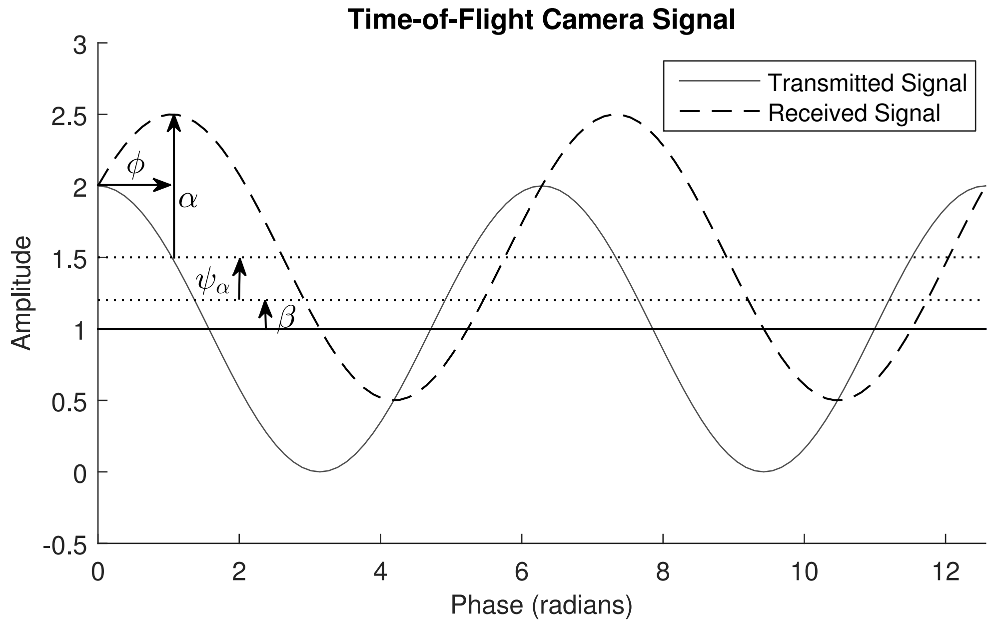


Figure 2.9: Modified version of Figure 2.2, to include the two background offset components.

2.3.4 General time-of-flight errors

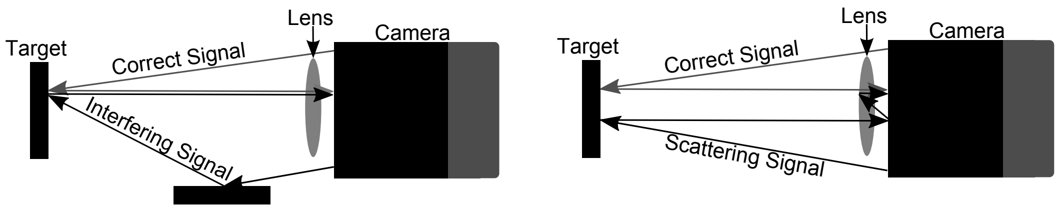
There are other factors that can be dealt with systematically, however do not directly effect the general phase step description of Equation 2.17. The first is a temperature related error, where time-of-flight cameras suffer from a drift in depth until the temperature of the camera is stabilised. This is caused by an increased rate of thermally generated electrons in the CMOS architecture, resulting in the storage sites being partially filled without useful information, reducing the dynamic range of the received signal (Kahlmann *et al.*, 2006).

The integration time of the camera also needs to be considered. A number of authors report acquiring different depth measurements after changing the integration time of the camera, where with increasing integration time, the scene shifts toward the camera (Foix *et al.*, 2011; Kahlmann *et al.*, 2006; Lindner and Kolb, 2007; Radmer *et al.*, 2008). None of these authors provide an explanation as to the cause of the offset, however it is corrected by either keeping the integration time constant during calibration and subsequent experiments (Kim *et al.*, 2008), or running calibrations for a number of different integration times, producing interpolated look-up tables for use with any integration time (Kahlmann *et al.*, 2006).

Lindner and Kolb (2007) investigate additional intensity related errors. They note that the measured depths of objects with lower reflectivity drift toward the camera, however do not identify any origin for this phenomenon. Radmer *et al.* (2008) further investigate this intensity-related error, providing a more robust analysis and model, as well as the ability to handle objects with different Lambertian reflectances.

Multipath interference occurs when multiple light paths interfere and are incident on the same pixel (Foix *et al.*, 2011; Gudmundsson *et al.*, 2007) (Figure 2.10a). Because the sensor is unable to discriminate between the photons from different paths, the multiple photon paths are combined, giving erroneous phase data. Corners are highly affected by this issue, as the emitted light can easily bounce between them, causing apparent rounding of the corner (Mesa Imaging, 2010). Multipath interference is difficult to calibrate for, because it is so highly scene dependent (Gudmundsson *et al.*, 2007). Another type of multipath interference occurs around object edges, where a pixel is unable to differentiate between the near and far surface. These so called 'jump edges' (Foix *et al.*, 2011) or 'mixed pixels' (Godbaz *et al.*, 2009) appear as intermediate phase values between the background and foreground depths.

Lens scattering is a different form of multipath interference, however rather than light scattering in the scene, light bounces and scatters between the sensor and the lens system (Figure 2.10b), resulting in interference to other pixels (Karel *et al.*, 2010). Objects with low amplitude are most affected by the lens scattering (Mure-Dubois and Hügli, 2007), resulting in a degradation of the depth image.



(a) Multipath Interference. (b) Lens Scatter Interference.

Figure 2.10: Multipath interference and lens scatter examples.

The final systematic error that can occur in time-of-flight cameras is motion blur. Motion blur occurs when either the object or camera moves during the capture of the four phase steps, and can be either lateral or radial to the cameras optical axis (Lindner and Kolb, 2009). Motion blur causes the mixing of phase values from different phase steps, resulting in erroneous depth measurements. Motion blur is further discussed in Section 2.5

2.3.5 Photogrammetric calibration and lens distortion

As mentioned above, time-of-flight cameras require a photogrammetric calibration for conversion between radial and Cartesian coordinate systems. This is broken down into three segments; intrinsic parameters, extrinsic parameters, and lens distortion. Intrinsic parameters do not depend on the position and orientation of the camera in space. Extrinsic parameters localise the camera position, allowing for relative conversions between coordinate systems in 3D space. This extrinsic calibration is useful when attempting to combine the camera with other instruments, where they can be converted to share a common coordinate system. Lens distortions are an inherent issue with non-ideal lenses, and are used for image correction when converting between radial and Cartesian coordinate systems.

It is often desirable for the captured radial data to be displayed in Cartesian x, y, and z coordinates, such that results can be directly compared and related to the real word. This also allows data to be calibrated and compared with data from other sources, allowing numerous instruments to work together through the extrinsic calibrations transformations to shared 'world' coordinates (Ma, 2004). To convert between radial and Cartesian coordinates, the pinhole camera model is used (Rapp, 2007). A number of intrinsic calibration parameters are then needed, which are used to project 2D data points into 3D space. A coordinate system is first considered, based on that of Figure 2.5. The optical axis is defined to lie on the z axis, running in a perpendicular line through the geometric centre of the image (u_0, v_0). Each pixel on the sensor is defined by a physical width s_x, and height s_y. The focal length (f) is defined as the distance between the image sensor and the origin of pinhole model. Given a physical pixel location (u', v'), the angle to the optical axis (ψ) can be found from simple trigonometry using the focal length of the camera, and the width and height of each pixel (Rapp, 2007) (Figure 2.11),

$$\psi = \arctan\left(\frac{\sqrt{(u' - u_0)^2 + (v' - v_0)^2}}{f}\right). \tag{2.18}$$

The Cartesian z coordinate (z') is then also found from trigonometry, based on the radial distance (d_r) associated with (u', v'),

$$z' = \cos(\psi) \cdot d_r. \tag{2.19}$$

The corresponding Caretesian x and y coordinates (x' and y') are obtained with similar arguments, however using the respective angles along the Cartesian x and y axes.

In order to efficiently convert from radial to Cartesian coordinates, The focal length, physical pixel sizes, and image centre are stored in a calibration matrix (K) (Ma, 2004),

$$K = \begin{bmatrix} \frac{f}{s_x} & 0 & u_0 \\ 0 & \frac{f}{s_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.20}$$

This calibration matrix can then be used in systems of linear equations to project from 2D to 3D coordinates. It can also be used along with the extrinsic calibration matrices of rotation R, and translation T, to transform to 'world' coordinates (Klette *et al.*, 1998).

As mentioned above, it is desirable to apply corrections for lens distortion when converting from radial to Cartesian coordinates. The distortion model presented by

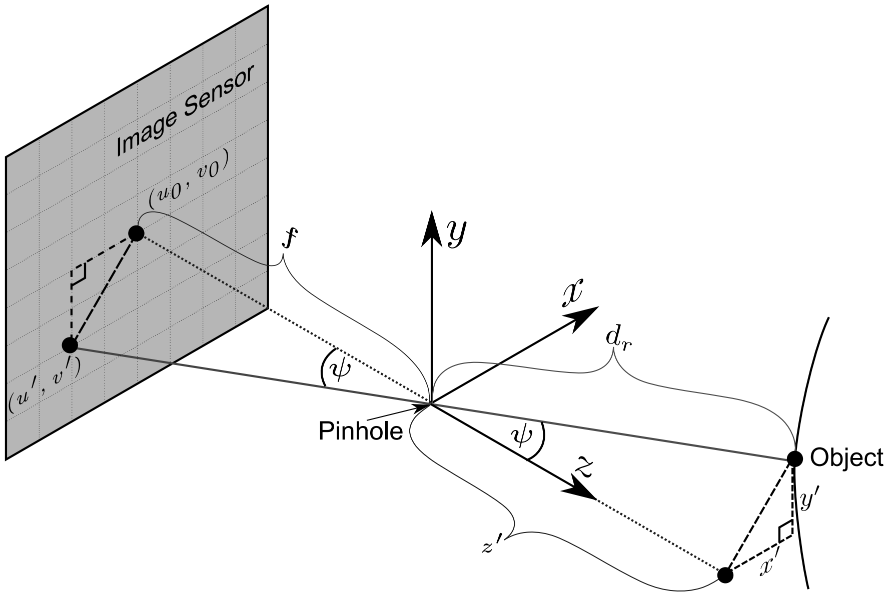


Figure 2.11: Pinhole model for projection between radial coordinates and Cartesian coordinates.

Brown (1971) is generally used to model the radial and tangential distortion (Szeliski, 2010; Tsai, 1987; Zhang, 2000). The radial distortion component causes displacement in the measured image, where the captured points are moved toward (pincushion distortion) or away (barrel distortion) from the image centre, proportional to the radial distance of the pixel (Szeliski, 2010). Tangential distortion occurs when the lens system and the imaging sensor are incorrectly aligned (Weng *et al.*, 1992), resulting in a geometric rotation of the image. Brown's model uses a series of coefficients, which projects a pixel (u', v'), to its distortion corrected position (u'', v''). The basic model has two coefficients (k_1 and k_2) for the radial distortion, and two coefficients (p_1 and p_2) for the tangential distortion. Using each pixel's distance from the image centre,

$$u'_r = u' - u_0, \tag{2.21}$$

$$v'_r = v' - v_0, \tag{2.22}$$

where u'_r and v'_r are the distances of u' and v' from the centre along their respective axes, these coefficients are combined with each pixel's geometric radial distance from the centre of the sensor (r),

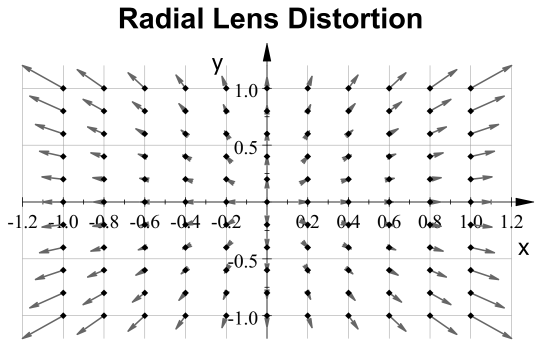
$$r = \sqrt{{u'_r}^2 + {v'_r}^2}. \tag{2.23}$$

The model presented by Brown (1971) for finding the radial and tangential distortion corrected positions, is given as,

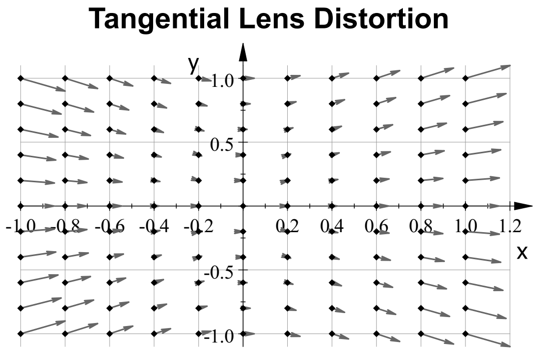
$$u'' = u'_r(1 + k_1 r^2 + k_2 r^4) + 2p_1 u'_r v'_r + p_2(r^2 + 2{u'_r}^2), \tag{2.24}$$

$$v'' = v'_r(1 + k_1r^2 + k_2r^4) + 2p_2u'_rv'_r + p_1(r^2 + 2{v'_r}^2). \tag{2.25}$$

It is unnecessary to apply the distortion model whilst manipulating data in radial (2D) coordinates, because of the geometrically equal nature of the 2D pixel locations. When the lens corrections are applied, each pixel's location is remapped, losing the homogeneity of the image coordinates. This might make the image look more realistic to the eye, however it increases the complexity in performing image processing. When converting to Cartesian (3D) coordinates, the 2D homogeneity is lost, allowing the lens corrections to be applied without drawbacks. Figure 2.12a shows an example of radial lens distortion (barrel), and Figure 2.12b shows an example of tangential distortion (there is an apparent rotation about the y axis, moving the positive x axis closer to the lens). The dots show original pixel locations. Pixels are mapped to new coordinates through distortion correction, shown by arrows.



(a) Radial Distortion.



(b) Tangential Distortion.

Figure 2.12: Radial and tangential distortion examples. Uncorrected pixel locations are shown with dots. Corrected pixel location mappings are shown with arrows.

2.4 Calibration

This section investigates previous works that have attempted to model and calibrate the errors discussed above for time-of-flight cameras. Different calibration techniques are presented for each error source.

2.4.1 Phase calibrations

The fixed phase offset (global, gradual, and fixed pattern phase offsets) calibration techniques are first investigated. Fuchs and May (2007); Fuchs and Hirzinger (2008) use a simple error function based on the pixel geometry to model the gradual and global phase offsets,

$$E_d = b_0 + b_1r + b_2c, \tag{2.26}$$

where E_d is the error term, b_0, b_1, and b_2 are parameters, and r and c are the row and column positions respectively. In this case, the b_0 parameter would store the global phase offset, while the b_1 and b_2 parameters linearly model the gradual phase offset. May *et al.* (2009) also adopt this technique, while Abdo and Borgeat (2010) expand the model to a higher degree polynomial, as the gradual phase offset is often not linear. Schiller *et al.* (2008) use a similar method, however extend the calibration model further, using additional terms with a higher order polynomial to additionally incorporate harmonic distortion. The fixed pattern phase offset is commonly calibrated by imaging a white wall, and measuring each pixel's deviation to the wall, storing the data in a lookup table (Kahlmann *et al.*, 2006). Lindner and Kolb (2006) use a similar technique, noting that if the calibration data is first transformed to Cartesian coordinates, this process can be sped up by comparing each pixel value to the mean wall distance. May *et al.* (2006) point out that storing the acquired calibration in radial coordinates simplifies applying the calibration to future data.

Lindner and Kolb (2006) present a calibration approach for the harmonic distortion, where the deviation between the measured and expected distance is found over a range of 0.75 m to 7.5 m. Rather than modelling the oscillation with a sinusoidal base function, uniform cubic B-splines are used. The advantage of the B-spline approach is a better local control, as well as evaluation advantages in requiring a constant number of operations. Kahlmann *et al.* (2006) present an alternative calibration technique for the harmonic distortion, where a distance measurement track line is used along with a number of targets with varying reflectances. The distance to the targets ranged from 1.25 m to 7.5 m, and the experiment was run with a range of integration times. A lookup table was generated based on the measured phase and the nominal distance. It is noted that modelling the harmonic distortion with cosine functions did not improve results. Fuchs and May (2007) present a similar approach, however the camera is attached to a robotic arm for calibration. This method estimates depth correction and eye-to-hand transformations (extrinsic parameters) simultaneously. The harmonic distortion is corrected for through the use of a polynomial term.

2.4.2 Amplitude and background calibrations

Stürmer *et al.* (2008) investigate the drop in amplitude with distance from the camera. They model this amplitude variation as

$$A_{dist} = I_{src}\frac{1}{(2d)^2+1}, \tag{2.27}$$

where A_{dist} is the acquired amplitude value, I_{src} is the light density of the source, and d is the distance to the scene. A cubic polynomial is then fitted to a plot of inverted mean amplitude versus distance, for a number of integration times. A second polynomial is fitted through the valid points, used in the final scale function for calibration. They also note that the strongest illumination is in the centre, resulting in higher intensities in the middle of the image. Instead of directly compensating for this effect, the scaling function is based on the mean amplitude across the sensor. Hussmann and Edeler (2010a) attempt to correct this uneven amplitude by exposing the sensor to a uniform illumination. A calibration lookup table is then generated, using a scalar multiplier for each pixel. Oprisescu *et al.* (2007) describe a similar post-processing amplitude correction as discussed in Section 2.2, where the SR4000 applies a distance based amplitude calibration, proposing a similar multiplication of the amplitude by the square of the distance. They note however that the depth image must be calibrated for errors first, otherwise the depth image error will also be squared before multiplying the amplitude image.

The dark current part of the background correction is found by simply generating a ‘dark image’, where no light enters the sensor (Hussmann and Edeler, 2010a; Lindner and Kolb, 2007), usually achieved by disabling the light source and applying a lens cap. This ‘dark image’ is then simply subtracted from each of the raw phase steps before processing.

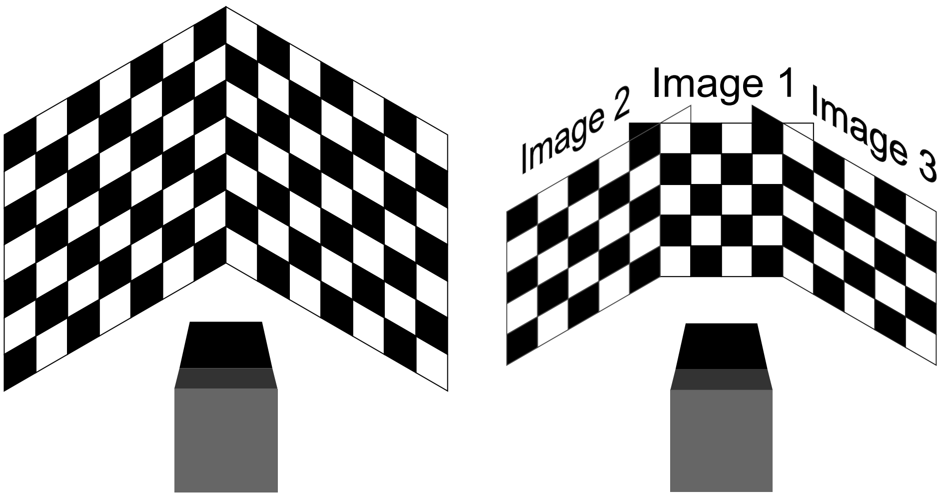
2.4.3 General time-of-flight calibrations

The temperature variation is often dealt with by allowing a warm-up period, and running experiments after this period (Foix *et al.*, 2011). Kahlmann *et al.* (2006) observe that the measured distance to a fixed target increases with temperature, until the process stabilises. Steiger *et al.* (2008) use B-Splines to model the depth offset with changing ambient temperature, using a thermometer in the vicinity at runtime. The SR4000 uses optical feedback to compensate for temperature variation (Lehmann *et al.*, 2009), however is unavailable when running the camera in ‘raw mode’.

Because multipath interference is highly scene dependent, it is difficult to calibrate. Gudmundsson *et al.* (2007) attempt to quantify the effects of multipath interference. An experiment is set up, where a corner is imaged using two perpendicular planes. It is seen that for this particular setup, the 90° corner angle is expanded to approximately 122° after fitting planes to the acquired depth data. Fuchs (2010) attempt to model multipath interference, assuming that all objects are Lambertian reflectors. Under this assumption, recorded depths share a much closer relationship to the received amplitude values, and can be used to aid in the multipath interference correction. Dorrington *et al.* (2011) and Godbaz *et al.* (2009) look to separate multiple return paths incident on a pixel, by capturing the same scene with multiple modulation frequencies. This technique uses the multiple measurements of the scene to generate simultaneous equations, which can be numerically solved to separate the return paths. Karel *et al.* (2010) investigate the lens scatter, showing that it is an additive and linear in nature. An overview of the effect of lens scattering on the depth image is presented, however it is noted that further work is required for modelling and compensation.

2.4.4 Photogrammetric calibration and lens distortion

Intrinsic, extrinsic, and lens distortion calibrations have been widely analysed in the field of computer vision (Faugeras, 1993), with popular approaches coming from Tsai (1987), and more recently Zhang (2000). Both techniques utilise the gridded nature of the checkerboard pattern, which have high contrast squares. Tsai (1987) uses a single image of two perpendicular checkerboard planes (Figure 2.13a), and uses straight lines and edge detection to model the checkerboards. The algorithm then outputs the various parameters for intrinsic, extrinsic and lens distortion correction. The technique proposed by Zhang (2000) on the other hand, requires only a single checkerboard. Multiple images are taken of the checkerboard, moving either the board or the camera between images to obtain different views (Figure 2.13b). At least three views are recommended, and at least eleven views are required for sub-pixel performance.



(a) Tsai (1987) uses a single image of two perpendicular checkerboard patterns for the photogrammetric calibration.

(b) Zhang (2000) uses multiple images of a single checkerboard pattern at different viewpoints for the photogrammetric calibration.

Figure 2.13: Experimental setups for photogrammetric calbiration from Tsai (1987) and Zhang (2000).

A number of attempts have been made to apply these photogrammetric calibrations to time-of-flight cameras (Foix *et al.*, 2011; Ho, 2013). Hussmann and Edeler (2010b) use the method of Tsai (1987) for the calibration, using the amplitude image of the camera. Lindner and Kolb (2006) use a checkerboard with the calibration technique from Zhang (2000), also using the amplitude image for the calibration. It is noted that the intrinsic parameters are detected sufficiently. In their further work, they add a standard two dimensional camera to the system, and successfully calibrate both cameras to share a common coordinate system through the extrinsic parameters. Fuchs and May (2007) then describe an alternative technique for obtaining the extrinsic calibration using the calibration technique of Zhang (2000). They note that the low resolution of time-of-flight cameras makes corner localisation difficult, and mount the camera on a movable robot arm. This arm then moves the camera to a number of depths and performs an extrinsic calibration simultaneously with a depth calibration. Kahlmann *et al.* (2006) describe a different technique rather than using a standard checkerboard. Here a planar test field is populated with LEDs of similar wavelength to the detector. These active targets are extracted from the intensity image, having a very high values, and are then used as inputs to the calibration procedure.

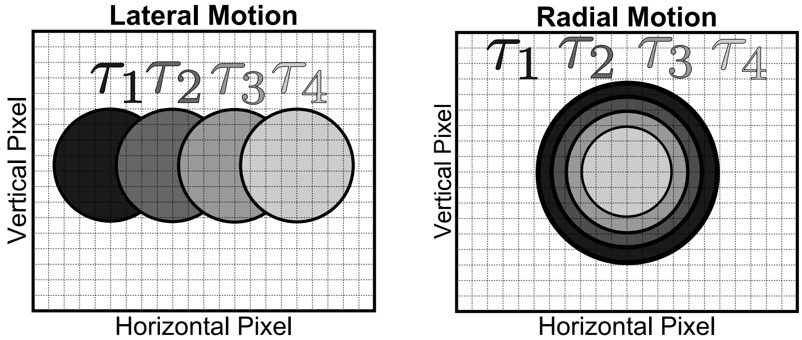
2.5 Motion correction

Time-of-flight cameras require at least three frames (phase steps) to produce a range and amplitude image combination. Motion distortion can occur during image acquisition, if either the camera or an object in the scene moves. The state-of-the-art is to acquire the phase steps sequentially (Lange, 2000; Foix *et al.*, 2011), hence motion between capturing these phase steps causes inter-frame motion blur. Each phase step also has an associated integration time, where the sensor is acquiring photons returning from the scene. Motion during this integration time can also cause blur, known as intra-frame motion blur.

Inter-frame motion blur happens due to the sequential nature of capturing the phase steps, where after each phase step is captured there is an associated read-out time, where the image is transferred from the sensor to memory for later demodulation. Motion can occur during this read-out time, resulting in a mismatch between sequential images. This effect worsens with increasing speed of the motion, increasing the number of phase steps, and increasing the read-out time.

Intra-frame motion blur occurs during the integration time of each frame. The integration is essentially an accumulation of photons incident on a pixel over time. Motion in the scene during this integration time causes adjacent pixels to collect light from the same point in scene, causing blurring. This effect worsens with increasing the integration time, as well as increasing the speed of motion.

Relative to the camera's optical axis, motion blur can be lateral, radial, or a combination of both. Pure lateral motion occurs where an object travels across the sensor's pixels, maintaining a constant radial distance from the camera (Figure 2.14a). Note that lateral motion relative to Cartesian space may have some radial motion associated, due to the radial nature of time-of-flight imaging (discussed further in Chapter 4). Alternatively, the motion can be radial to the camera's sensor, where the object moves closer or farther from the camera over the capture time. An object moving away from the camera would appear to reduce in size through the capture duration (Figure 2.14b).



(a) Lateral motion (b) Radial motion

Figure 2.14: Lateral and radial motion examples.

A number of authors have attempted to correct for motion blur in time-of-flight cameras. Hussmann *et al.* (2011) present a straight-forward approach to correcting lateral inter-frame motion blur in real-time (25 fps). They note that distance uncertainties after demodulation arise where object motion causes misalignment in the raw phase steps. They take a simplified motion case, where an object is moving along a conveyor belt with a constant velocity, requiring motion correction in a single direction. The proposed technique uses the state-of-the-art four phase shift algorithm (Equations 2.4 – 2.6), where the first phase step is used as base, and image processing shifts subsequent phase steps post capture, aligning them with the first phase step. To accomplish this, three binary images (I_B) are generated, obtaining the region of motion from a thresholding of the raw phase step differences ($I_{B(\tau_2-\tau_1)}$, $I_{B(\tau_3-\tau_1)}$, $I_{B(\tau_4-\tau_1)}$). The width of this difference gives the number of pixels the object has travelled, and the phase steps can be shifted for alignment. Hussmann *et al.* (2011) note that this motion correction technique is only possible if the raw phase steps share the same characteristics. The phase steps are calibrated for the offset and amplitude gain multipliers before demodulation (Hussmann and Edeler, 2010a), and the harmonic distortion is corrected after the depth image is produced. There is no mention of the other error sources discussed in Section 2.3, in particular the fixed phase offsets. This technique is shown to easily be integrated into a field-programmable gate array (FPGA) for real-time on chip motion correction.

Lottner *et al.* (2007) describe their technique to detect motion artefacts. They note that when there is motion in the scene, it is not possible to compute distance correctly, and the most affected areas are the edges of objects, where depths can be vastly different when foreground and background objects are mixed into the same demodulation. Their lateral motion investigation places the camera on a moving stage, imaging a perpendicular foreground box edge in front of a flat wall background. A 'simulated motion' approach is used, where the camera is positioned at four evenly spaced positions, and one phase step is captured at each location. When these phase steps are combined, the result is an apparent motion, where inter-frame blur is isolated from intra-frame motion blur. It is seen that there is a large spike at object boundaries, where the depth values are mismatched. After using thresholding to detect erroneous pixels, a suggested correction for these is to identify neighbouring pixels, using a weighted average for depth adjustment. Radial motion is then investigated, where the camera moves toward a still plane, resulting in a systematic positive bias in consecutive phase steps (the reported distance is further than the true distance). No suggestion is offered for correction of erroneous depth values due to radial motion.

Lindner and Kolb (2009) investigate both lateral and radial motion. They introduce the use of optical flow (Horn and Schunck, 1981), where realignment of corresponding depth images is achieved by tracking individual points in objects over time. They assume that the intensity is homogeneous, such that moving points in subsequent phase steps appear with the same intensity. On top of this, pixel homogeneity is assumed, where raw phase step values should match between pixels (each pixel should be fully calibrated). A correction for radial motion is suggested, where an additional optical flow calculation is performed based on the estimated velocity from the two previously corrected depth images.

Jimenez *et al.* (2014) propose an alternative method for motion correction using the four raw phase steps. There is a redundancy in the four phase step demodulation technique, where only two phase steps are required to obtain a depth and amplitude combination. Four phase steps are used in the state-of-the-art case, as it suppresses the background offsets. A calibration is presented which initially corrects this background offset in each of the four phase steps. The result is that only two phase steps are required, giving two redundant phase steps per image capture. Motion is then detected through identifying ‘events’, which are defined as rapid changes in intensity between phase steps, using thresholding. The relative direction of this event (positive or negative) can be used to determine direction of motion. The redundancy in the phase steps is then utilised, where only two of the phase steps are selected for demodulation, based on the location of the detected event in the sequence. In this work, it is assumed that the intra-frame blur is small.

Other unique methods for motion correction include the use of customised sensors, which are able to obtain multiple correlation samples simultaneously (Schmidt, 2008). This reduces the motion blur (depending on the number of samples captured simultaneously), however these sensors are difficult to produce and are not widely used (Jimenez *et al.*, 2014).

Intra-frame blur correction is not well covered in the literature, where it is usually assumed to be small. Streeter and Dorrington (2014) look to adapt the conventional photography technique of coded exposure (Raskar *et al.*, 2006) to time-of-flight imaging. Coded exposure in standard photography uses a ‘fluttered shutter’, which is a physical shutter opening and closing rapidly in a pseudo-random binary sequence.

The result of this fluttering is the manipulation of the Fourier domain of the captured image. The deconvolution of the image and a motion blur function becomes well posed, allowing the moving image to be de-blurred (Wehrwein, 2010).

Streeter and Dorrington (2014) adapt this technique, with the combination of optical flow for correcting transverse motion with both inter-frame and intra-frame motion blur. A sequence of eight phase steps are captured, each of which are coded with a six step binary sequence to influence the blur pattern. Instead of closing the shutter during the zeros of the binary sequence, a phase shift of $\frac{\pi}{2}$ is added, allowing an initial estimate of phase and amplitude for each of the eight phase steps. Optical flow is then used to estimate the direction of motion across the phase steps, after which the intra-frame and inter-frame motion are corrected. The standard demodulation technique (Equation 2.3) is then used to produce a phase and amplitude image combination for the scene. The effects of the motion were improved at the cost of increasing noise.

Chapter 3

Calibration

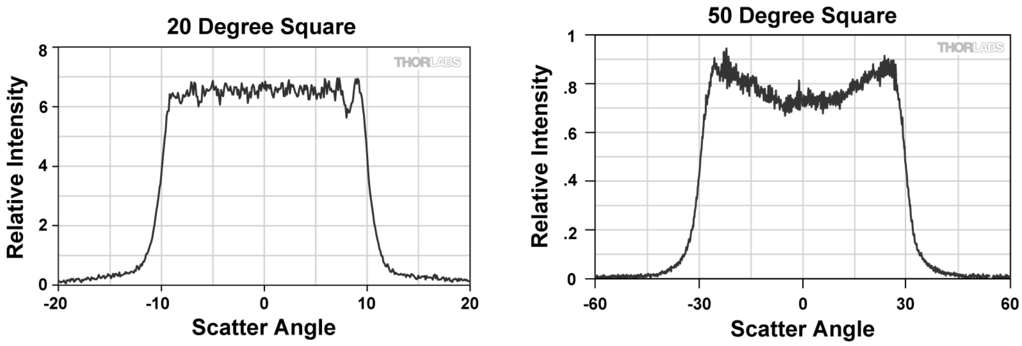
Calibration is a two step process, defined as firstly establishing a relation between quantity values (raw data in this case) and a measurement standard (the factory calibrated data), and secondly using this relation to obtain a measurement result (from the raw data) (BiPM *et al.*, 2008). This chapter investigates the calibrations discussed in Section 2.3. More specifically, the calibration set used in the Mesa Imaging SR4000 (Section 2.2) is investigated. The SR4000 comes fully calibrated in its off-the-shelf state. Unfortunately, these calibrations are lost when acquiring data in 'raw mode', where the raw phase steps are captured for use in algorithms such as the motion correction algorithm presented herein. Other algorithms such as the 'Mixed Pixel Restoration Algorithm' (Dorrington *et al.*, 2011) require data captured in multiple frequencies, whereas factory calibrations are often only available for a single frequency. It is advantageous to be able to apply any number of these calibrations at will, for use with various data sets and algorithms. This chapter is an investigation into how the SR4000 deals with each error source, and attempts to retrieve the associated calibrations where possible. A key part of retrieving these calibrations is with a novel data acquisition technique presented in Section 3.1. The background calibrations discussed in Section 2.3 are unable to be retrieved due to the nature of the demodulation technique in time-of-flight cameras, hence are calibrated using techniques discussed in Section 2.4. Analyses are then performed on the various retrieved calibrations, to analyse their similarity to the data acquired using the SR4000 in its factory calibrated state.

3.1 Obtaining the calibration data

As described in Section 2.4, typical calibration techniques involve imaging a planar target (Kahlmann *et al.*, 2006) to attempt to correct the phase image to some ground truth. One of the downsides of imaging a planar target is that time-of-flight cameras measure the radial distance to the scene, hence a transformation is required if the calibration is to be derived from a fitted plane (Lindner and Kolb, 2006).

A novel data acquisition technique is presented, in which the SR4000 is set up on the edge of a precision linear translation stage (Macron Dynamics, Inc., Croydon, PA, USA). A retro-reflector of size 20 mm by 29 mm is set on the stage, and positioned accurately by the translation stage from 0.5 m up to 3.5 m from the camera. A retro-reflector is a material which is engineered to reflect light back to its source with minimum scattering. The result is that if the camera's light source is assumed to be a point source, the retro-reflector should also approximate a point source, resulting in near planar waves at the sensor. These planar waves should result in flat field phase and amplitude images, as opposed to receiving different phase angles across an image of a flat wall in Cartesian space.

Ideally the lens would be removed so that returning light would propagate directly onto the image sensor, so that any negative effects from the lens system are not apparent during the initial calibration, however the lens cannot be removed from the SR4000 without invalidating the factory calibration set. Instead, a diffuser is placed directly over the camera lens, to distribute the light evenly across the sensor, and help reduce the high intensity light returning from the retro-reflector, preventing amplitude saturation. The requirement of the diffuser is to aid in even light distribution, hence using an ordinary Gaussian filter does not suffice, as the intensity is highest in the centre. Thorlabs (Thorlabs Inc, Newton, NJ, USA) develop engineered diffusers (Thorlabs, Inc., 2015) suitable for the task at hand. These diffusers are engineered with the intention of spreading incident light with specific intensity patterns. For example, the 20° and 50° square pattern diffusers (Figure 3.1), transmit light in a square shape (Figure 3.2), following the relative intensity pattern. The 20° square spreads light with a relative intensity (with respect to a reference diffuser) of approximately 6.5, while the 50° square has a similar pattern, however it has a much lower relative output intensity of approximately 0.8 while spreading light across a wider area.



(a) 20° square diffuser relative intensity plot. (b) 50° square diffuser relative intensity plot.

Figure 3.1: 20° and 50° square diffuser transmitted intensity plots (Source: Thorlabs, Inc. "Engineered Diffusers^{TM}" Technical datasheet.[1] Used with permission).

[1] https://www.thorlabs.com/newgrouppage9.cfm?objectgroup_id=1660

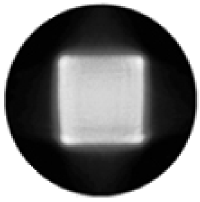


Figure 3.2: Square diffuser transmitted light shape (Source: Thorlabs, Inc. "Engineered Diffusers^{TM}" Technical datasheet.[2] Used with permission).

Because the diffuser spreads the transmitted light evenly across a range of angles, there is an associated increase in total path length for off-axis angles, increasing the measured phase. However, as the diffuser is positioned as close as possible to the sensor, this additional distance is assumed to be negligible. For this experiment, the 50° square diffuser is used, as it gives much lower relative intensity, and spreads the incident light farther across the sensor. The experimental setup of the retro-reflector on the translation stage is illustrated in Figure 3.3. A small tube is added to the system, with the purpose of mitigating light which has scattered off other surfaces (causing multipath interference) from entering the sensor.

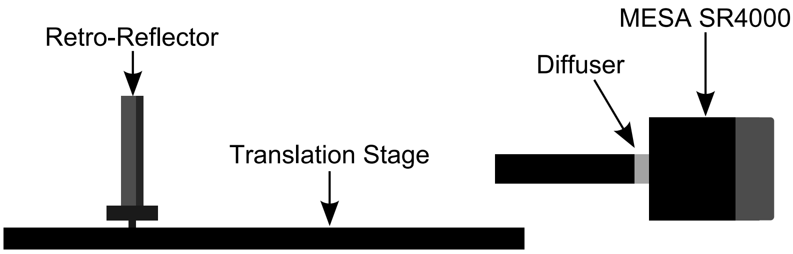


Figure 3.3: The setup used to capture the calibration data is illustrated in a side elevation view. The camera images a retro-reflector on a translation stage. A diffuser is placed in front of the sensor as close to the lens as possible to distribute light, along with a small tube to reduce scattered light entering the camera (Source: Charleston *et al.* Proc. SPIE, vol. 9528. "Extracting the MESA SR4000 Calibrations": 2015. `http://dx.doi.org/10.1117/12.2183654`. Used with permission).

For the initial experiments, inspection showed that using a fixed integration time of 2 ms gives the highest intensity value possible without saturating the sensor at close distances, providing the highest quality signal, as well being sensible for use with the motion correction algorithm discussed in Chapter 4. Other integration times could be used with different diffuser configurations for applications with specific integrations times, or alternatively all integration times could be covered using interpolated lookup tables as discussed by Kahlmann *et al.* (2006). A warm-up time of 60 min is taken for the camera's internal temperature to stabilise, in order for any variation due to temperature to subside.

Using this setup, images are taken with depth increments of 0.05 m over the

[2] https://www.thorlabs.com/newgrouppage9.cfm?objectgroup_id=1660

3 m range, giving measurements at 61 unique distances. A total of 100 images are captured at each distance, and the mean image calculated. Images are acquired from the SR4000, both in the camera's factory calibrated state with selected filters applied, as well as in 'raw mode'. Cartesian transforms are also calculated and stored for the factory calibrated data at each distance. This experiment is repeated at a lower modulation frequency of 15 MHz, to be calibrated for use with the 'Mixed Pixel Restoration Algorithm' (Dorrington *et al.*, 2011) in Section 3.3.6.

3.2 Extracting the error sources

In this section, the error sources discussed in Section 2.3 are extracted from the factory calibrated data in the SR4000, based on Equation 2.17. The equation is repeated here for reference,

$$\tau_{jkn} = \eta_{jk}\sigma_{jk}A_{jk}\cos(\phi_{jk} + \theta_n + \delta_{jk} + \rho_\phi) + (\psi_{\alpha jk} + \beta_{jk}). \tag{3.1}$$

3.2.1 Phase calibrations

The phase offsets in the SR4000 that are corrected by the factory calibration are analysed, and the corrections extracted. The first error source found is harmonic distortion (ρ_ϕ), by taking the mean phase value at each distance (for both raw data and factory calibrated data), and plotting the phase of the raw data against the factory calibrated distance (Figure 2.8). If the translation stage were to cover the entire ambiguity distance of the SR4000 (5 m), the harmonics could be directly found and compensated for, but in this case images are only captured in the 0.5 m to 3.5 m range. The difference between the raw phase data and the factory calibrated data at each distance is then stored in a look-up table. The harmonic distortion is corrected by simple subtraction of the values from the look-up table. Interpolation provides estimates of the values in-between the measured points. In this case, distances recorded outside the measured range (0.5 m to 3.5 m) are not able to be corrected. For the applications in this thesis, this range of distances is sufficient. This range can be adjusted by moving the camera with respect to the experimental set-up on the translation stage, or extended by using a longer translation stage. Shifting the camera and repeating the experiments would allow for the complete ambiguity distance to be covered, however repositioning the camera would introduce more uncertainty to the measurements.

The fixed phase offsets (δ_{jk}) are independent of distance, and are separated into a global offset, a gradual offset, and a fixed pattern offset. The global phase offset is found in a similar way to the harmonic distortion, by plotting the raw phase values (after being corrected for harmonic distortion) against the factory calibrated distance values. This produces a linear slope, and the offset is found as the intercept on the phase axis (Figure 2.6). This value is then stored to be subtracted from the phase value of all pixels in an image.

The gradual phase offset is then corrected, which presents as a gradual change in phase across the image sensor. In the case of the SR4000, the pattern is almost entirely horizontal across the sensor (i.e., there is minimal gradual phase offset in the vertical axis). This offset is corrected by fitting a surface to the raw image data (with the harmonic distortion and global phase offsets corrected). A similar surface is fitted to the factory calibrated data, and the difference between the two surfaces at each pixel is stored in a look-up table, to be subtracted from the phase data during calibration.

Once the gradual offset is corrected, the final phase calibration of the fixed pattern phase offset is found. Once the harmonic distortion, global phase offset, and gradual phase offset are subtracted from the raw phase data, this should be the only remaining phase offset, and is found from the remaining difference between the raw phase data and the factory calibrated data. This offset is also stored in a look-up table. The global, gradual, and fixed pattern phase offsets can be optionally combined (δ_{jk}) and stored in a single look-up table for speed during calibration.

3.2.2 Amplitude Calibrations

The amplitude offsets of the SR4000 are then found. Hussmann and Edeler (2010a) suggest the use of a flat plane for amplitude calibration, as all pixels should receive uniform illumination, and differences in this illumination can then be calibrated out. Due to the fact that time-of-flight cameras measure light radially, a flat plane gives reduced amplitude with distance from the centre of illumination, as the distance the light travels increases with distance from the centre of illumination. An ideal target would be the inside of a diffuse sphere, with the camera at the centre of the sphere. In order to also calibrate for the distance dependent harmonic distortion, a number of concentric spheres would be required, resulting in a very high cost solution. The proposed technique with a retro-reflector should result in a uniform illumination, as near planar waves should be incident on the sensor, giving a similar effect using a much simpler setup with common components for a much lower cost.

The amplitude calibrations have two major parts. The first part is investigating the fact that the amplitude is non-linear over distance (γ_ϕ). One correction for this comes from modelling the amplitude decay with distance (Stürmer *et al.*, 2008). Secondly, the uneven illumination on the sensor (σ_{jk}) is also a multiplicative factor, where the apparent illumination is strongest in the centre (Kim *et al.*, 2008). The SR4000 in its off-the-shelf state provides a calibration for both of these amplitude corrections, however they are combined into the same on-camera calibration (Mesa Imaging, 2010). These both multiply the amplitude image, however only the factor due to uneven illumination (σ_{jk}) is desired for the calibration set (because the distance based calibration is applied in post-processing). The distance based correction from the SR4000 is a multiplier which is proportional to the square of the measured distance, normalised to the ambiguity distance,

$$\gamma_\phi \propto \frac{\phi^2}{d_a}. \tag{3.2}$$

Dividing γ_ϕ from the raw amplitude, then taking the ratio of the factory calibrated amplitude to the raw amplitude gives the radial amplitude correction caused by uneven illumination,

$$\alpha = \hat{\alpha}\gamma_\phi\sigma_{jk}, \tag{3.3}$$

where $\hat{\alpha}$ is the raw amplitude.

The gain fixed pattern correction (η_{jk}) is then found from a uniformly illuminated sensor. With a uniformly illuminated sensor, no other calibrations are required (such as illumination calibration) to find the gain fixed pattern correction. A multiplicative factor is found for each pixel, based on the mean amplitude value of the sensor.

3.2.3 Background Calibrations

Background corrections are not able to be obtained from the factory calibration set. This is because time-of-flight cameras inherently correct for background offsets during standard demodulation (Equation 2.3). The dark current offset (β_{jk}) is usually found in the absence of light, taking an image of a scene without any illumination (Hussmann and Edeler, 2010a). The SR4000 cannot easily be operated with the light source disabled which made the measurement of the dark current offset more difficult than it otherwise would have been. An ideal solution would be to take an image in an infinite dark open space, where light emitted by the SR4000 is unable to return to the sensor. Unfortunately this is impossible in practice, so an alternative solution is to take an image of a large open space in darkness, such that there is minimal light returning from the camera's light source, and no external light from the environment is entering the sensor. Alternatively, the technique used herein applies a foam (or other non-scratching) gasket to the edge of the lens, and a lens cap is attached. This can be experimentally difficult with the way the SR4000 is designed, and there can still be some light leakage through to the sensor. However, this technique is much simpler to setup than capturing an image of darkness, at the potential cost of the quality of the result.

The other background correction is the amplitude dependent background offset ($\psi_{\alpha jk}$). Because this correction is dependent on the amplitude, the ideal way to correct for this would be to demodulate the four phase steps to find the amplitude, after which the background of four phase steps could be corrected. This is a redundant step in standard camera operation on a static scene. However, this calibration is required for the motion correction algorithm presented in Chapter 4, because the inherent background correction is lost. This calibration is found by plotting amplitude versus background light (with the dark current subtracted), for each pixel (Figure 3.4). A linear relationship between the background and amplitude for each pixel is observed, hence each slope and offset calibration can be stored in a lookup table for each pixel.

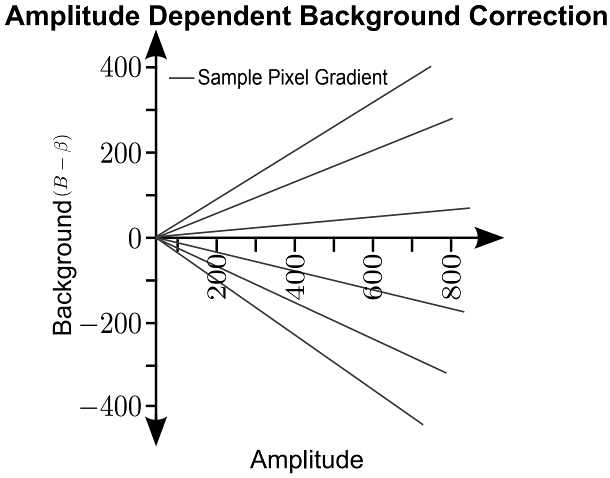


Figure 3.4: Amplitude dependent background correction. Each line is the characteristic of a distinct pixel, and has an associated gradient and offset for calibration.

3.2.4 Photogrammetric calibration and lens distortion calibrations

Intrinsic Calibrations are required when converting between radial and Cartesian coordinates. Because the SR4000 provides a Cartesian transform, this calibration is not explicitly required for this work. Unless otherwise stated, the method presented by Dorrington *et al.* (2011) is used in the conversion between radial and Cartesian coordinates after the raw data has been calibrated. Alternatively, Mesa Imaging provide example code for extracting the camera's intrinsic calibrations using C++ with OpenCV (Bradski, 2000). Nevertheless, an attempt is made to retrieve the camera calibrations using the technique presented by Zhang (2000), along with MATLAB's camera calibration tools.

The calibrations are then tested on an independent set. A flat wall is imaged from a number of angles and distances. An analysis is performed, to compare the factory calibrated SR4000 data to both the raw data and the data calibrated using the described techniques.

3.3 Results and discussion

As mentioned above, the integration time is kept constant for each calibration constructed and used in this Thesis. Sufficient warm-up time of at least 60 min is also allowed for the phase drift due to temperature to stabilise. A simple experiment is performed to analyse the magnitude of the warm-up related error, where the camera is run for 60 min, capturing a depth image every 30 s. The result shows that the factory calibrated phase image remains almost stable over this time period, while the phase from the raw data increases with time (hence temperature). A total drift of 0.07 rad is seen (Figure 3.5), which corresponds to a distance of 55.7 mm for a modulation frequency of 30 MHz. While this effect could potentially be modelled to calibrate the phase drift during warm-up, more work is required to determine if this is consistent over multiple experiments, and whether external temperature affects the warm-up time.

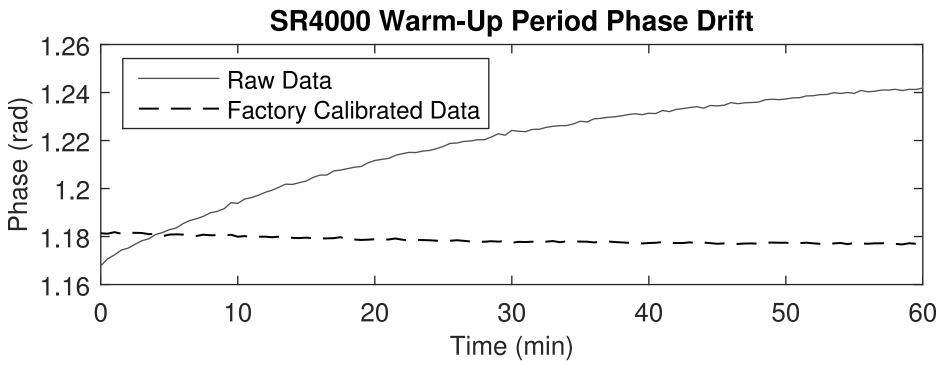


Figure 3.5: Phase variation during SR4000 warm-up period.

3.3.1 Phase calibrations

Using the techniques described above, the phase offsets are extracted from the SR4000. The harmonic distortion of the raw data is shown in Figure 3.6, which is analogous to the example data shown in Figure 2.8. Only the first 1.5 m of the data is shown for viewing purposes, as the difference between the raw data and the factory calibrated data becomes difficult to visualise with increasing scale.

Taking the difference between the raw data and the factory calibrated data shows the oscillation over distance (Figure 3.7). It can be seen that the absolute maximum harmonic distortion offset for the range of 0.5 m to 3.5 m is approximately 0.035 rad, relating to a distance error of 28 mm. Each of the 61 offset values (along with associated factory calibrated distance) are stored in a lookup table, and interpolation is used to find the harmonic distortion error when applying the calibration. This error is then directly subtracted from the phase image output by the camera.

The global phase offset (Figure 3.8) is found where the raw data has been extrapolated to find the y-intercept (after subtracting the harmonic distortion), when plotted against the factory calibrated distance. This graph is analogous to the example data shown in Figure 2.6. The global phase offset for this particular camera is shown to be 0.057 rad, or 45.36 mm. This value is stored and subtracted from all pixels during phase calibration.

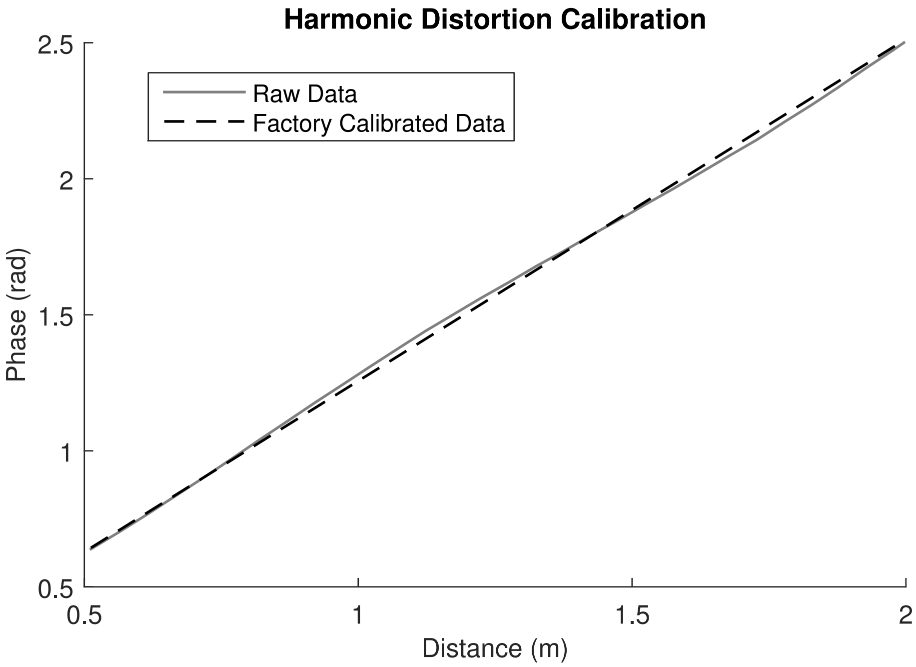


Figure 3.6: Harmonic distortion is seen where the raw data oscillates about the factory calibrated data (Source: Charleston *et al.* Proc. SPIE, vol. 9528. "Extracting the MESA SR4000 Calibrations": 2015. `http://dx.doi.org/10.1117/12.2183654`. Used with permission).

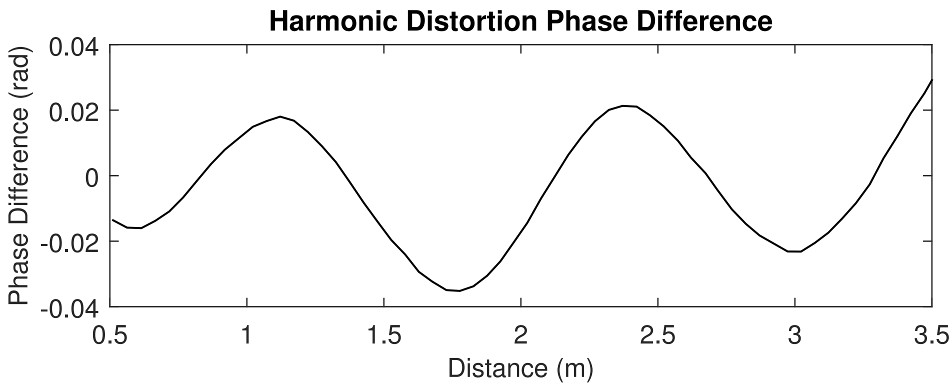


Figure 3.7: The difference between the raw data and the factory calibrated data shows the harmonic distortion.

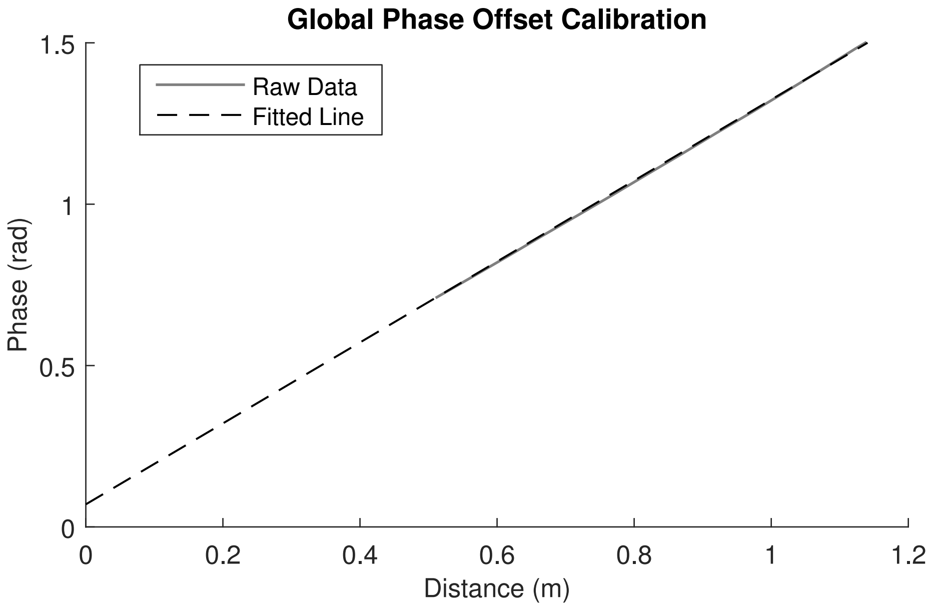


Figure 3.8: The global phase offset is found as the phase axis intercept of the extrapolated raw data (Source: Charleston *et al.* Proc. SPIE, vol. 9528. "Extracting the MESA SR4000 Calibrations": 2015. `http://dx.doi.org/10.1117/12.2183654`. Used with permission).

Fitting a surface to the remaining phase offsets (fixed pattern and gradual phase offsets) gives a calibration surface for the gradual phase offset (Figure 3.9). A drastic change along the horizontal pixels can be seen, while the vertical pixels have a more consistent change.

The absolute maximum for the gradual phase offset is approximately 0.076 rad, or approximately 60 mm. These changing slopes are consistent across the respective rows and columns. Taking the mean of all horizontal and all vertical pixels, gives a better idea of the gradual offset in each direction (Figure 3.10). Finally, a similar surface is fitted to the factory calibrated data, which should be a flat plane at the measured distance. A lookup table is used to store the difference between these two surfaces, to be subtracted from each pixel's phase value during calibration. An alternative calibration technique could look to store parameters of polynomial equations for the vertical and horizontal gradual phase offsets, where the gradual phase offset for each pixel can then be calculated as required.

After removing all other phase offsets from the raw data, the only remaining calibration is the fixed pattern phase offset (Figure 3.11). There are a number of large peaks in this pattern, where the absolute maximum is 0.17 rad, or 135 mm. The majority of the fixed pattern is not as extreme, hence the mean absolute error of this fixed pattern is found as 0.015 rad, or 12.2 mm, with a standard deviation of 0.0122 rad, or 9.7 mm.

A single pixel row can be isolated, to examine typical variation (Figure 3.12). There is no obvious trend or offset in the pattern for this particular row, which

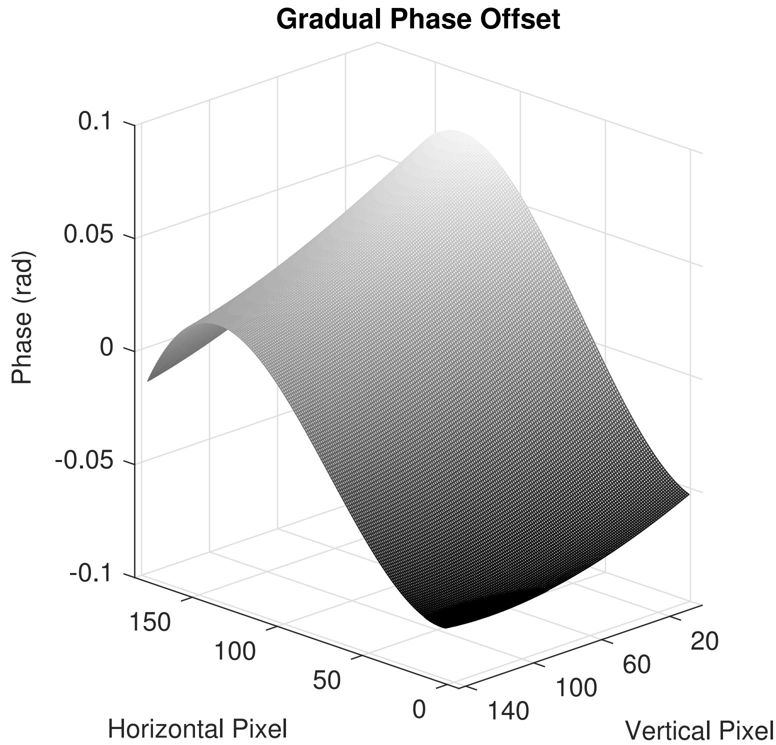
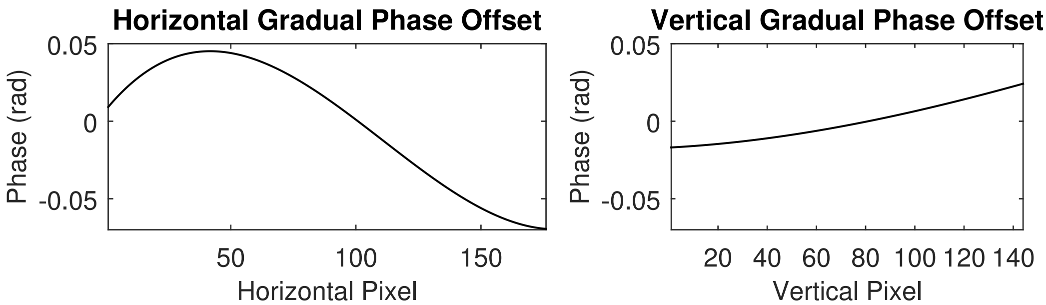


Figure 3.9: A surface is fitted to the raw data to find the gradual phase offset.



(a) Gradual horizontal offset.

(b) Gradual vertical offset.

Figure 3.10: The gradual phase offset can be separated into its respective horizontal and vertical components. It can be seen that the horizontal gradual offset is much more severe than the horizontal offset.

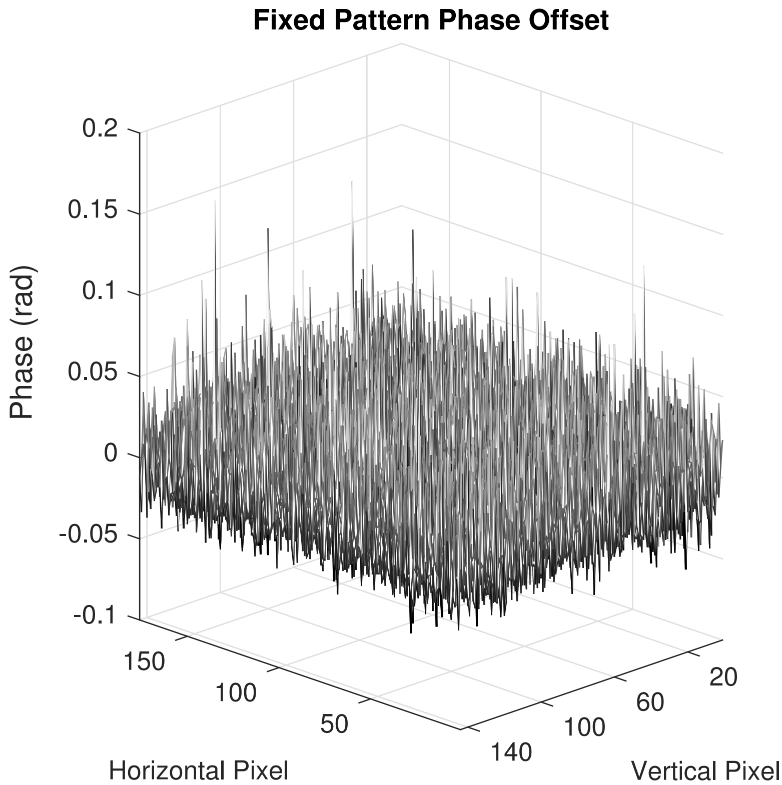


Figure 3.11: The fixed pattern phase offset is found as the remaining difference to the factory calibrated data.

Table 3.1: Comparison between phase offsets, showing the absolute maximum, mean, and standard deviation for each technique.

Absolute Error	Harmonic Distortion	Global Phase Offset	Gradual Phase Offset	Fixed Pattern Phase Offset
Maximum (mm)	27.9	45.4	60.5	135
Mean (mm)	12.3	-	28.8	12.3
SD (mm)	7.15	-	16.3	9.71

was confirmed with an autocorrelation. Taking the mean of all values in the fixed pattern gives a value that is negligibly close to 0 rad (to machine double precision), implying that the global and gradual offsets have been calibrated correctly for this data set. This fixed pattern phase offset is also stored in a lookup table, where each pixel's respective value can be subtracted from the phase value during calibration.

Comparing the absolute maxima of these fixed pattern phase offsets (Table 3.1) shows that the most severe absolute phase offset comes from the fixed pattern phase offset, however analysing the mean absolute errors shows that the gradual offset has a higher effect on the phase overall. The standard deviations (SD) of these absolute means shows that the gradual phase offset also has the highest variation.

All of the distance independent phase offsets (global, gradual, and fixed pattern) can then be combined into a single phase offset calibration (Figure 3.13). The

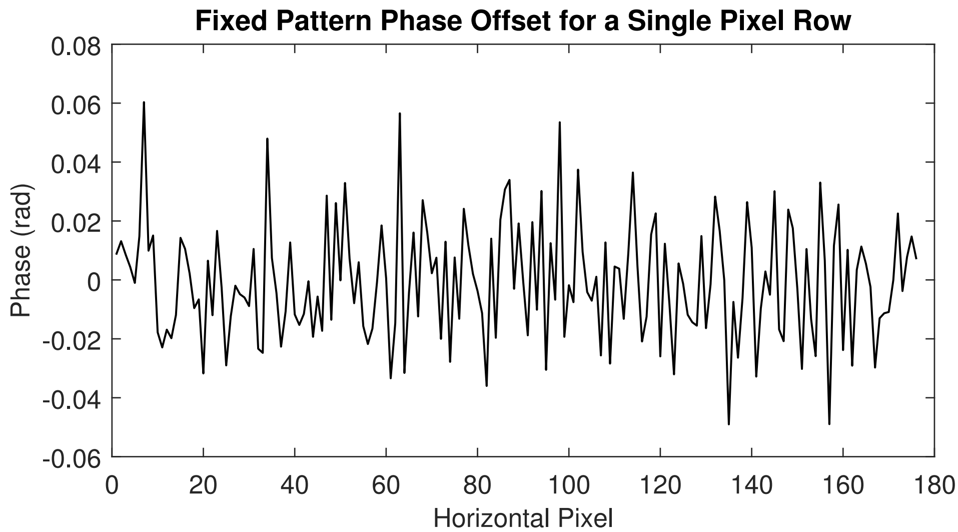


Figure 3.12: Isolating a single row of the fixed pattern phase offset shows no obvious trends or offsets.

absolute maximum of these fixed offsets is 0.25 rad, which relates to approximately 200 mm.

3.3.2 Testing of the phase calibration

An independent set is then acquired for comparing these calibrations to the factory calibrations of the SR4000. A diffuse flat plane is imaged from a number of angles and distances to cover multiple possible scenarios. The captured data is then transformed to Cartesian (x,y,z) coordinates, so that the imaged plane appears as a flat plane. A single row of captured data from a perpendicular plane (Figure 3.14), gives a comparison between the data sets. The raw data is compared to the factory calibrated data from the SR4000, before and after being calibrated using the phase correction techniques above. It can be seen that the raw data varies much more than the calibrated data, compared to the camera's factory calibrated output. This graph shows only the effect of the phase calibrations, namely the harmonic correction, global phase offset, gradual phase offset, and fixed pattern phase offset. This graph is analogous to Figure 2.7, with the addition of data calibrated with the presented techniques.

Quantitatively, the results of the phase calibrations can be analysed by finding the root mean squared error (RMSE) and standard deviation (SD), of the raw data and calibrated data, to the factory calibrated plane. If there are m pixels in each of the images for comparison, then the RMSE is given as,

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(x_i - y_i)^2} \tag{3.4}$$

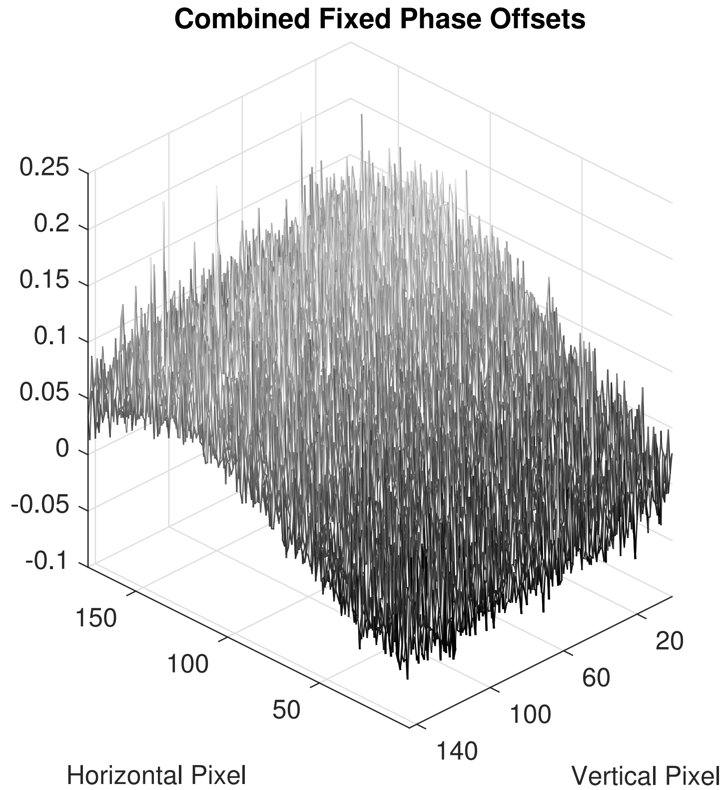


Figure 3.13: The fixed phase offsets (global, gradual, and fixed pattern) are combined into a single calibration image.

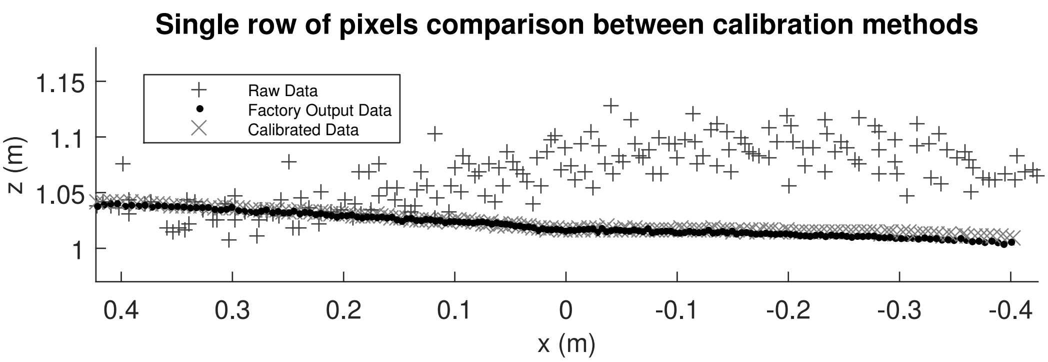


Figure 3.14: Comparison between raw data, factory calibrated data, and data calibrated using the presented techniques (Source: Charleston *et al.* Proc. SPIE, vol. 9528. "Extracting the MESA SR4000 Calibrations": 2015. `http://dx.doi.org/10.1117/12.2183654`. Used with permission).

Table 3.2: RMSE and SD of the phase calibrations, comparing the raw data, can calibrated data to the factory calibrations.

	Raw Data		Calibrated Data	
View	RMSE (m)	SD (m)	RMSE (m)	SD (m)
Perpendicular	0.0626	0.0376	0.0082	0.0022
Left/Right	0.0613	0.0384	0.0049	0.0031
Up/Down	0.0632	0.0378	0.0051	0.0037
Average	0.0624	0.0379	0.0061	0.0030

where x is the reference dataset (factory calibrated data), y is the comparison dataset, and i is the index. The standard deviation is found in a similar way, with

$$SD = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left((x_i - y_i) - (\bar{x} - \bar{y})\right)^2} \tag{3.5}$$

where $\bar{x}$ and $\bar{y}$ are the mean values for the x and y datasets respectively. It can be seen that the SD removes any bias. If the datasets were identical other than some offset, the RMSE would report that offset, while the SD would report a value of zero. If the means of the two datasets are very similar, it can be seen that the second half of SD equation approaches zero, giving the same equation (hence the same value) as the RMSE. These equations both provide useful information, where the RMSE is more important because absolute accuracy is desired, however having a low SD with a high RMSE can identify biases between the datasets.

Table 3.2 shows the results when comparing the factory calibrated data to the raw data, and the calibrated data using the described techniques. It can be seen that the calibrated data is over an order of magnitude closer to the factory calibrated data than the raw data. The SR4000 datasheet (Mesa Imaging, 2011) describes the absolute accuracy of the factory calibrated data to be within ±10 mm. Because the reported RMSE of the calibrated data (6.1 mm) is within this limit, this calibration is deemed to be acceptable for use in the 'Mixed Pixel Restoration Algorithm', and the motion correction algorithm described in Chapter 4.

These data sets can also be compared and analysed graphically. Plotting the raw data and calibrated data directly against the factory calibrated data to obtain a correlation coefficient is not appropriate, mainly because the correlation coefficient measures the strength of the relationship between the two variables, hence a high correlation can still be achieved without a 1:1 gradient (Bland and Altman, 1986). This can be problematic in scenarios where one measurement technique is being directly compared to another. An alternative way to represent and interpret the data is through a Bland-Altman plot (Bland and Altman, 1986), which attempts to convey whether or not two measurement techniques are in agreement, by plotting the difference between the two techniques against their mean.

Bland-Altman plots of the range data are generated using the perpendicular images of the flat plane at a number of distances. The plots have three lines added, being the offset (the mean difference), and the two 95 % intervals (±1.96 SD of the difference). The SD of the difference is also shown at the top right of the plot. The key to a Bland-Altman plot is the interpretation of these values, where it is at the discretion of the user as to whether these values are acceptable for defining agreement between the two techniques. Because the SR4000 datasheet reports an absolute accuracy of ±10 mm, this is a good metric to use for judging the data. The mean difference line should lie inside this region of ±10 mm, and ideally the 95 % intervals would also lie within this region. Any trends with increasing mean (distance from the camera in this case) can also be seen in the Bland-Altman plot.

Because the calibration set is known to be worse toward the edges of the image sensor (likely due to interference from the tube-diffuser setup), the Bland-Altman plot of the raw data against the factory output data (Figure 3.15) has been calculated with a central window of 80×100 pixels. This shows a mean difference of −41 mm, with an SD of 32 mm, while 95 % of the data falls between −100 mm and 22 mm. Based on the fact that the mean difference falls so far out of the ±10 mm range, as well has the 95 % interval being so wide, the raw data does not agree well with the factory calibrated data. A trend with increasing mean distance is also seen, where the apparent mean difference oscillates, following the curve of the harmonic distortion (Figure 3.7).

A similar plot is produced for the data calibrated with the presented techniques against the factory output data (Figure 3.16). The scale of the y-axis (difference) is a factor of 10 smaller than that of the previous plot. This is because the data is an order of magnitude different, hence the need to present Figures 3.15 and 3.16 with different y-axis scales. There is a mean difference of 5.3 mm, with an SD of 2.9 mm. The 95 % interval shows the data falling between −0.3 mm and 11 mm. Because the mean offset falls within the ±10 mm, and the SD is relatively low, it can be said that the calibrated data is in agreement with the factory output data. This means that data calibrated with the presented techniques can be used to replace the factory output data. There is some variation in the data with distance, however it is not obvious as to what cause of this is (most likely due to slight errors in the harmonic distortion correction).

Finally, a combined plot is produced, where both the above plots are combined onto the same scale. Only a single distance image is used, at approximately 1.15 m (Figure 3.17). The 95 % lines have been omitted for simplicity, as well as the mean difference of the calibrated data. This plot shows the magnitude of the difference between the raw data and the factory calibrated data is very large, and on this scale the mean of the calibrated data is near linear. There is still an apparent offset to the calibrated data from 0 m, however because it falls within the ±10 mm tolerance range, further investigation is not required for the applications presented herein.

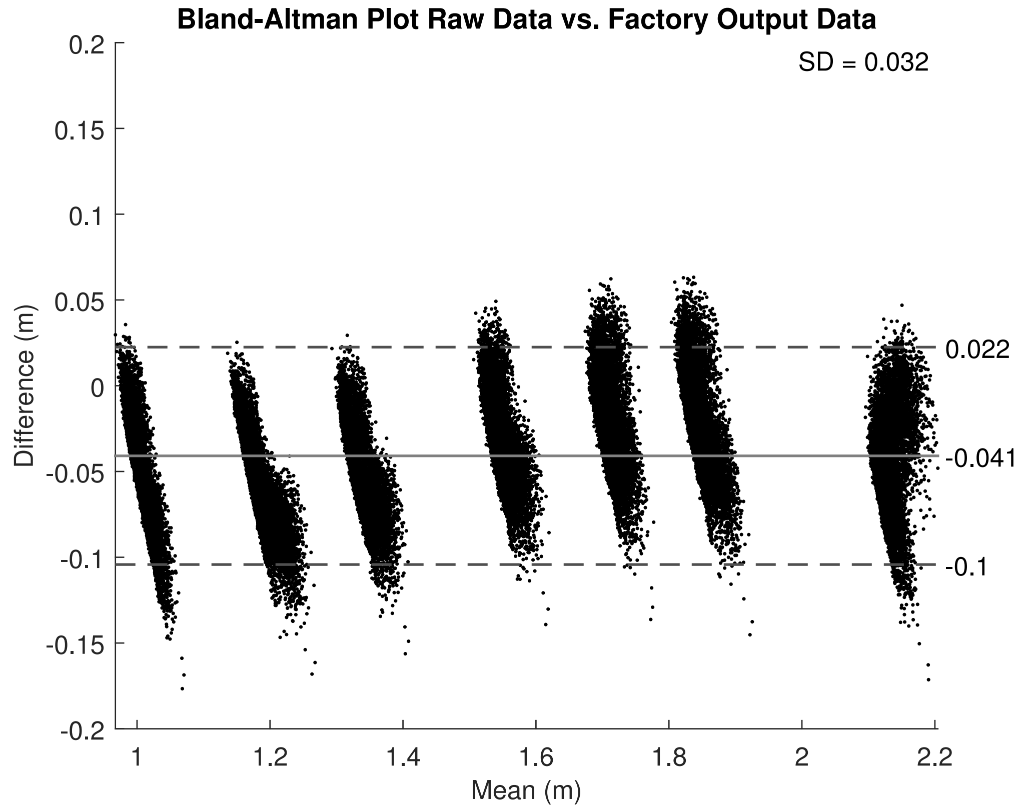


Figure 3.15: Bland-Altman plot of raw data against factory output data. The mean difference is shown with a full line, while the 95 % confidence intervals are shown with dashed lines.

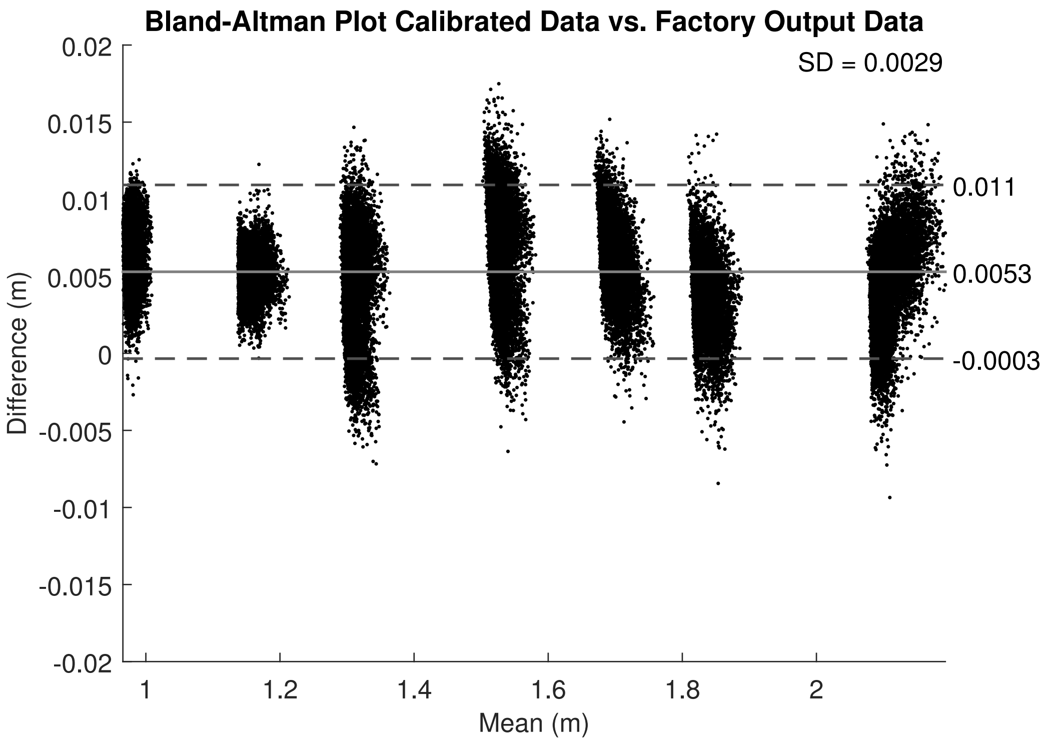


Figure 3.16: Bland-Altman plot of calibrated data against factory output data.

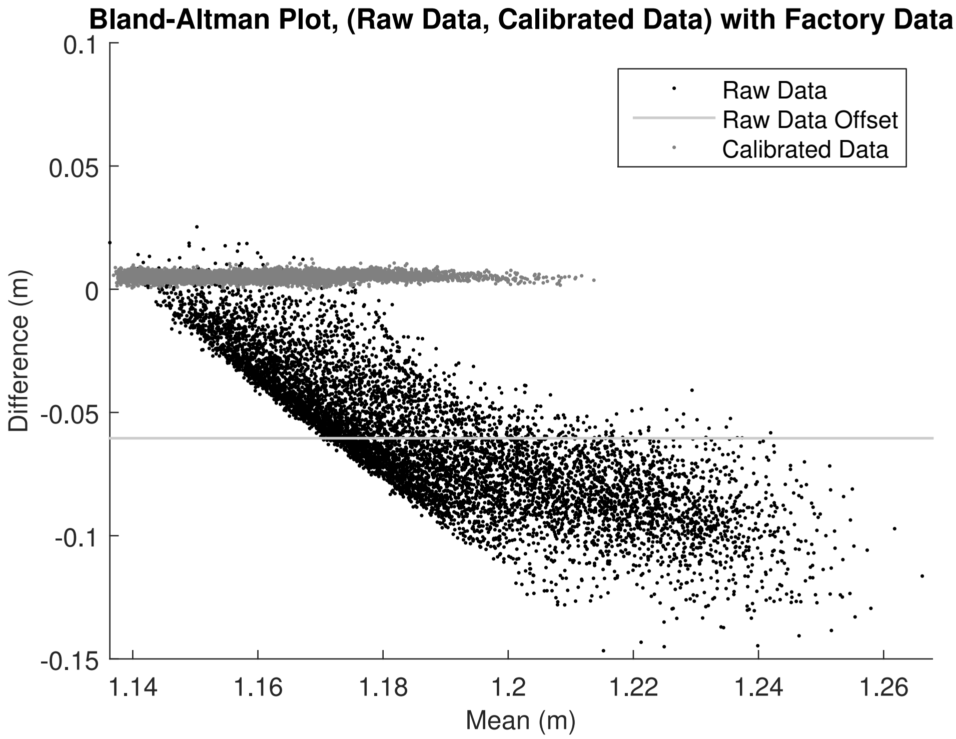


Figure 3.17: Combined Bland-Altman plot with raw data and calibrated data against factory output data.

3.3.3 Amplitude calibrations

The factory amplitude correction in the SR4000 combines both the distance based amplitude correction (γ_ϕ), as well as the radial amplitude correction due to uneven illumination (σ_{jk}). Because the distance based amplitude correction is undesired, the two calibrations are separated. In the application of these calibrations, the raw data is calibrated before and during the demodulation, meaning the retrospective calibration of amplitude based on distance (γ_ϕ) is unnecessary. The radial illumination amplitude pattern obtained in retrieving this calibration is displayed in Figure 3.18. The radial multiplier is approximately one at the centre, and increases with radial distance. The main objective of the overall calibration set is to obtain improved phase images, hence applying these amplitude calibrations after demodulation is not useful for the applications herein. However, this amplitude calibration is useful for the motion correction algorithm in Chapter 4, when applied prior to demodulation.

Because the factory calibrated amplitude output includes the undesired distance dependent calibration (γ_ϕ), comparison of results to the extracted radial amplitude calibration (σ_{jk}) is difficult. The distance dependent calibration is divided from the factory calibrated data, which is then compared to an amplitude image corrected with the extracted calibration. Because the amplitude scale is arbitrary (no relationship is described between the intensity and amplitude values in the SR4000

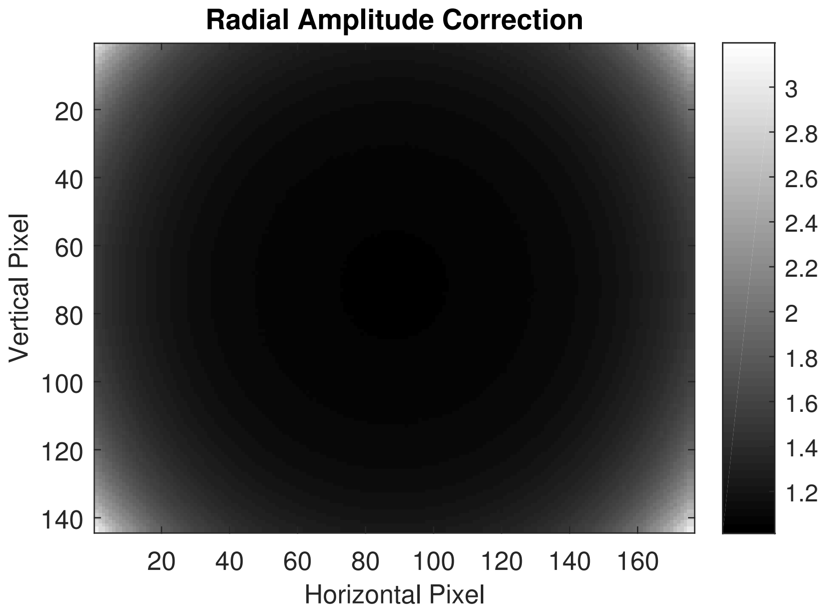


Figure 3.18: Extracted radial amplitude correction multiplier from the SR4000 (Source: Charleston *et al.* Proc. SPIE, vol. 9528. "Extracting the MESA SR4000 Calibrations": 2015. `http://dx.doi.org/10.1117/12.2183654`. Used with permission).

datasheet (Mesa Imaging, 2011)), the ratio of the factory and extracted radial calibrations are taken and normalised (an ideal calibration would yield values of one for all pixels). The mean of this ratio is then found to be 0.994, with a SD of 8.9×10^{-3}, which is essentially negligible meaning that the two amplitude calibrations are near identical.

The SR4000 does not appear to correct the gain fixed pattern amplitude factor (η_{jk}), caused by variation in the amplitude response between pixels. It is very difficult to obtain this factor, as even with the retro-reflector setup, there are still effects caused by uneven illumination (ideally the amplitude of the raw data would be perfectly flat). This is partially a result of the transmitted illumination pattern of the 50° square diffuser (Figure 3.1b). This pattern can be seen in the raw amplitude pattern viewed across a single row of pixels (Figure 3.19). The higher relative intensity edges are visible, where the centre of illumination raises slightly. One solution would be to attempt to model and correct this relative intensity pattern, however the data is not openly accessible from the Thorlabs technical datasheet (Thorlabs, Inc., 2015), hence modelling the illumination profile based on the image alone is challenging. Further work needs to be done to attempt to obtain a more uniform illumination on the sensor in order to retrieve the gain fixed pattern, however it appears likely that for the SR4000, the gain fixed pattern is negligible compared to the radial amplitude correction.

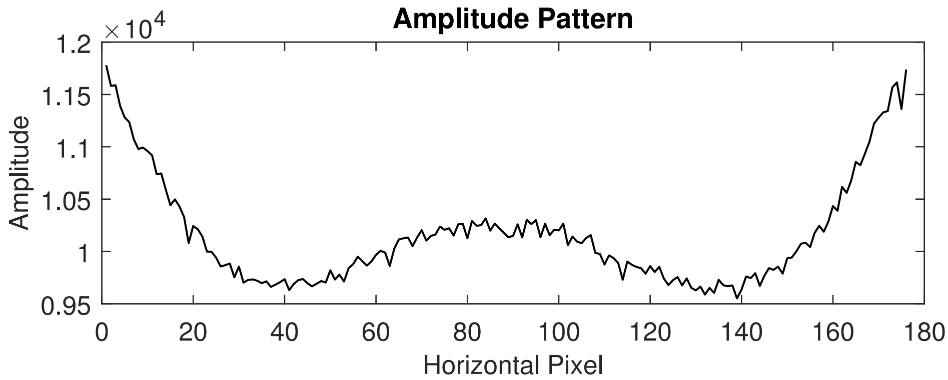


Figure 3.19: Amplitude pattern across a single row of pixels.

3.3.4 Background calibrations

Because the dark current is inherently corrected in the standard demodulation of the signal, the calibration does not exist in the factory calibration set of the SR4000, hence cannot be extracted. This correction is only required in the motion correction algorithm, due to the modification of the demodulation technique (where the background offsets are no longer inherently corrected). This means that the background corrections generated herein are unable to be compared to any of the SR4000 outputs for verification. The dark current (Figure 3.20) is extracted by attempting to cover the lens with a retrofitted lens cap butted to the camera. There does however appear to be a slightly higher intensity toward the centre of the pattern, where there may be some small light leakage onto the sensor from an imperfect experimental setup.

The other background correction is the amplitude dependent background offset. This calibration is found by plotting the amplitude versus the background light level (with the dark current subtracted) for each pixel. The slope for each pixel is then fitted, where the gradients (Figure 3.21) and intercepts (Figure 3.22) are stored in lookup tables. During calibration, the amount of background correction is found based on the amplitude of each pixel. As with the dark current offset, the results of this calibration are difficult to quantify. To give an idea of the amount of variation in this calibration, both the gradients and intercepts are normalised within each respective image. After normalising the gradients, they ranged from -0.57 to 1, with a mean of 0.214 and an SD of 0.209. Similarly, the normalised range of intercepts of these lines is from -2.87 to 1, with a mean of -0.09 and an SD of 0.17. The mean intercept value is close to 0, hence the intercepts have less impact compared to gradients. Plotting a small number of pixels (Figure 3.23) shows that the slopes are near linear, hence fitting the pixels with straight lines is acceptable.

3.3.5 Photogrammetric calibration and lens distortion

The final step in the calibration is the transformation between radial coordinates and Cartesian coordinates. As mentioned above, the SR4000 has a built in transforma-

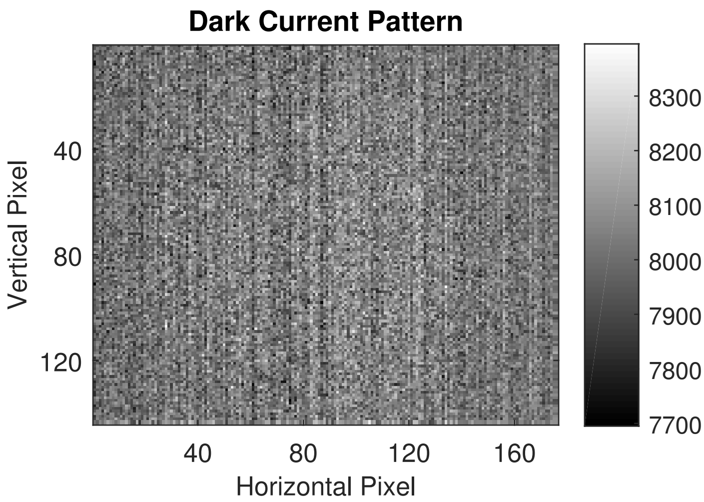


Figure 3.20: Dark current calibration pattern.

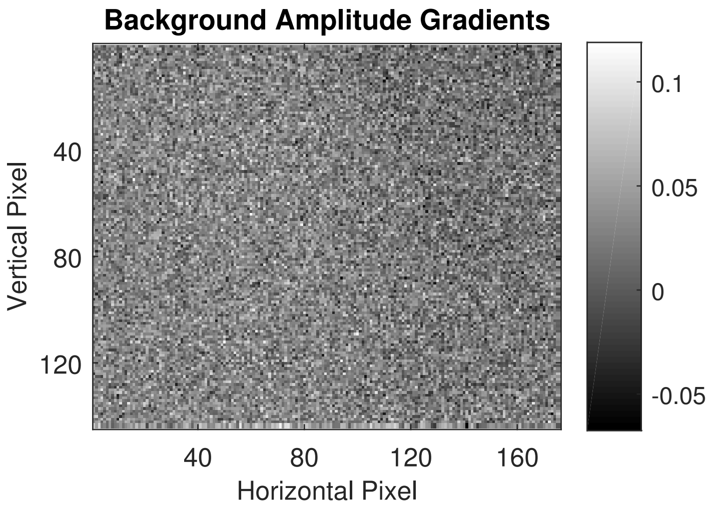


Figure 3.21: Amplitude dependent background gradients.

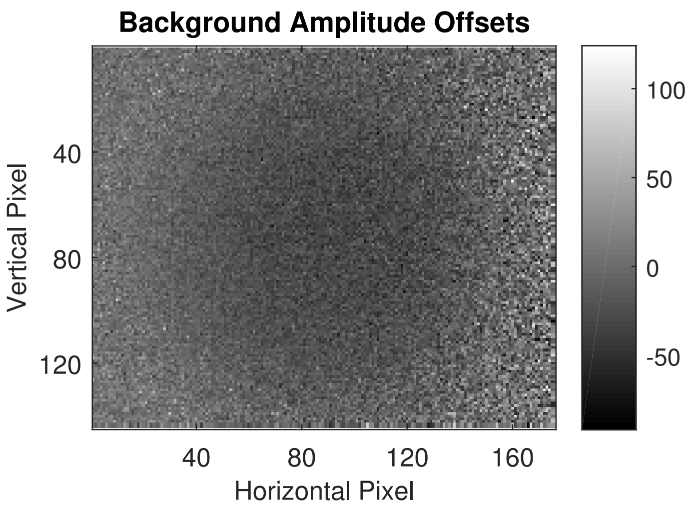


Figure 3.22: Amplitude dependent background offsets.

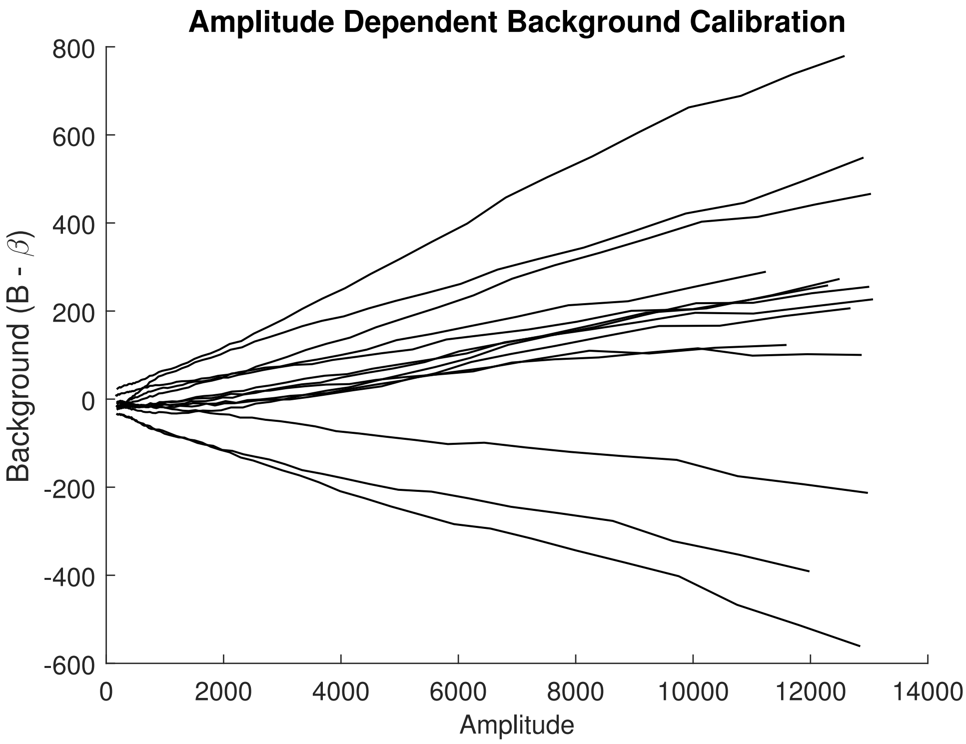


Figure 3.23: Amplitude dependent background calibration for a number of pixels. Each line represents the calibration for a different pixel.

tion function, or alternatively the method presented by Dorrington *et al.* (2011) can be used. Here it is attempted to retrieve the matrix of intrinsic parameters (K), as well as the lens distortion coefficients (k_1, k_2, p_1, and p_2) such that transformation techniques described in Section 2.3 can be manually applied if desired.

Using the example code provided by Mesa Imaging[3] with C++ and OpenCV, the calibration matrix K was extracted as,

$$K = \begin{bmatrix} 250.40 & 0 & 88.58 \\ 0 & 250.41 & 72.20 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.6}$$

with associated lens distortion coefficients of $k_1 = 12.65$, $k_2 = -10.4$, $p_1 = 6.6 \times 10^{-6}$, and $p_2 = 3.1 \times 10^{-5}$. These lens distortion coefficients can be used to plot a distortion model, similar to that of Figure 2.12, where the effects of both the radial and tangential distortion can be seen (Figure 3.24).

Because Mesa Imaging supply key information on the data sheet for the SR4000 (Mesa Imaging, 2011), the intrinsic parameter matrix can be estimated. The focal length is given as 10 mm, and the pixel pitch as 40 µm in both the horizontal and

[3]http://forum.mesa-imaging.ch/viewtopic.php?f=33&t=169: accessed 11/06/2015.

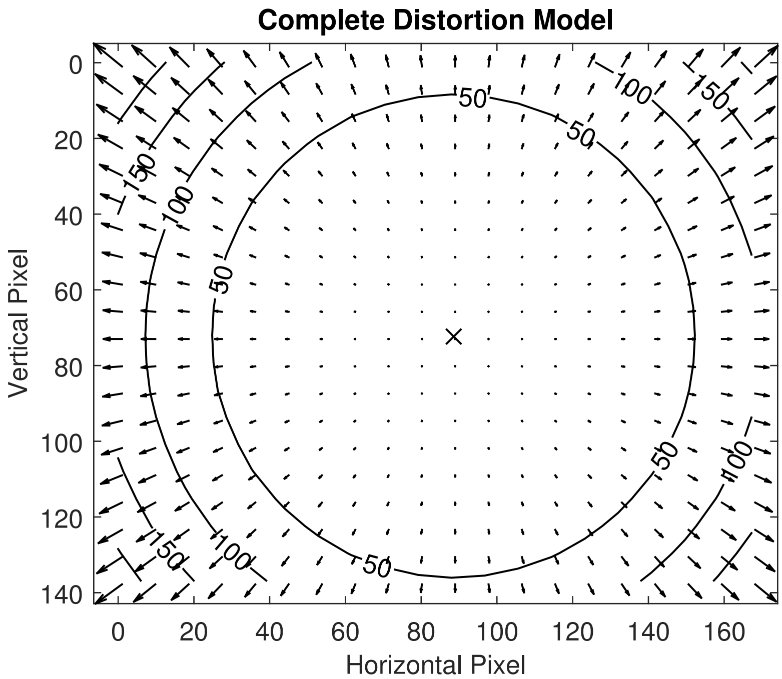


Figure 3.24: SR4000 distortion model. The arrows show the pixel correction mapping.

vertical directions. This results in

$$\frac{f}{s} = \frac{10\,\mathrm{mm}}{40\,\mu\mathrm{m}} = 250 \tag{3.7}$$

The principle point is given as the central pixel which is given as half of the pixel array size of 176×144, giving u_0 as 88, and v_0 as 72. These values are very similar to those found above using the code from MESA Imaging using C++ and OpenCV.

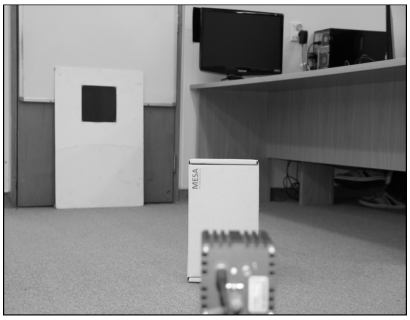
Finally, the intrinsic parameters as well as the lens distortion can be estimated using the technique from Zhang (2000), along with the camera calibration toolbox in MATLAB. A series of seven images were taken of an 8×8 checkerboard, with squares of size 30 mm. An example of one of these calibration images (Figure 3.25) shows the configuration of the checkerboard setup. The lens distortion can be clearly seen on the checkerboard on the left edge of the image. The amplitude image of the factory calibrated data was used for the calibration. The reported focal lengths from this calibration were reported as $\frac{f}{s_x} = 247$ and $\frac{f}{s_y} = 235$, while the principle point was reported as $u_0 = 87.2$, and $v_0 = 75.9$. The distortion coefficients were reported as $k_1 = -0.7$, $k_2 = 0.22$, $p_1 = -0.0066$, and $p_2 = 0$. Although the intrinsic parameters are relatively accurate, the distortion coefficients are not. A better image calibration set covering more of the image space might improve these results.

Figure 3.25: Example image of the photogrammetric calibration using the technique presented by Zhang (2000).

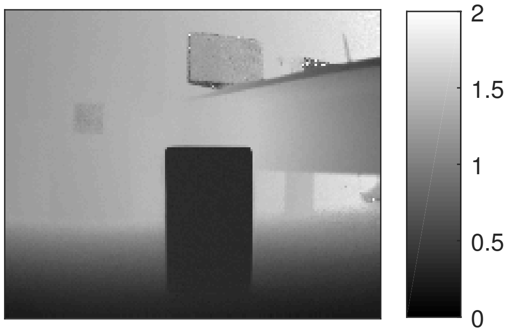
3.3.6 Applications

The main purpose for obtaining these calibrations is for the motion correction algorithm presented in Chapter 4. These calibrations are useful for any algorithm which either: obtains data captured in 'raw mode' where the calibration set is lost; or algorithms that require capturing multiple frequencies (where one or more frequencies require calibration), such as the Mixed Pixel Restoration Algorithm (MPRA) (Dorrington *et al.*, 2011), which looks to correct multipath interference. The SR4000 is able to operate at multiple modulation frequencies, however only the 30 MHz frequency is calibrated. This calibration technique can be modified to be applied at different frequencies, such as at the 15 MHz frequency the SR4000 is able to operate at. The 30 MHz factory calibrated phase data can be halved, and the 15 MHz raw phase data can then be calibrated from this. The results of this calibration are not shown, as there is no noticeable difference to the 30 MHz calibration other than the phase values being halved.

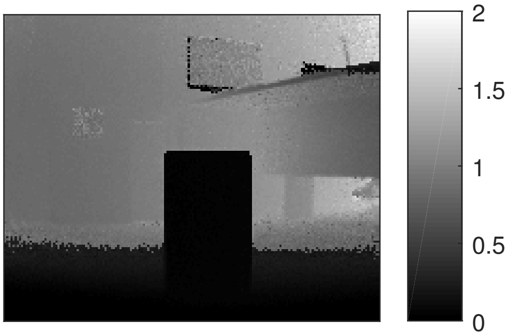
A scene was developed to promote lens scattering (Figure 3.26a). A black square is placed on a background wall, which has low intensity light returning to the camera, thus is susceptible to multipath interference. A white box is placed in the foreground of the scene, which has very intense return signals and can cause multipath interference, especially to low intensity signals like the black square. The background wall should be flat in the camera's factory output view of the phase image of the scene (Figure 3.26b), however because of the lens scattering, the black square appears closer to the camera. Applying the MPRA without calibrating the second frequency (Figure 3.26c), it can be seen that while the signal from the black square has been improved, the scene in general has become much more noisy, and there is a large phase jump on the carpet. Applying the calibrations found above to the 15 MHz MPRA (Figure 3.26d) shows a much less noisy signal return. Although there is still a slight drag forward of the black square, it is much less than that of the factory output data.



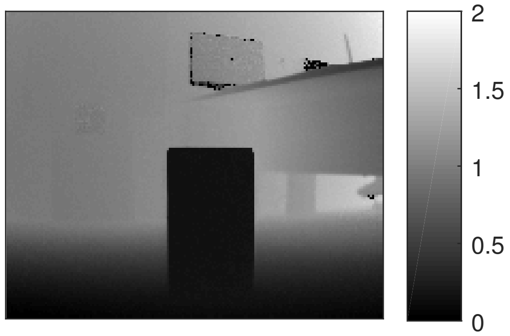
(a) Conventional 2D camera. The SR4000 can be seen in the foreground



(b) Factory calibrated output.



(c) MPRA uncalibrated.



(d) MPRA calibrated.

Figure 3.26: Lens scattering scene before and after use of the 'Mixed Pixel Restoration Algorithm' (MPRA).

Chapter 4

Motion Correction

As discussed in Section 2.5, motion blur is a serious issue in dynamic time-of-flight scenes. Numerous techniques have been presented to correct for this motion blur, however the technique presented by Hussmann *et al.* (2011) is further investigated herein, which looks to correct for blur from motion with a constant velocity, by using the first phase step (τ_1) as a reference, and aligning the subsequent phase steps (τ_2, τ_3, and τ_4). The number of pixels each phase step is shifted is initially estimated using the binary image technique presented by Hussmann *et al.* (2011). A vector is then generated to store the amount of pixels to shift each phase step, $\mathbf{p}_{shift} = [s_1\ s_2\ s_3\ s_4]$, where s_n defines the number of respective pixels each phase step τ_n is shifted. This technique assumes that the motion is purely tangential, and intra-frame blur is negligible. It is also important to note that because of the shifting of the phase steps, information is lost at the edges of the demodulated phase and amplitude images in the direction of motion. For an object that moves a total distance of m pixels across the four phase steps, a total of $2m$ pixels are lost.

This chapter aims to modify the motion correction technique presented by Hussmann *et al.* (2011), with the intention to improve the technique for a more robust motion correction algorithm. The key problem with the technique in its current state is in calibration, where only the harmonic distortion, dark current, and amplitude gain calibrations are performed. The work presented in this chapter includes all of the offsets in the expanded phase step equation (Equation 2.17). To achieve this, the phase steps must be corrected for further background and amplitude corrections. The demodulation technique is then modified to include the additional phase calibrations.

A number of experiments are performed, which look to firstly quantify the effects of motion in time-of-flight cameras, followed by the improvements applying the technique from Hussmann *et al.* (2011), and finally the proposed improvements on the technique. Initially, 'simulated motion' is used for testing, where the object is not moving in real time. The translation stage is set to a fixed position, where the first phase step is captured. The translation stage is then moved a fixed distance, and the second phase step is captured. This is repeated for all four phase steps, after which they are demodulated. The advantage of this technique is that it gives the impression of motion, without having any intra-frame motion blur. The image can also be averaged at each location for increased noise reduction during initial testing.

The final part of this chapter investigates software simulated motion, where a time-of-flight camera is generated in software, in a similar motion affected scenario as the practical experiment. This simulation is used for comparison to the practical results, to aid in analysing the effect of each error source. The error sources are reverse engineered to be applied to the simulated time-of-flight camera, after the motion artefacts are added.

Because time-of-flight cameras measure distance radially from the camera, there is an issue when the direction of motion is along a single Cartesian axis (with the exception of the optical z axis). If an object remains constant in the Cartesian z direction, moving only in positive fixed increments of the x direction , it can be seen that the angle (θ) between the increments decreases with distance from the optical axis, while the radial distance r increases (Figure 4.1).

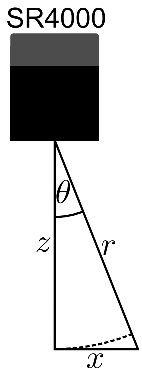


Figure 4.1: Relationship between radial and Cartesian coordinates. The dashed line shows a constant radius of length z.

Assuming the motion is purely in the horizontal direction of the field of view, the severity of this coordinate difference can be calculated. The SR4000 has a field of view of 43°, or 0.75 rad (Mesa Imaging, 2011), giving $\theta = 0.375\,\text{rad}$. The SR4000 has 176 horizontal pixels (px), giving

$$\frac{0.375\,\text{rad}}{88\ \text{px}} = 0.0043\frac{\text{rad}}{\text{px}}. \tag{4.1}$$

Using a fixed operating distance of $z = 0.5\,\text{m}$, the maximum distance from the optical axis in the positive x direction is found as

$$x_{max} = (0.5\,\text{m})\tan(0.375) = 0.197\,\text{m}. \tag{4.2}$$

Using fixed increments of 0.01 m along the x axis, the difference in angles can be found at the optical axis, as well as at x_{max}. At the optical axis, a difference in angle of 0.02 rad is seen, which corresponds to a width of 4.7 px. At x_{max}, a difference in angle of 0.174 rad is seen, relating to a width of 4.0 px. Because this difference is relatively small (less than one pixel), as well as the fact that the motion is often over a small range, this effect is assumed to be negligible.

4.1 Demodulation

The motion correction technique presented in this section relies on use of the knowledge of the direction and speed of motion. Because this work is targeted at industrial applications such as fruit grading lines, the velocity is known with high accuracy. In a more general case however, the velocity of the motion is unknown. Automated algorithms such as optical flow (Horn and Schunck, 1981) could be used to estimate this, as is discussed by Streeter and Dorrington (2014). The motion detection technique presented by Hussmann *et al.* (2011), where binary images are found from phase step differences is feasible when the velocity is constant. In this section, with the knowledge of the speed and direction of motion, a novel demodulation algorithm is presented, which modifies Equation 2.3.

In order to correctly calibrate for motion correction, the expanded phase step equation (Equation 2.17) needs to be analysed, which is repeated here for reference,

$$\tau_{jkn} = \eta_{jk}\sigma_{jk}A_{jk}\cos(\phi_{jk} + \theta_n + \delta_{jk} + \rho_\phi) + (\psi_{\alpha jk} + \beta_{jk}). \tag{4.3}$$

Using Euler's formula, this equation can be rewritten as a complex exponential function,

$$\tau_{jkn} = \frac{\eta_{jk}\sigma_{jk}A_{jk}}{2}\left(e^{i\phi_{jk}}e^{i\theta_n}e^{i\delta_{jk}}e^{\rho_\phi} + e^{-i\phi_{jk}}e^{-i\theta_n}e^{-i\delta_{jk}}e^{-i\rho_\phi}\right) + (\psi_{\alpha jk} + \beta_{jk}). \tag{4.4}$$

Using the standard demodulation technique (Equation 2.3), it can be seen that the background portion ($\psi + \beta$) of the equation sums to zero, cancelling out during demodulation. It can also be seen that the phase (δ, ρ) and amplitude (η, σ) offsets can be removed after demodulation, where the complex phasor is separated into its phase and amplitude components, and the phase and amplitude calibrations can be applied to each respective component.

When implementing the technique from Hussmann *et al.* (2011), each phase step is shifted prior to demodulation. As a result, pixels from different phase steps are no longer correctly aligned, meaning that the background, phase, and amplitude calibrations that are dependent on pixel location (jk) are no longer valid. The result is that caution needs to be taken when applying motion correction, so that the calibrations are applied correctly.

4.1.1 Background calibration

The first part of each phase step to correct is the background, because it is no longer cancelled out during demodulation due to its dependence on pixel location. It is also corrected before the amplitude and phase, because it is additive to each phase step. The dark current calibration (β_{jk}) is first removed simply by subtraction off each phase step. The amplitude dependent offset ($\psi_{\alpha jk}$) is more complicated, because the amplitude is unknown until the four phase steps are demodulated. Similarly, without calibrating the amplitude dependent offset, the correct amplitude is unable to be found, resulting in an under-determined system.

In the simulated motion case, this calibration can be found and applied correctly. This is because four phase steps can be taken at each position, where only the respective phase step of each position is used in demodulation. From this, the amplitude at each position can be found and used to calibrate each respective phase step for the amplitude dependent offset before demodulation.

It is a much more challenging prospect to apply this calibration in a real-time case, due to only having a single phase step at each position. For the full speed motion corrections attempted in this thesis, a pre-determined amplitude image of the object is used for the calibration. The location of the object in each phase step is found, and the amplitude image is aligned with the phase step. The amplitude dependent background offset is then found and subtracted from the phase step. This technique is not practical in situations where the shape of the object is unknown before the motion correction. Issues can also arise where there can be differences in both the object's light reflectance, as well as the pose of object relative to the camera location. These are assumed to be negligible for this work.

Future work would look to improve on the derivation of this amplitude related background offset. An estimate could potentially be found with some constraints on the background, using the phase step to provide an amplitude estimate. Other techniques could attempt to further investigate the demodulation technique, to manipulate it in a similar fashion as discussed below in Section 4.1.3, with the aim of including a term for improved background correction.

After background subtraction, the remaining phase step equation is,

$$\tau_{jkn} = \eta_{jk}\sigma_{jk}A_{jk}\cos(\phi_{jk} + \theta_n + \delta_{jk} + \rho_\phi). \tag{4.5}$$

4.1.2 Amplitude calibration

Once the background has been subtracted from each phase step, the amplitude can be corrected. The amplitude corrections are divided from each remaining phase step. As discussed in Section 3.3.3, the gain amplitude multiplier (η_{jk}) is negligible in the case of the SR4000. For this reason it is set to 1 in Equation 4.5. The radial amplitude correction (σ_{jk}) can then be simply divided from each phase step. After the amplitude calibrations have been applied, the remaining phase step formula is,

$$\tau_{jkn} = A_{jk}\cos(\phi_{jk} + \theta_n + \delta_{jk} + \rho_\phi). \tag{4.6}$$

4.1.3 Phase calibration

The only remaining offsets are from phase errors. Because the harmonic distortion (ρ_ϕ) does not depend on the pixel position, it can be subtracted after demodulation, hence is removed from Equation 4.6. Each pixel in the remaining fixed phase offset (δ_{jk}) has specific properties associated with it, giving each pixel a unique phase calibration. When the phase steps are shifted onto one another and demodulated, the phase calibration becomes invalidated. This is clear when considering the fact that the Fourier transform of the combination of the four phase steps is required to obtain the phase image. If the intensity values of differing pixels in different phase steps are shifted onto each other, the phase calibration will be invalidated. The result is that the fixed phase offset is now dependent on the phase step (δ_{jkn}).

Because the fixed phase offset term is within the cosine of Equation 4.6, it is unable to be removed through subtraction or division from the remaining phase steps. This means that it can only be removed after demodulation. Re-writing Equation 4.6 in exponential form after removing the harmonic distortion term gives,

$$\tau_{jkn} = \frac{A_{jk}}{2}\left(e^{i\phi_{jk}}e^{i\theta_n}e^{i\delta_{jkn}} + e^{-i\phi_{jk}}e^{-i\theta_n}e^{-i\delta_{jkn}}\right). \tag{4.7}$$

Attempting to demodulate this using the regular demodulation technique (Equation 2.3) gives,

$$\sum_{n=1}^{4}\tau_{jkn}e^{-i\theta_n} = 2A_{jk}e^{i\phi_{jk}}e^{i\delta_{jk}} + \sum_{n=1}^{4}\frac{A}{2}e^{i\phi_{jk}}e^{i\delta_{jkn}}e^{\theta_n}. \tag{4.8}$$

In this case, the second half of the equation does not sum to 0, resulting in invalid phase and amplitude images after demodulation. With the knowledge of the number of pixels each phase step needs to be shifted, an additional phase offset term ($e^{i\delta_{jkn}}$) can be multiplied to Equation 4.7, resulting in

$$\begin{aligned}\tau_{jkn}e^{i\delta_{jkn}} &= \frac{A_{jk}}{2}\left(e^{i\phi_{jk}}e^{i\theta_n}e^{i\delta_{jkn}} + e^{-i\phi_{jk}}e^{-i\theta_n}e^{-i\delta_{jkn}}\right)e^{i\delta_{jkn}}\\ &= \frac{A_{jk}}{2}\left(e^{i\phi_{jk}}e^{i\theta_n}e^{i2\delta_{jkn}} + e^{-i\phi_{jk}}e^{-i\theta_n}\right).\end{aligned} \tag{4.9}$$

This allows the second half of the equation to sum to 0, resulting in

$$\begin{aligned}
\sum_{n=1}^{4} \tau_{jkn} e^{-i\theta_n} e^{i\delta_{jkn}} &= \frac{A_{jk}}{2} \sum_{n=1}^{4} e^{i\phi_{jk}} e^{i\theta_n} e^{i\delta_{jkn}} e^{-i\theta_n} e^{i\delta_{jkn}} \\
&\quad + \frac{A_{jk}}{2} \sum_{n=1}^{4} e^{-i\phi_{jk}} e^{-i\theta_n} e^{-i\delta_{jkn}} e^{-i\theta_n} e^{i\delta_{jkn}} \\
&= 2A_{jk} e^{i\phi_{jk}} \sum_{n=1}^{4} e^{i2\delta_{jkn}} + 2A_{jk} e^{-i\phi_{jk}} \sum_{n=1}^{4} e^{-i2\theta_n} \\
&= 2A_{jk} e^{i\phi_{jk}} \sum_{n=1}^{4} e^{i2\delta_{jkn}}. \qquad (4.10)
\end{aligned}$$

This can be rewritten as,

$$\begin{aligned}
\sum_{n=1}^{4} \tau_{jkn} e^{-i\theta_n} e^{i\delta_{jkn}} &= 2A_{jk} e^{i\phi_{jk}} \Delta \\
&= \alpha_{jk} e^{i\phi_{jk}} \Delta, \qquad (4.11)
\end{aligned}$$

where $\Delta = \sum_{n=1}^{4} e^{i2\delta_{jkn}}$. Because the number of pixels each phase step is shifted is known, Δ can be found and divided from Equation 4.11 after demodulation. The phase (ϕ) and amplitude (α) can then be found. The phase image is then calibrated for harmonic distortion.

4.1.4 Interpolation

In general, when an object moves between each phase step, the number of pixels the object has moved is never an integer amount. For example, the object may move 4.3 pixels per phase step, giving non-integer pixel shifts for re-alignment. This means that when applying the motion correction algorithm, attempting to shift each phase step by an integer amount of pixels (as with the binary image technique from Hussmann *et al.* (2011)) is inaccurate. This becomes less of an issue with increasing camera resolution, however with the SR4000 having such a low horizontal resolution of 176 pixels, it needs to be considered (each pixel covers approximately 2.5 mm at a distance of 0.5 m).

The solution to the problem comes through interpolation, where the number of pixels each phase step is shifted is interpolated, in order for the four phase steps to be more accurately aligned. Interpolation works by generating new values between existing data points. In the most simple case, a linear fit is produced between adjacent data points, and the new values are generated from any point along this line (Figure 4.2). More complicated interpolation considers more data points generating polynomial or spline fits to the existing data points to generate the new values. Here, linear interpolation is performed on each phase step, after being calibrated for background and amplitude errors. Because the phase offset is not calibrated until after the modified demodulation is performed, the phase offset also becomes interpolated. As a result, the additional phase offset that is added to the phase step equation (Equation 4.9) must also be interpolated such that it matches the phase offset arising from the demodulation.

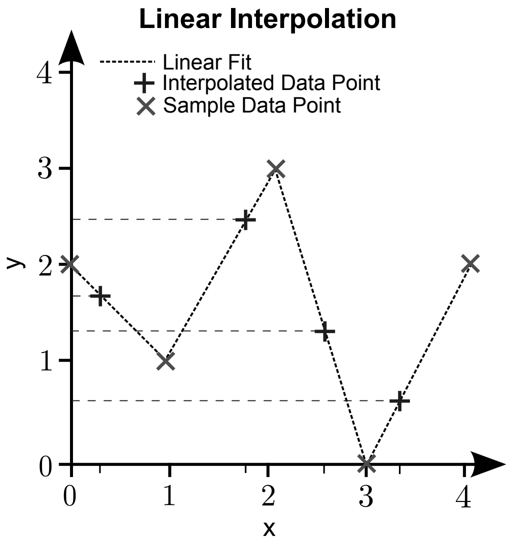


Figure 4.2: Linear interpolation example. The original data points (crosses) are linearly interpolated, shown by the dashed line. New data points are generated using this interpolated line (pluses).

Before interpolation, the amount of sub-pixel motion needs to be estimated. In this case, it is assumed that the amount of motion between phase steps is equal. Each phase step moves an integer number of pixels, as well as having an additional non-integer amount of motion between zero and one pixels exclusive for the interpolation case (additional motion of exactly zero or one pixels would require no interpolation). An initial estimate of the motion is found using the binary image technique from Hussmann *et al.* (2011) described above. Because the initial experiments are not time sensitive, a brute force approach is taken based on this initial motion estimate to find the optimum number of pixels to shift. A range of ± 1 pixel is covered for interpolation, where the RMSE is recalculated at 0.1 pixel increments to find the lowest RMSE value. The amount of interpolation that yielded the lowest RMSE is then used for the final motion correction output.

In a real-time motion setup, future work could look to synchronise the frame rate of the camera with the amount of motion (where the velocity of motion is known beforehand). This would allow captures that are very close to integer shifts for each phase step, removing the necessity for interpolation and reducing the processing time.

4.2 Experimental methods

A number of experiments are conducted to analyse both the algorithm presented by Hussmann *et al.* (2011), as well as the proposed calibration improvements to the motion correction algorithm. A simplistic case with a polystyrene sphere of radius 50 mm is first investigated. The sphere is coated with diffuse white paint, to attempt to provide a strong signal return without saturating the camera. The 'simulated motion' technique is first investigated, where the addition of each calibration to the motion correction algorithm is analysed in detail. The sphere is placed on a lateral translation stage, at a distance of 0.5 m from the camera (Figure 4.3). A factory calibrated capture is taken at the initial position (p_1), as well as a raw data capture, giving the four phase steps acquired in 'raw mode'. The translation stage is then moved in 0.01 m increments, corresponding to approximately 4 pixels per phase step. Further raw data is captured at each of the subsequent positions (p_2, p_3, and p_4), and 100 repetitions are taken at each position, initially for averaging for noise reduction, then for performing statistical analyses. The integration time is also optimised to provide strong signal returns without saturating the sensor, giving a best case scenario in correcting the inter-frame motion blur.

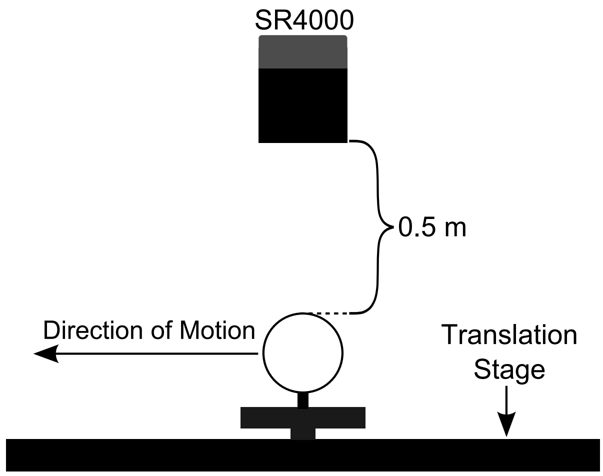


Figure 4.3: Motion correction experimental setup. A diffuse white sphere is placed on a lateral translation stage, 0.5 m from the camera. Images are then captured as the sphere moves horizontally across the field of view of the camera.

The proposed motion correction technique and calibrations discussed in Section 4.1 are then applied to the captured data. The factory calibrated data acquired at p_1 is used as a still image reference. Motion affected data across the four positions is then analysed, using data acquired in the camera's factory calibrated state as a starting point for motion correction. The algorithm described by Hussmann *et al.* (2011) is then applied, followed by the calibration improvements discussed above. Statistical comparisons are then performed to compare the different motion correction algorithms. A similar experiment is performed, to analyse the effects of reducing the both the integration time and the amount of averaging at each position.

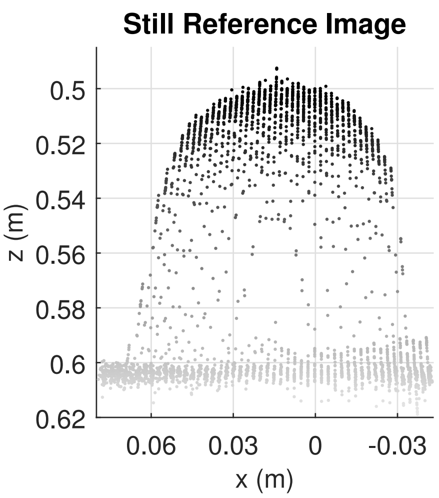
The same experiment is then repeated, however this time with the sphere moving in real-time. The camera is set to acquire raw data as fast as possible at a fixed position on the translation stage. Three experiments are performed, with the translation stage moving at $0.5\,\mathrm{m\,s^{-1}}$, $1.0\,\mathrm{m\,s^{-1}}$, and $1.5\,\mathrm{m\,s^{-1}}$.

Finally, the experiment is repeated with an apple, using industrial fruit grading specifications (see Appendix A for the full specification list). The apple distance is kept at 0.5 m, and the translation stage speed is set to $1.0\,\mathrm{m\,s^{-1}}$.

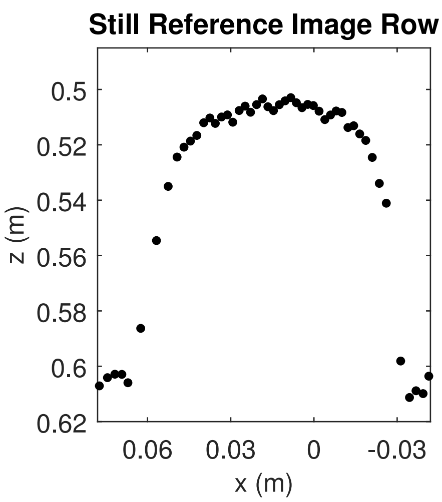
4.3 Results and discussion

4.3.1 Simulated motion

Using the 'simulated motion' setup with the 50 mm polystyrene sphere 0.5 m from the camera, a still reference image was first captured at p_1. The side elevation view of the sphere (Figure 4.4a) shows the top half of the sphere starting at 0.5 m, as well as the platform of the translation stage at approximately 0.6 m. A single pixel row across the centre of the sphere (Figure 4.4b) shows its cross section. It can be seen that the still reference image closely approximates the sphere. There is a small spike at the top of the 3D side elevation, likely due to slight saturation from specular reflection at the perpendicular sphere surface. There are also a number of floating pixels around the edges of the sphere, where the platform and the edge of the sphere overlap onto common pixels. No attempts are made in this work to correct floating pixels, and multipath interference is minimised by reducing scattering surfaces in the surrounding scene.



(a) 3D side elevation of the stationary reference image.



(b) Single image row side elevation of the stationary reference image.

Figure 4.4: Stationary reference image of the 50 mm radius sphere. A side elevation is shown, with a full 3D image, as well as taking a cross section across a single row.

One phase step is then taken from each position (p_1, …, p_4), and combined to produce a motion affected image. Without applying motion correction, the output from the camera in its factory calibrated off-the-shelf state is found (Figure 4.5). This data can be compared to the still image by taking the RMSE between the two, which is 75.9 mm, with a standard deviation of 75.5 mm. The spherical shape cannot be seen in this image, demonstrating that the camera is unusable for motion affected scenes in its off-the-shelf state. There is some detectable structure about $x = 0$ m, where the spherical shape has been severely distorted in the direction of motion.

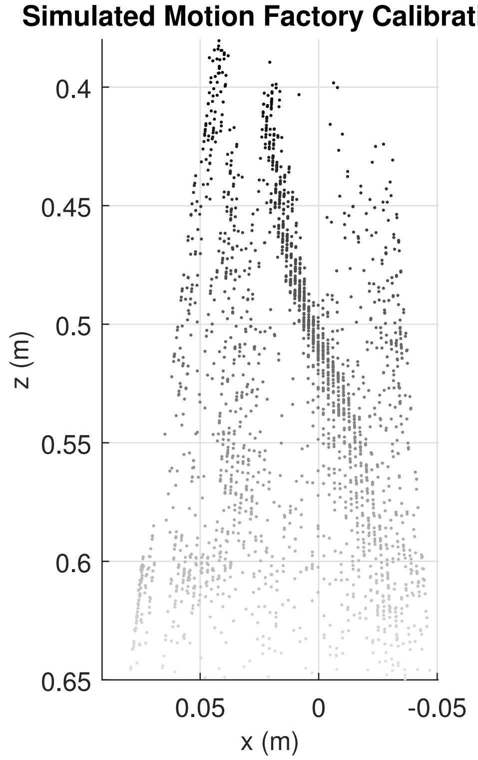


Figure 4.5: Simulated motion factory calibrated output. A 3D side elevation view of the 50 mm radius sphere under motion shows that the spherical shape is indistinguishable.

The technique described by Hussmann *et al.* (2011) is then implemented. This is done by initially producing binary images between the first phase step and subsequent phase steps. An example binary image (Figure 4.6) from the difference of the second and first phase step ($\tau_2 - \tau_1$) clearly shows the motion effects on either side of the translation stage platform. A window around the area of interest has been extracted from the image for display purposes. In this example, it can be seen that there are approximately four pixels over the threshold. In this case, the second phase step (τ_2) would be shifted back four pixels, to align with the first phase step (τ_1). This process is then repeated for the remaining two phase steps, after which all phase steps should be aligned. In this case, each phase step has moved approximately four pixels farther than the previous, giving a pixel shift vector of $\mathbf{p}_{shift} = [0\ 4\ 8\ 12]$, where the first phase step is not shifted.

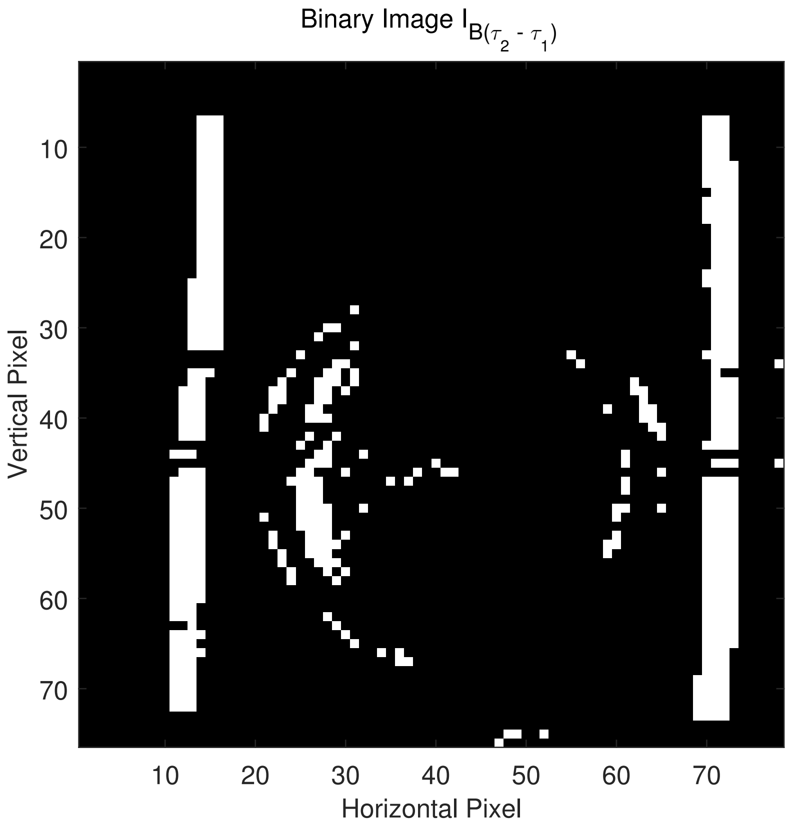


Figure 4.6: Binary image example from the technique presented by Hussmann *et al.* (2011). The first phase step has been subtracted from the second phase step, and a threshold applied. The resulting binary image shows the approximate number of pixels the object has moved between the phase steps.

After subtracting the dark current, the image is demodulated using the standard demodulation technique (Equation 2.3). The complex phasor is then separated into the phase and amplitude components, where the harmonic distortion is calibrated. The naively demodulated and calibrated phase image using the technique from Hussmann *et al.* (2011) (Figure 4.7) shows the sphere to have much more structure than the factory calibrated image without motion correction (Figure 4.5). When compared to the still image, the naive correction showed an RMSE of 58.7 mm, with an SD of 15.1 mm. Although this correction has only a slightly improved RMSE compared to the factory calibrated data without motion correction (Table 4.1), the SD is much lower.

Finally, the additional calibrations to the motion correction algorithm described above are implemented. The dark current and amplitude dependent background offset are first subtracted from each phase step. An example of the amplitude dependent background offset for the first phase step (Figure 4.8) shows the amount of variation over the sphere object. The edges of the sphere have lower amplitude,

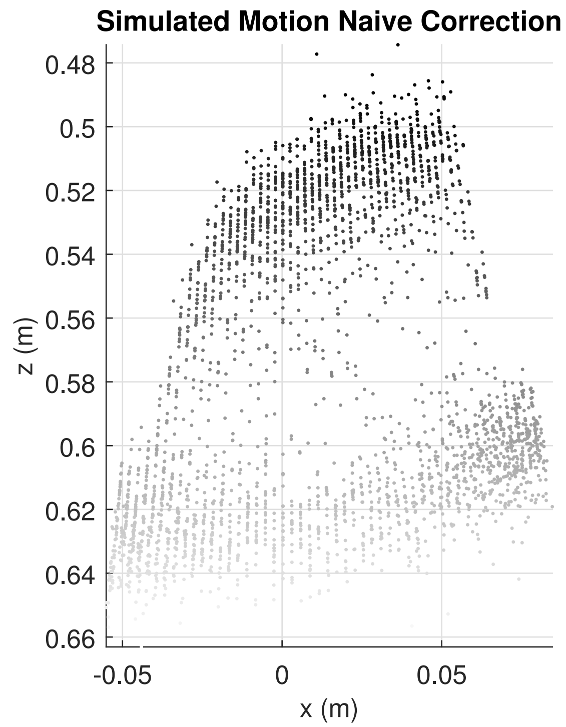


Figure 4.7: Simulated motion naive correction from Hussmann *et al.* (2011). A 3D side elevation of the 50 mm radius sphere under motion shows a better approximation than without motion correction, however significant error is still apparent, especially from the fixed pattern phase offsets.

Table 4.1: RMSE and SD comparison between no motion correction and the technique presented by Hussmann *et al.* (2011).

	Factory Calibrated Data (No Motion Correction)	Naive Motion Correction (Hussmann *et al.*, 2011)
RMSE (m)	0.0759	0.0587
SD (m)	0.0755	0.0151

resulting in a lower subtracted value. The platform of the translation stage is visible, as well as part of the edges of tracks of the translation stage.

Each phase step is then divided by the radial amplitude calibration (Figure 3.19), after which the phase steps are aligned through interpolation. The image is then demodulated (applying the phase offset corrections), after which the complex phasor is separated into the amplitude and phase components, subtracting the harmonic distortion to produce a final motion corrected output image (Figure 4.9a). The still reference image is repeated for direct comparison (Figure 4.9b). When compared to the still reference image, the proposed motion correction algorithm has an RMSE of 4.2 mm, and a SD of 4.2 mm. This is an RMSE improvement of approximately 72 mm compared to applying no motion correction, and an improvement of approximately 55 mm compared to the naive motion correction from Hussmann *et al.* (2011).

For comparison, the RMSEs and SDs of the proposed motion algorithm are found

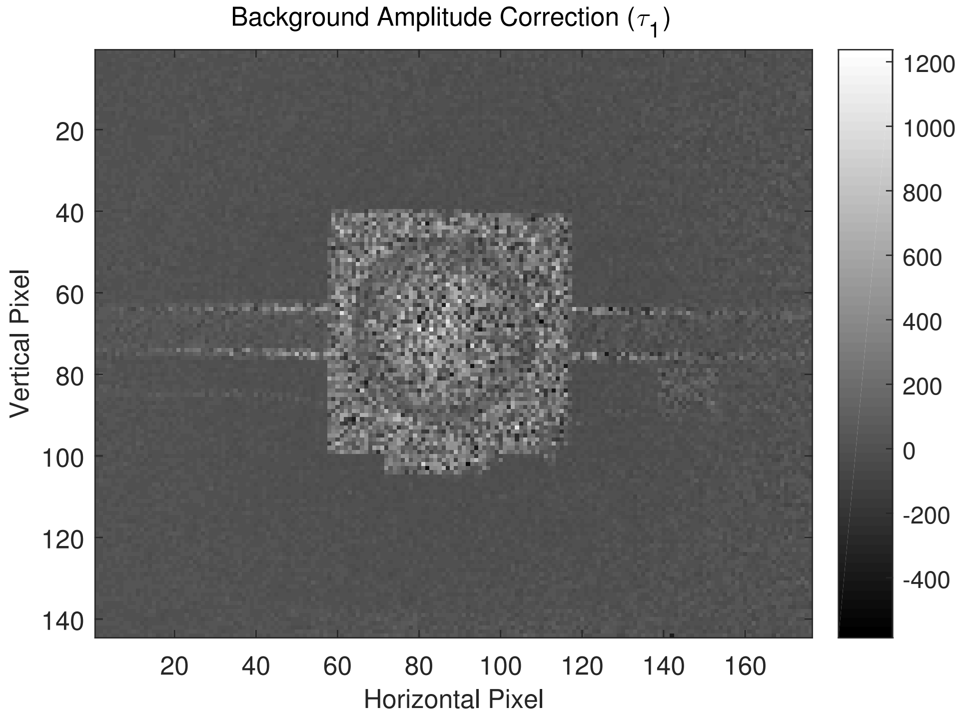
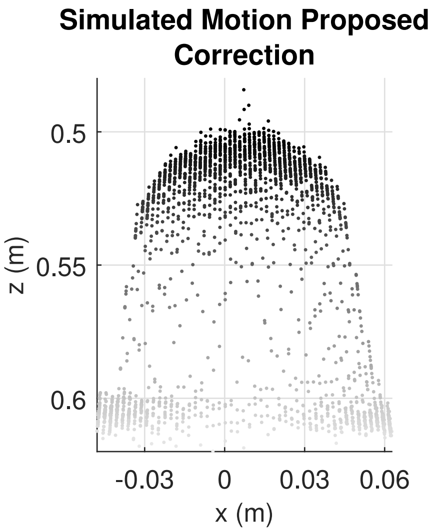
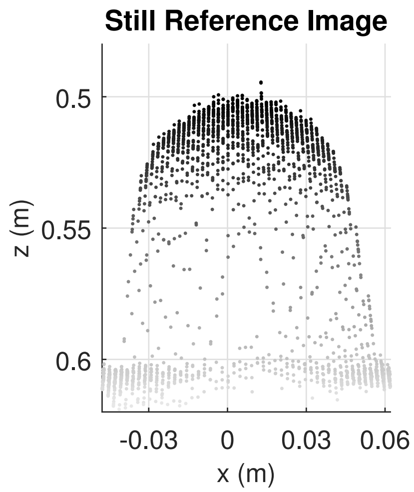


Figure 4.8: Amplitude dependent background offset for the first phase step (τ_1).



(a) 3D side elevation of the proposed motion correction algorithm.



(b) 3D side elevation of the staionary reference image.

Figure 4.9: Simulated motion with the proposed improvements to the motion correction algorithm from Hussmann *et al.* (2011). The still reference image is included for side-by-side comparison.

Table 4.2: RMSE and SD comparison between applying the different calibrations to the motion correction technique. The background, amplitude, phase, and interpolation calibrations are applied are applied to the motion affected data individually, then finally combined for an overall improvement.

	Background	Amplitude	Phase	Interpolation	Combined
RMSE (m)	0.0594	0.0818	0.0305	0.0635	0.0042
SD (m)	0.0124	0.0584	0.0301	0.0272	0.0042

when applying only the background, amplitude, and phase offset calibrations individually (Table 4.2). The greatest improvement to RMSE is seen in the phase calibration, while the greatest improvement to SD is seen in the background correction. When combining the three calibrations, over an order of magnitude improvement over using the camera in its off-the-shelf state is seen, with an RMSE and SD of 4.2 mm. The amplitude has the lowest effect in both RMSE and SD, because the amplitude correction is strongest at the edges of the image, whereas in this case the region of interest is close to the centre of the image.

4.3.2 Significance testing

The various improvements and RMSEs are then tested for significance. Because 100 repetitions of the simulated motion were performed, an RMSE can be generated for each repetition. The mean of these RMSE values is found, as well as the SD of the RMSE values. A two-sample t-test is performed between each stage of the motion correction to test for significance, where the null hypothesis is that the two data sets share a common distribution. That is to say, if the null hypothesis is true, then the two data sets are likely the same (the proposed motion correction results are likely to be from random chance rather than an actual improvement).

The RMSE is computed for each of the 100 repetitions, for each of the following cases. The 'Standard' results arise when applying no motion correction (using the camera in its off-the-shelf state), and act as a baseline. The 'Naive' results indicate the improvements when using the technique proposed by Hussmann *et al.* (2011). The motion correction technique discussed above is then applied sequentially. Initially, only the phase offset calibrations are added in the 'Phase' results. The motion correction is then performed again, with the addition of the background calibrations (both phase and background are applied) giving the 'Background' results. Finally, the amplitude corrections are added, giving the overall motion correction from all calibrations in the 'Amplitude' results.

The mean RMSEs show gradual improvement when adding in more calibrations, with the lowest improvement being from the amplitude calibration implementation. The SD of the RMSEs was negligibly small in all cases, with the largest SD being 2.8×10^{-4}. Plotting each of these steps (Figure 4.10), with each point showing the RMSE, and error bars showing ± 3 SD, visually shows the improvements when adding

Table 4.3: Mean RMSE and SD of the RMSE applying different motion correction techniques.

	Standard	Naive	Phase	Background	Amplitude
RMSE (m)	0.0759	0.0587	0.0305	0.0044	0.0042
SD (m)	7.4×10^{-5}	2.8×10^{-4}	6.9×10^{-5}	6.1×10^{-5}	6.1×10^{-5}

additional calibrations to the motion correction algorithm. To test for significance, paired t-test were run between each of the displayed steps. The null hypothesis was rejected in all of the t-test results ($h = 1$), with p values of zero to machine precision.

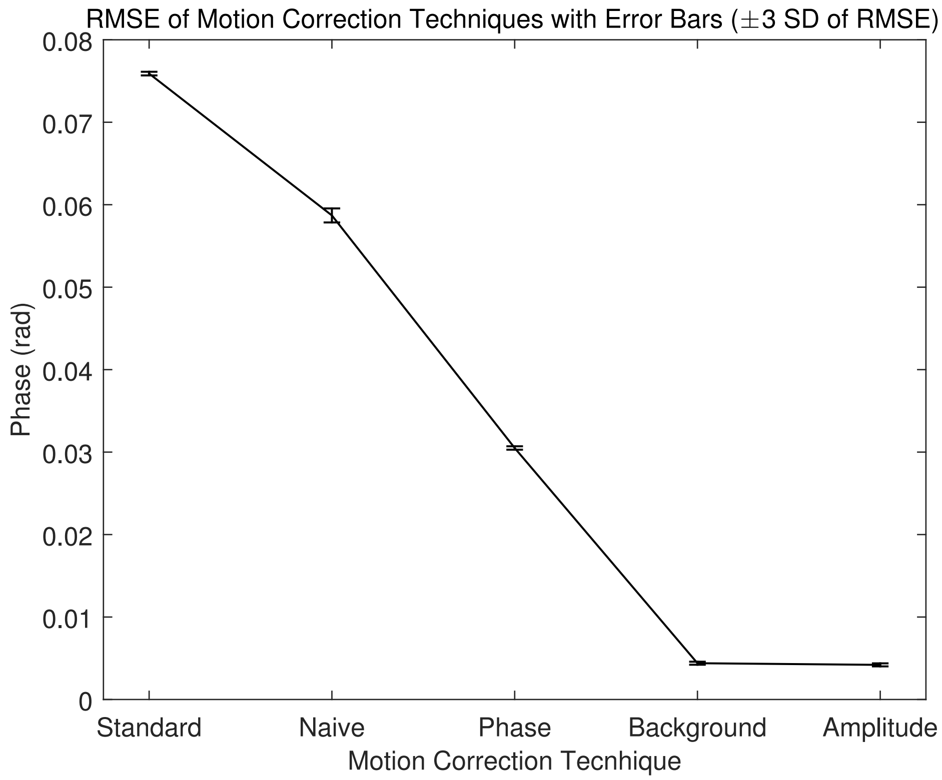


Figure 4.10: RMSEs with various motion corrections applied. Error bars are plotted, showing ± 1 SD.

4.3.3 Averaging and integration time

The next experiment looks to examine the effects of reducing both the integration time and the number of averages per phase step in the simulated motion (Table 4.4). It is seen that reducing the number of frames averaged from 100 to 1, gives an RMSE and SD reduction of approximately 0.9 mm. When reducing the integration time by 0.5 ms, the RMSE and SD are almost doubled to 8.1 mm. Reducing the integration time is much more detrimental than reducing the number of averaged frames. This is promising in terms of real-time motion, as reducing integration time can be compensated by increasing the power of the light source in the case of a

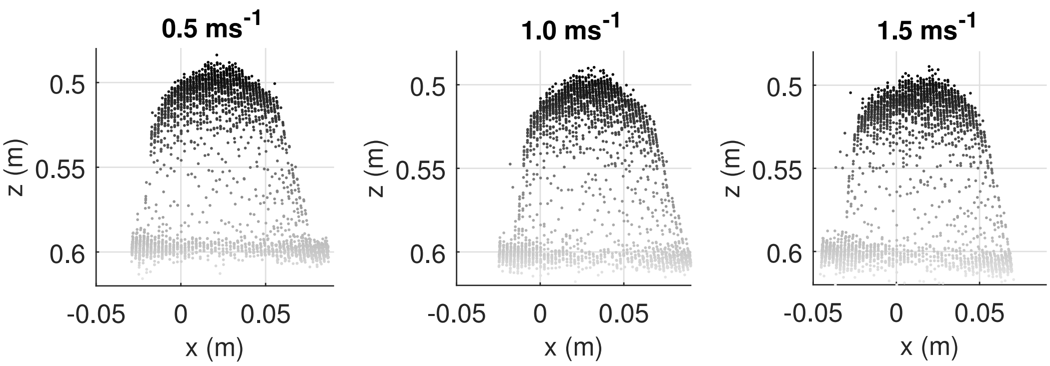
Table 4.4: Comparison of RMSE and SD when comparing the number of frames and integration time used with the motion correction algorithm.

	100 frames, 1.8 ms	1 frame, 1.8 ms	100 frames, 1.3 ms
RMSE (m)	0.0042	0.0051	0.0081
SD (m)	0.0042	0.0050	0.0081

custom time-of-flight camera.

4.3.4 Real-time motion

The motion experiments are then repeated at full speed, where the object is physically moving whilst being captured by the time-of-flight camera. The translation stage is programmed to begin moving at a specified speed, and once a certain point along the translation stage is reached (in this case close to the optical centre), the camera is set to trigger and capture the moving sphere. Because the amplitude is unable to be calculated for each position, the amplitude of the still reference image is used for the amplitude dependent background offset. The experiment is repeated for three speeds, of $0.5\,\mathrm{m\,s^{-1}}$, $1.0\,\mathrm{m\,s^{-1}}$, and $1.5\,\mathrm{m\,s^{-1}}$ (Figure 4.11). Visually these three phase images are very similar, where it is difficult to distinguish between the different speeds. Comparing the RMSEs and SDs for these three experiments (Table 4.5) shows both RMSEs and SDs as 6.1 mm, and 6.5 mm, and 8.0 mm for the speeds of $0.5\,\mathrm{m\,s^{-1}}$, $1.0\,\mathrm{m\,s^{-1}}$, and $1.5\,\mathrm{m\,s^{-1}}$ respectively. It can be seen that increasing the speed has an effect on the RMSE and SD. This is partially caused by the introduced intra-frame motion blur, where the object moves slightly during the integration time of each phase step. There is also likely some error introduced from the amplitude dependent background offset, where the crude technique of using the still image's amplitude for background correction has been used.



(a) $0.5\,\mathrm{m\,s^{-1}}$ motion correction. (b) $1.0\,\mathrm{m\,s^{-1}}$ motion correction. (c) $1.5\,\mathrm{m\,s^{-1}}$ motion correction.

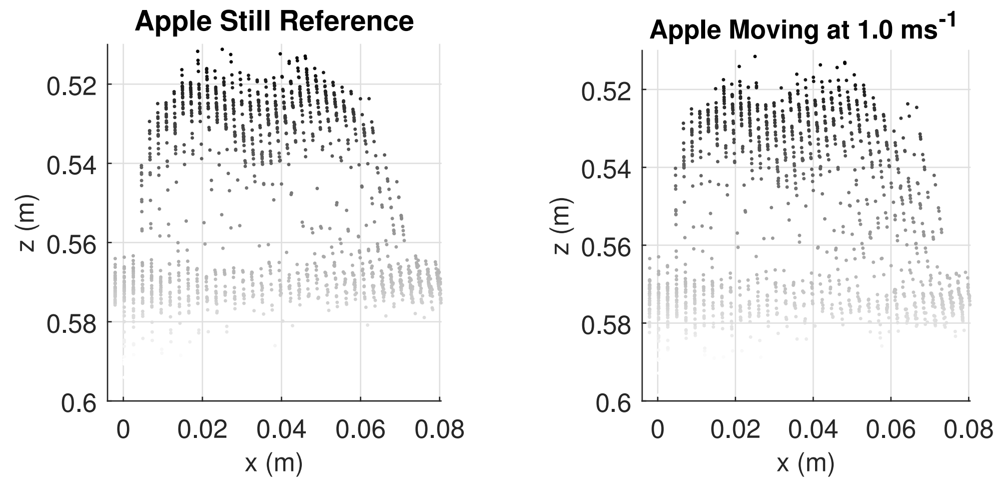
Figure 4.11: 3D side elevation of real-time motion at $0.5\,\mathrm{m\,s^{-1}}$, $1.0\,\mathrm{m\,s^{-1}}$, and $1.5\,\mathrm{m\,s^{-1}}$, using the proposed motion correction technique.

The final practical motion correction experiment looks to investigate an apple

Table 4.5: Real-time motion comparison of the RMSE and SD, using the proposed correction technique. The object is set to move at speeds of $0.5\,\mathrm{m\,s^{-1}}$, $1.0\,\mathrm{m\,s^{-1}}$, and $1.5\,\mathrm{m\,s^{-1}}$.

	$0.5\,\mathrm{m\,s^{-1}}$	$1.0\,\mathrm{m\,s^{-1}}$	$1.5\,\mathrm{m\,s^{-1}}$
RMSE (m)	0.0061	0.0065	0.0080
SD (m)	0.0061	0.0065	0.0080

moving on the translation stage, running at industrial specifications for fruit grading. The speed is set to move at $1.0\,\mathrm{m\,s^{-1}}$, at a distance of 0.5 m. The integration time is kept at 1.8 ms, as with the previous experiments. A still image of an apple is first captured for reference (Figure 4.12a), where the apple is standing upright. The top of the image shows a dip where the top of the core begins. The image appears noisier than the sphere, likely due to the non-symmetric shape of the apple as well as the specular nature of the apple's skin. The proposed motion correction algorithm is applied to the apple moving in real-time at $1.0\,\mathrm{m\,s^{-1}}$ (Figure 4.12b). The apple's shape follows that of the still reference image relatively closely, however there is some additional noise. The RMSE and SD between the still reference image and the moving apple corrected with the proposed motion correction algorithm is 6.8 mm and 6.4 mm respectively. Comparing this result to when applying no motion correction, as well as when applying the naive motion correction from Hussmann *et al.* (2011) shows similar improvements as seen in the spherical object case (Table 4.6).



(a) 3D side elevation of a still apple for reference.

(b) 3D side elevation of a motion corrected apple moving at $1.0\,\mathrm{m\,s^{-1}}$.

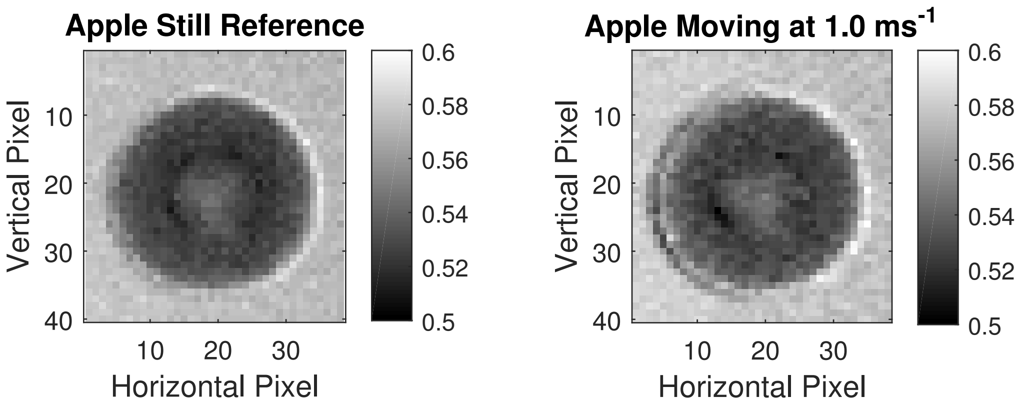
Figure 4.12: Proposed motion correction of an apple moving at $1.0\,\mathrm{m\,s^{-1}}$. A still image of the apple is also shown for reference.

Because of the non-symmetric nature of the apple's shape, 2D images of the apple better shows the profile of the apple (Figure 4.13). The still reference image shows the top of the core in the centre of the apple increases in depth. In the motion corrected case, the core follows the correct pattern increasing in depth, however

Table 4.6: Comparisons between motion correction techniques for an apple moving at $1.0\,\mathrm{m\,s^{-1}}$.

	No Motion Correction	Naive Motion Correction	Proposed Motion Correction
RMSE (m)	0.0844	0.0607	0.0068
SD (m)	0.0843	0.0378	0.0064

the biggest sources of error appear on the edges in the direction of motion (left and right). These edges appear worse than that of case with the moving spherical object. This may be due to the crude background dependent amplitude correction, the intra-frame motion blur, or another potential issue with the apple measurements is in having a smaller target than the sphere, whilst maintaining the same distance to the object, resulting in a lower object resolution. The apple also has steeper edges, where a sphere has a gradual change in depth toward the edge.



(a) 2D image of the top profile of a stationary reference apple.

(b) 2D image of the top profile of a motion corrected apple moving at $1.0\,\mathrm{m\,s^{-1}}$.

Figure 4.13: 2D image showing the top profile of a motion corrected apple moving at $1.0\,\mathrm{m\,s^{-1}}$. A still image of the apple is also shown for reference.

4.4 Simulation

In this section, a software simulation of a time-of-flight camera is produced, generating a motion affected sphere based on the practical experiments performed above. The purpose of this simulation is firstly for verification of the motion correction algorithm presented above. The simulation is secondly used for analysing remaining error sources, as well as investigating the interpolation discussed above, analysing the effect of non-integer pixel shifts between phase steps.

4.4.1 Image generation

Initially, a simulated time-of-flight camera is generated based on the SR4000. The same resolution (144×176 pixels) is used, generating values for the phase and amplitude images based on those acquired in the above experiments.

In order to generate the raw phase steps for the simulated camera, phase and amplitude images are first produced and the demodulation is reversed. The phase steps are then passed to the same modified motion correction demodulation algorithm as in Section 4.2. Analysing the expanded phase step equation after removing harmonic distortion (ρ_ϕ, can be removed independently of demodulation) and fixed pattern gain offset (η, negligible for the SR4000) gives,

$$\tau_n = \sigma A \cos(\phi + \theta_n + \delta) + (\psi_\alpha + \beta). \tag{4.12}$$

Ignoring the background components for the initial phase step generation, the remaining phase step equation can be separated into components to isolate the phase offset (δ) using angle sum trigonometric identities,

$$\tau_n = \sigma A\big(\cos(\phi + \theta_n)\cos(\delta) - \sin(\phi + \theta_n)\sin(\delta)\big). \tag{4.13}$$

The components containing only the amplitude and phase are then isolated,

$$I_{c_n} = A\cos(\phi + \theta_n), \tag{4.14}$$

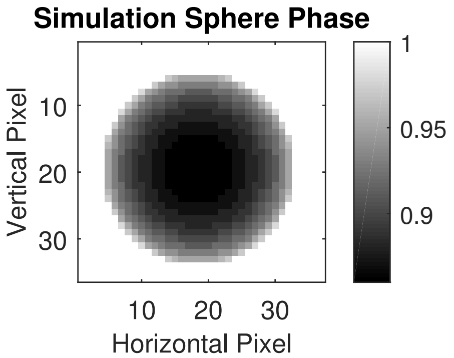
$$I_{s_n} = A\sin(\phi + \theta_n). \tag{4.15}$$

Phase and amplitude images are simulated with values similar to the images of the spherical object in Section 4.2. The spherical object is created with a radius of 15 pixels, and the initial position is defined as the central pixel. Two sets of four amplitude and phase images are generated (Figure 4.14), where the first set is a still reference image and the second set is affected by motion. The phase and amplitude images that are affected by motion have pixel offsets according to the vector $\mathbf{p}_{shift} = [s_1 \ s_2 \ s_3 \ s_4]$, where s_n defines the number of pixels each respective phase and amplitude image combination associated to the phase step τ_n is shifted. In this case, each subsequent phase step is shifted by an additional 5 pixels, giving a pixel shift vector of $\mathbf{p}_{shift} = [0 \ 5 \ 10 \ 15]$.

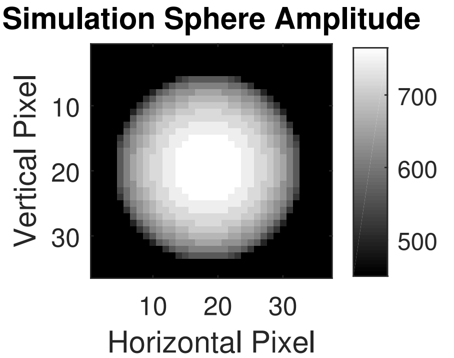
After combining the phase and amplitude as per Equations 4.14 and 4.15, the phase offset term is included along with the amplitude and background error sources. The four raw phase steps are generated based on Equation 4.13,

$$\begin{aligned}\tau_n &= \sigma(I_{c_n}\cos(\delta) - I_{s_n}\sin(\delta)) + (\psi_\alpha + \beta)\\ &= \sigma A\cos(\phi + \theta_n)\cos(\delta) - \sigma A\sin(\phi + \theta_n)\sin(\delta) + (\psi_\alpha + \beta)\\ &= \sigma A\cos(\phi + \theta_n + \delta) + (\psi_\alpha + \beta).\end{aligned} \tag{4.16}$$

The resulting four phase steps are influenced by the phase, amplitude, and background offsets discussed in Chapter 3. Demodulating the raw phase steps both with and without the error sources added (Figure 4.15) shows the effects of the motion



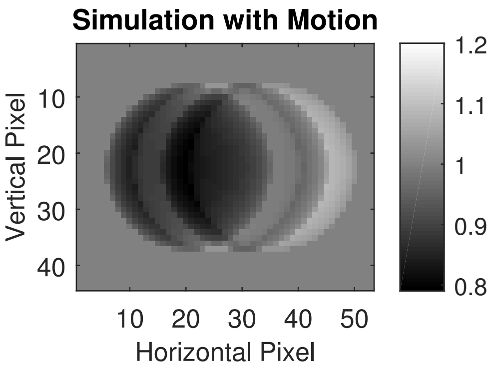
(a) Phase image generated containing the simulation sphere. The colourbar shows the depth in radians.



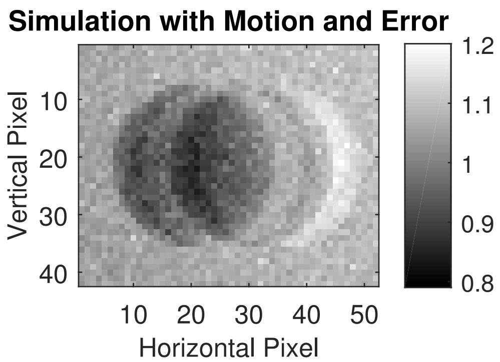
(b) Amplitude image generated containing the simulation sphere. The colourbar shows the relative amplitude.

Figure 4.14: Phase and amplitude image combination for the simulation sphere object at a fixed position.

in the phase image. It can be seen that the sphere is highly deformed by motion blur, even more so with the addition of the phase, amplitude and background offsets. A 3D side elevation view of the sphere without applying motion correction (Figure 4.16a) shows that the sphere object is highly distorted, however the phase, amplitude, and background offsets are corrected (analogous to the factory calibrated output). The RMSE and SD when compared to a still reference sphere, applying no motion correction are 16.5 mm and 16.0 mm respectively.



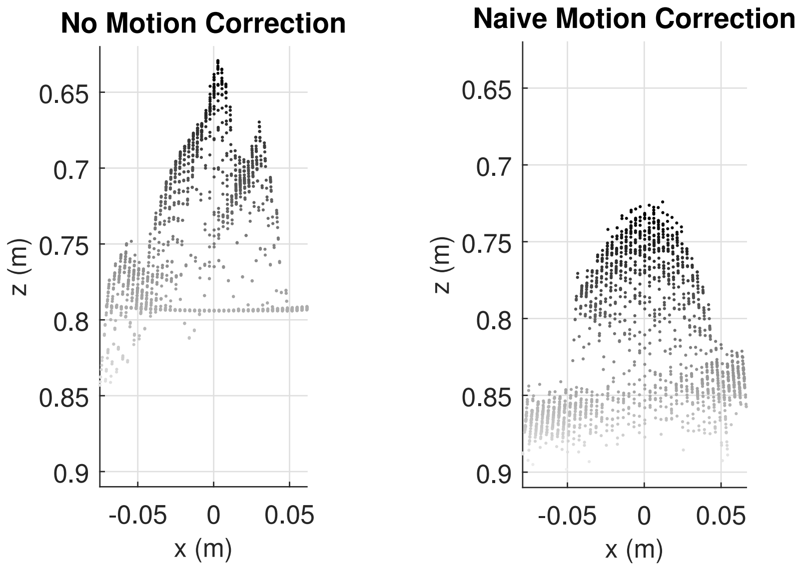
(a) Phase image of the demodulated raw phase steps affected by motion. The colourbar shows the depth in radians.



(b) Phase image of the demodulated raw phase steps affected by motion, with phase, amplitude, and background offsets added. The colourbar shows the depth in radians.

Figure 4.15: Phase images of the simulation of the motion affected sphere. The sphere is shown both with and without the phase, amplitude, and background errors added.

The naive motion correction from Hussmann *et al.* (2011) is then applied to the motion affected sphere (Figure 4.16b). For the naive motion correction, the RMSE is found as 62.5 mm, with an SD of 17.3 mm. In this case, the naive correction performs worse than when no motion correction is applied. This means that in this case, the effect of the phase, amplitude, and background offsets is more severe than the effects of motion. One possible reason this is different to result of the practical case from Section 4.3 is due to the edges of the translation stage. The region of interest in the practical case is based on the output of the motion corrected image, hence overlap on the edge of the translation stage in the case without motion correction generates additional motion artefacts.



(a) 3D side elevation of the motion affect sphere, without motion correction.

(b) 3D side elevation of the motion affect sphere, applying the motion correction algorithm presented by Hussmann *et al.* (2011).

Figure 4.16: 3D side elevation phase images of the demodulated motion affected sphere object. The sphere is shown both without motion correction, and with the motion correction technique from Hussmann *et al.* (2011) applied.

The improved motion correction algorithm described is then implemented for the simulation, and the motion is completely corrected, showing a negligible RMSE and SD (Figure 4.17).

4.4.2 Scaling

One of the key goals of the simulation is the investigation of non-integer pixel shifts in the motion between phase steps. This is investigated by adding non-integer shifts to the sphere simulation data. In order to correctly perform a non-integer shift, the image needs to first be scaled up. The increase in scale allows the amount of shift to be more accurately manipulated before downscaling. In this case, a scaling factor of five was used, where the image, the object, and the baseline shifts were multiplied by five. Instead of a shift vector of $\mathbf{p}_{shift} = [0\ 5\ 10\ 15]$, the baseline shifts become $\mathbf{p}_{shift} = [0\ 25\ 50\ 75]$. This allows the object to be shifted by numbers non-divisible by five. In this case, the shift vector was changed to $\mathbf{p}_{shift} = [2\ 23\ 52\ 73]$, relating

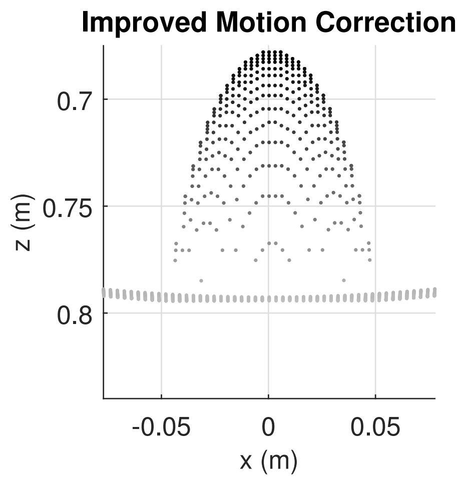


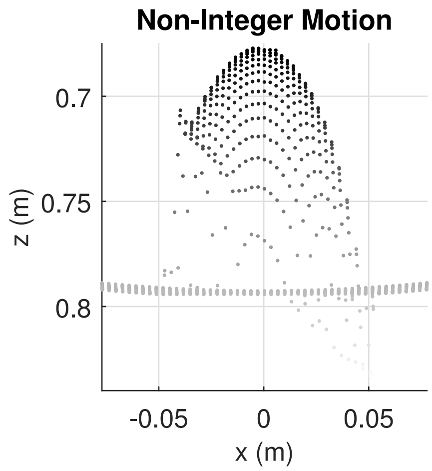
Figure 4.17: Motion corrected sphere using the improved technique.

to unscaled shifts of $\mathbf{p}_{shift}$ = [0.4 4.6 10.4 14.6], which is at the extreme of offsets from each pixel.

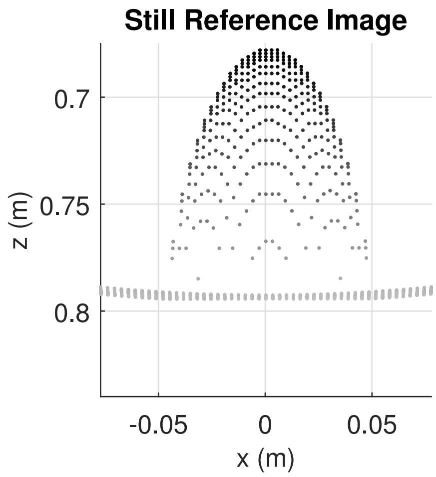
Because the phase step equation has been separated into a phase/amplitude component, and a phase offset component, the phase/amplitude component is able to be scaled up, without compromising the offset component. After each phase step is scaled up and shifted according to the vector $\mathbf{p}_{shift}$, the image is down-sampled to return it to the original resolution (144×176). This is done by selecting the central pixel in each upscaled group of 5×5 pixels.

The motion correction algorithm is first run without any errors from phase, amplitude, or background added. The motion correction algorithm is initially run without interpolation, where a shift of $\mathbf{p}_{shift}$ = [0 5 10 15] is used for correction. The 3D side elevation (Figure 4.18) shows the greatest error at the edges of the sphere in the direction of motion. A still reference image is included for direct comparison. The RMSE and SD are both shown to be 11 mm (Table 4.7). Interpolation is then added to the motion correction algorithm, allowing near exact pixel shifts to re-align the phase steps as closely as possible. The RMSE and SD are both found to be 1.8 mm, which is an order of magnitude improvement over using the non-interpolated pixel shifts. This demonstrates that non-integer shifting is an issue in motion correction on data from the SR4000. This is however a worse case scenario, with shift offsets close to half a pixel away from each phase step. The phase, amplitude, and background offsets are then added to the raw phase steps, to analyse any effects from interpolation. It is seen that the RMSE and SD are 19 mm, which is negligibly close to the case without offsets added. This suggests that the interpolation does not affect the error correction when running the proposed motion correction algorithm.

Future work on the simulation would look to firstly implement the intra-frame motion blur. This would aid in proving the above analysis for real-time motion,



(a) 3D side elevation of a simulation with a motion corrected sphere, without using interpolation.



(b) 3D side elevation of a still simulated sphere, for direct reference.

Figure 4.18: 3D side elevation of a motion affected sphere object, corrected with the proposed algorithm. A still sphere is added for direct reference.

Table 4.7: Simulation RMSE and SD for the motion affected sphere object using the proposed motion corrected algorithm. Results are shown both with and without using interpolation, as well as with phase, amplitude, and background errors added.

	No Interpolation	Interpolation	Interpolation with Errors
RMSE (m)	0.011	0.0018	0.0019
SD (m)	0.011	0.0018	0.0019

where the proposed motion correction algorithm performs worse with increasing object speed. Similarly, the effects of varying the intensity between phase steps could be analysed, which would help to investigate the effects of the amplitude dependent background offset correction. Varying intensity could also help to identify effects of non-Lambertian reflectors, where the amount of incident light on the sensor depends on the pose of the object relative to the camera. Finally, the effects of adding Gaussian noise to the simulation could be investigated. This would provide a more realistic simulation, and could be implemented by analysing the SR4000 phase steps to find the signal to noise ratio, and implementing it on the simulation phase steps.

Chapter 5

Conclusion and Outlook

The main goal of the work presented in this thesis was to develop an improvement on existing motion correction algorithms in time-of-flight cameras, for use in industrial applications. Initially, background information gives an in-depth review on the operation of time of flight cameras. The SR4000 from Mesa Imaging was described in detail, with emphasis on its operation in both its factory calibrated state as well as in 'raw mode'. The noise and error sources in time-of-flight cameras were then discussed, along with the calibration techniques from previous authors. The background section concluded with an overview of the motion blur problem in time-of-flight cameras. A number of correction techniques were discussed, with particular emphasis on the technique presented by Hussmann *et al.* (2011), which was further investigated in Chapter 4.

The calibration set of the the MESA SR4000 is then investigated. Each source of error was discussed and identified, then extracted from the SR4000 for later use. The phase offsets were first investigated, where the key findings showed the most significant source of error was the gradual phase offset, with a mean absolute error of 28.8 mm. When comparing the factory calibrated data and 'raw data', an RMSE of 62.4 mm was seen, with a SD of 37.9 mm. Comparing the factory calibrated data and data calibrated using the described techniques, an RMSE of 6.1 mm is seen, with a SD of 30 mm, giving over an order of magnitude improvement for both the RMSE and SD.

The phase offset calibrations were then compared using a Bland-Altman graph, where the quoted absolute accuracy of the SR4000 ($\pm$10 mm) was used as a guide as to whether or not the measurement techniques are in agreement. It was found that the raw data did not fall within the limits, hence was not in agreement with the factory data. Data calibrated with the described techniques mostly fell within the limits, hence was defined as in agreement for the applications presented in the thesis.

Amplitude calibrations were investigated, however where much less of an issue compared to the phase offsets. Because the amplitude calibration pattern is radial, the biggest effect is at the edges of the image. Because the experiments on motion correction were relatively close to the centre, the amplitude calibration was not as vital as the phase correction.

The final time-of-flight camera specific calibration investigated the background signal. These calibrations are not required when using time of flight cameras in ordinary operation, because of the inherent background correction in the demodulation technique (Equation 2.3). When applying the motion correction technique in Chapter 4, the inherent background correction is lost, hence the background calibrations were extracted from the SR4000 for use in this application. The dark current was found with a lens cap butted to the sensor, and an amplitude dependent background correction was derived based on the extracted calibration set from the SR4000's factory calibrations.

Photogrammetric calibrations are required when converting data between 2D and Cartesian 3D coordinates. These are found based on the focal length of the camera, the principle (central) point of the image sensor, and the pixel pitch. Photogrammetric calibration also includes lens distortion, which takes into account both radial and tangential distortion. Each point on the 2D sensor is then able to be projected into 3D space, giving a Cartesian x, y, and z coordinate.

The photogrammetric calibrations were found in three ways. Firstly, the technique discussed by Dorrington *et al.* (2011) was implemented, where the calibration is extracted from the camera by generating normal vectors for each pixel, which can be applied to future 2D data. Secondly, sample C++ code provided by Mesa Imaging was used along with OpenCV to extract the various parameters to find the camera calibration matrix, as well as the lens distortion coefficients. Finally, the technique presented by Zhang (2000) using a series of checkerboard patterns was used. The technique from Dorrington *et al.* (2011) was the easiest to implement, as well as being closely related to the extraction techniques described above. The C++ technique generated all of the parameters, which is useful when manually applying the calibration, giving more flexibility. When implementing the checkerboard technique, it was seen that the camera calibration matrix was estimated relatively well, however the lens distortion coefficients were not accurate. This could be further improved by investigating optimal checkerboard patterns/positions and increasing the number of captured images.

Motion blur in time-of-flight cameras was then investigated. In particular, the motion correction technique from Hussmann *et al.* (2011) was explored, which looks to correct inter-frame motion blur by re-aligning the phase steps after capture. The presented work only applies calibration to a few of the errors discussed in Chapter 3. The biggest remaining issue lies in the calibration of the phase offsets, which are unable to be corrected through the standard demodulation technique. A new demodulation technique was proposed, which incorporates the phase offsets, such that they can be correctly calibrated. The background correction is also important for the calibration, as it is no longer corrected during demodulated. Instead, an amplitude image of the object was used to calibrate the background of each phase step. Additionally, because of the low resolution of the SR4000, when the object moves a non-integer amount of pixels between phase steps, there are additional errors generated. Interpolation was applied to both the phase steps as well as the demodulation technique, compensating for the non-integer shifts.

A number of experiments are then performed for verification of the improvement of the proposed motion correction algorithm. The SR4000 was initially set to capture a diffuse sphere of radius 50 mm moving along a lateral translation stage at a distance of 0.5 m from the camera. Initial experiments are performed using 'simulated motion', where the sphere was imaged at fixed increments of approximately 4 pixels along the translation stage, then combined to give the impression of motion. The RMSE without any motion correction applied was seen to be 75.9 mm, while the RMSE using the motion correction from Hussmann *et al.* (2011) was 58.7 mm. Applying the proposed improvements to the motion correction gave an RMSE of 4.3 mm. Significance testing was then performed between sequential improvements on the motion correction technique, starting without motion correction, followed by the naive correction from Hussmann *et al.* (2011), then adding incremental improvements to the proposed algorithm. The null hypothesis was rejected in all cases, indicating that it is highly unlikely that the obtained data sets shared a common distribution, that is, it is highly unlikely that these results occurred by chance.

Tests were then performed to compare between reducing both the integration time of the camera (from 1.8 ms to 1.3 ms), as well as reducing the number of frames averaged (from 100 to 1). The results showed that reducing the integration time had a greater affect than reducing the number of averages, with RMSE increases of 39 mm and 9 mm respectively.

Full speed experiments were then run, where the sphere was set to move at speeds of $0.5\,\mathrm{m\,s^{-1}}$, $1.0\,\mathrm{m\,s^{-1}}$, and $1.5\,\mathrm{m\,s^{-1}}$. The camera was triggered to capture data as quickly as possible, once the object reached a certain position. The results showed RMSEs of 6.1 mm, 6.5 mm, and 8.0 mm respectively. This increase in RMSE with speed is likely due to increasing intra-frame motion blur.

The final experiment performed was imaging a moving apple at industrial specifications (Appendix A). The fully motion corrected apple showed an RMSE of 6.8 mm, compared to RMSEs of 84.4 mm and 60.7 mm for the cases with no motion correction, as well as the naive motion correction presented by Hussmann *et al.* (2011) respectively.

The final section of this thesis was in the development of a software simulated time-of-flight camera. A motion affected sphere was produced and imaged by the simulated camera, with the main interest being in the investigation of the effects of non-integer pixel shifts. The phase steps were scaled up, such that they could be precisely moved before being scaled back down. Shifting each phase step to extremes of close to half a pixel each showed RMSEs of 11 mm, and 1.8 mm, without and with interpolation respectively. This suggests that there is in fact an issue with non-integer shifts for the SR4000. The experiment was then repeated, with the inclusion of the offsets and errors discussed in the calibration section on the simulated camera. The results of the experiment with both motion correction algorithm and interpolation showed that the RMSE was negligibly close to the case without the errors applied (RMSE = 1.9 mm). This suggests that the addition of interpolation is not detrimental to the motion correction algorithm.

5.1 Limitations and future work

In theory, the calibration extraction process described in Chapter 3 can be applied to any time-of-flight camera that is factory calibrated, and allows access to the raw data. The main advantages of the calibration technique presented in this thesis is in simplicity, using readily available components that are relatively inexpensive. The other advantage is that high accuracy placement of the retro-reflector is not necessarily required, because the calibration is derived from the factory calibrated camera output. The translation stage is not strictly required, however it is a convenient way of moving the retro-reflector. The techniques described above could however be applied to a completely uncalibrated time-of-flight camera, where in this case the precise distance would be required (some uncertainty may arise from placement of the camera and retro-reflector). In this case, the 'raw mode' data would be compared to a perfectly flat plane generated at the known distance of the retro-reflector to generate the described calibrations, rather than to the factory calibrated data. Alternative to the retro-reflector setup, concentric spheres could be used for the calibration, as each pixel would measure the same distance in an ideal experimental setup. Multiple spheres are required in this case when attempting to model the harmonic distortion, hence calibration using this technique is expensive.

The main limitations with the calibration set of the SR4000 lie in the background and amplitude calibration. Because the background calibrations are corrected during demodulation, they are unable to be extracted from the factory calibration set. The result is that the dark current and amplitude dependent background correction had to be created separately to the calibration extraction. The dark current was generated using a retrofitted lens cap, which is not an ideal solution, as there can be light leakage. Future work could look to take the SR4000 to a large open space, where a dark image could be captured with minimal light returning to the camera, however this is a slight logistical issue as power is required for the camera. Alternatively (and more expensively), light absorbing material could be used to line a wall or room that is being imaged, to negate the light reflected to the camera. Time-of-flight cameras which give control of the light source can simply switch the source off, and minimise the light from the environment to capture the dark image. The background dependent amplitude correction also needed to be calibrated, however was done so using the data captured with the calibration set. The radial amplitude correction was also found through extraction of the calibration set, however it could only be found after correction of other amplitude calibrations in the SR4000.

There are also some limitations with the experimental setup. Even though precise knowledge of the distance to the retro-reflector is unnecessary when comparing 'raw data' to factory calibrated data, some care still needs to be taken in the alignment of the camera. Under the assumption that the retro-reflector acts as a point source, the camera's sensor should be as close to perpendicular as possible to the direction of the point source. The point source should also be as close as possible to the centre of the sensor. If the camera is incorrectly aligned. The planar wave will hit the sensor at different times, resulting in an uneven phase distribution across the sensor.

Other future work would look to further investigate the phase variation during the warm-up time of the SR4000. The exact cause of the phase increase during the warm-up is unknown. The SR4000 uses optical feedback (Lehmann *et al.*, 2009) in its factory calibrated state to correct for the effects of temperature variation, however this feature is disabled when operating the camera in 'raw mode'. Future work could look to model this temperature variation, perhaps with an external temperature sensor. Other considerations need to be investigated, such as ambient temperature, where the environment could change the total time for the camera to warm up.

The final consideration for future work in calibration is the integration time of the camera. For the work presented in this thesis, the integration time was kept constant between performing the calibrations and their application. As discussed in Chapter 2, the integration time affects the measured phase, and is commonly either kept constant, or calibrated for a range of integration times. Future work could look to adopt this method of multiple integration times, or research the cause of the error, providing an additional calibration dependent on the integration time.

The main limitation for the inter-frame motion blur during real-time motion arises from the amplitude dependent background correction. As was discussed in Chapter 4, the amplitude is required to correctly calibrate the background before demodulation. Knowledge of the amplitude is not possible in the case of real-time motion, as the object moves between phase steps. In the practical experiments above, the location of the object was found for each phase step, where a background correction was generated using the pre-determined amplitude. This technique fails when the exact shape of the object is not known before demodulation (as is the case with differently shaped apples on a conveyor belt system). Alternative methods could look to estimate the amplitude based on only single phase steps, after analysing the relationship between the raw phase steps and amplitudes for different object poses. Other possibilities include further investigation to the demodulation technique, where further work could seek to manipulate the demodulation algorithm to include the background, compensating for it after demodulation.

Future work would look to consider the intra-frame motion blur, where the object is moving during the integration time of the camera. In this case, a relatively low integration time of 1.8 ms was used, compared to the read out time affecting the inter-frame motion blur (4.6 ms). Intra-frame blur becomes more of an issue as the speed of motion increases, as was demonstrated in the real-time experiment from Chapter 4, where the RMSE and SD increased with speed. The simplest solution is to continue to reduce the integration time, reducing the intra-frame motion blur. The trade-off in this situation is an increase in noise, as was seen in the experiment where the integration time was reduced by 0.5 ms, and the RMSE almost doubled from 4.2 mm to 8.1 mm. The simplest way to compensate for this would be to increase the power output from the light source, however with the SR4000 this is not possible. For the case of a custom camera designed for industry, the power output and integration time could be optimised, however with considerations of eye safety arising from increased power.

Alternatively, solutions for correcting the intra-frame motion blur would be further investigated in future work. The coded exposure technique presented by Streeter and Dorrington (2014) provides a good first attempt into the correction of intra-frame motion blur. Optical flow is used in the technique, where eight phase steps are captured and coded with a six step binary sequence. This technique would be unable to be used in the case of the SR4000, as the phase step capture method is unable to be modified. As a result, further work into intra-frame blur correction using this technique requires an alternative camera, or a custom developed camera.

Other future work for motion correction would look into generalisation of the detection of travel distance. Because the examples presented in Chapter 4 implemented interpolation, an estimate of motion was initially found using binary images, followed by a brute force technique, using ± 1 pixel around this estimate to find the required amount of interpolation. The result is a slow processing time, which is not ideal in industrial solutions. The optimal solution would be to synchronise the frame rate with the speed of motion, where near integer pixel shifts could be generated, removing the need for interpolation. This is not possible with the SR4000, because of the fact that the frame rate is only modified when changing the integration time. This has adverse effects, either saturating the sensor or increasing noise from modifying the integration time.

Further generalisation of the motion correction would look to provide a more robust technique for determining the direction of motion. This is not an issue for industrial applications, as the velocity is usually constant and known (such as on a conveyor belt), however is an important step for generalising the motion correction to further applications (such as consumer use). Techniques combining optical flow could help in this case, however investigations into the effect of acceleration would need to be investigated, where the number of pixels shifted between phase steps is no longer constant

Finally, with further investigation it would be advantageous to improve the simulation of the time-of-flight camera presented in Section 4.4. Firstly, Gaussian noise could be added to the system, based on the signal to noise ratio (SNR) of the SR4000, for more realistic simulation. Other improvements would simulate the varying amplitude of the object between phase steps. This could be used to assist in the development of an estimation for the background dependent amplitude calibration discussed above. The effects of the pose changing between phase steps could also investigated with manipulation of the original phase step images. The intra-frame motion blur could be implemented by adding additional shifts of the object, and averaging between them. This would then be combined with the inter-frame motion blur to produce a fully motion affected object, which could be used to determine the magnitude of each type of motion blur.

5.2 Outlook

With the addition of the work presented in this thesis, the outlook of inter-frame motion correction in general for time-of-flight cameras in industrial situations is very promising. With respect to the fruit grading industry, this work showed that it is possible to reproduce an apple moving along a translation stage with industrial specifications, with an off-the-shelf time-of-flight camera. An RMSE of 6.5 mm was seen using the proposed motion correction algorithm, which is a large improvement over the existing motion correction technique from Hussmann *et al.* (2011), which showed an RMSE of 60.7 mm for the same experiment.

There are many improvements that could be made to the proposed motion correction technique, however the majority of them arise due to limitations of the SR4000. In order to bridge the gap between the current state using the SR4000 and an industrially ready product, a number of improvements are required, starting with an improved camera. Other off-the-shelf cameras could be investigated, however the development of a custom industrial camera would likely produce the best results. This camera would be completely customisable, where the issues such as synchronisation for removing interpolation could be resolved. Custom light sources could be developed, which would be optimised for a particular industry (for example apples may proved better results at a particular wavelength), as well as being optimised to produce the greatest signal returns depending on the integration time required for the application. The phase step sequence could be modified to implement more complicated algorithms such as coded exposure if further intra-frame motion correction was required. Finally, a custom on board field programmable gate array (FPGA) could be used to greatly improve the processing time.

The future of time-of-flight imaging in motion affected industrial scenes is promising. As demonstrated in this thesis, under controlled conditions great improvements can be made toward object reconstruction. The work presented herein provides many potential extensions, both in terms of additional research as well as in the development of industrial-grade cameras for a variety of applications. Although the work was aimed at grading produce, the motion correction techniques could be applied to many industries where there is controlled motion.

References

Abdo, N. and A. Borgeat. 3D camera calibration. *Freiburg, Germany: Albert-Ludwigs-University Freiburg* (2010).

BiPM, I., I. IFCC, I. ISO, and O. IUPAP. International vocabulary of metrology–basic and general concepts and associated terms, 2008. *JCGM*, **200**, pp. 99–12 (2008).

Bland, J. M. and D. G. Altman. Statistical methods for assessing agreement between two methods of clinical measurement. *The Lancet*, **327(8476)**, pp. 307–310 (1986).

Bradski, G. The OpenCV Library. *Dr. Dobb's Journal of Software Tools* (2000).

Brown, D. C. Close-range camera calibration. *Photogrammetric Engineering*, **37**, pp. 855–866 (1971).

Büttgen, B., T. Oggier, M. Lehmann, R. Kaufmann, and F. Lustenberger. CCD/CMOS lock-in pixel for range imaging: Challenges, limitations and state-of-the-art. *Proceedings of the 1st Range Imaging Research Day*, pp. 21–32 (2005).

Charleston, S. A., A. A. Dorrington, L. Streeter, and M. J. Cree. Extracting the mesa sr4000 calibrations. In: *SPIE Optical Metrology, Videometrics, Range Imaging, and Applications*, volume 9528 of *Proc. SPIE*, p. 95280S. Munich, Germany (2015).

Chiabrando, F., D. Piatti, and F. Rinaudo. SR-4000 ToF camera: further experimental tests and first applications to metric surveys. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, **38(5)**, pp. 149–154 (2010).

Cree, M. J., L. V. Streeter, R. M. Conroy, and A. A. Dorrington. Analysis of the softkinetic depthsense for range imaging. In: *Image Analysis and Recognition*, pp. 668–675. Springer (2013).

Dorrington, A. A., J. P. Godbaz, M. J. Cree, A. D. Payne, and L. V. Streeter. Separating true range measurements from multi-path and scattering interference in commercial range cameras. In: *Three-Dimensional Imaging, Interaction, and Measurement*, volume 7864 of *Proc. SPIE*, p. 786404 (2011).

Dorrington, A. A., C. B. D. Kelly, S. H. McClure, A. D. Payne, and M. J. Cree. Advantages of 3D time-of-flight range imaging cameras in machine vision applications. In: *16th New Zealand Electronics Conference (ENZCon'09)*, pp. 95–99. Dunedin, New Zealand (2009).

Faugeras, O. *Three-dimensional computer vision: a geometric viewpoint.* MIT, Cambridge, MA (1993).

Foix, S., G. Alenya, and C. Torras. Lock-in time-of-flight (ToF) cameras: a survey. *IEEE Sensors Journal*, **11(9)**, pp. 1917–1926 (2011).

Fuchs, S. Multipath interference compensation in time-of-flight camera images. In: *Pattern Recognition (ICPR), 2010 20th International Conference on*, pp. 3583–3586. IEEE (2010).

Fuchs, S. and G. Hirzinger. Extrinsic and depth calibration of ToF-cameras. In: *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008.*, pp. 1–6. IEEE (2008).

Fuchs, S. and S. May. Calibration and registration for precise surface reconstruction with time-of-flight cameras. *International Journal of Intelligent Systems Technologies and Applications*, **5(3)**, pp. 274–284 (2007).

Godbaz, J. P., M. J. Cree, and A. A. Dorrington. Multiple return separation for a full-field ranger via continuous waveform modelling. In: *Image Processing: Machine Vision Applications II*, volume 7251 of *Proc. SPIE*, p. 72510T. San Jose, CA, USA (2009).

Gudmundsson, S. A., H. Aanæs, and R. Larsen. Environmental effects on measurement uncertainties of time-of-flight cameras. In: *Signals, Circuits and Systems, 2007. ISSCS 2007. International Symposium on*, volume 1, pp. 1–4. IEEE (2007).

Ho, Y.-S. Challenging technical issues of 3D video processing. *Journal of Convergence*, **4(1)**, pp. 1–6 (2013).

Horn, B. K. and B. G. Schunck. Determining optical flow. In: *1981 Technical Symposium East*, pp. 319–331. International Society for Optics and Photonics (1981).

Howard, I. P. and B. J. Rogers. *Binocular vision and stereopsis.* Oxford University Press (1995).

Hsu, S., S. Acharya, A. Rafii, and R. New. Performance of a time-of-flight range camera for intelligent vehicle safety applications. In: *Advanced Microsystems for Automotive Applications 2006*, pp. 205–219. Springer (2006).

Hussmann, S. and T. Edeler. Pseudo-four-phase-shift algorithm for performance enhancement of 3D-TOF vision systems. *IEEE Transactions on Instrumentation and Measurement*, **59(5)**, pp. 1175–1181 (2010a).

Hussmann, S. and T. Edeler. Robot vision using 3D TOF systems. In: *Robot Vision*, edited by A. Ude. InTech (2010b). ISBN 978-953-307-077-3.

Hussmann, S., A. Hermanski, and T. Edeler. Real-time motion artifact suppression in tof camera systems. *Instrumentation and Measurement, IEEE Transactions on*, **60(5)**, pp. 1682–1690 (2011).

Jimenez, D., D. Pizarro, and M. Mazo. Single frame correction of motion artifacts in PMD-based time of flight cameras. *Image and Vision Computing*, **32(12)**, pp. 1127–1143 (2014).

Kahlmann, T., F. Remondino, and H. Ingensand. Calibration for increased accuracy of the range imaging camera swissranger^{TM}. *Image Engineering and Vision Metrology (IEVM)*, **36(3)**, pp. 136–141 (2006).

Karel, W., S. Ghuffar, and N. Pfeifer. Quantifying the distortion of distance observations caused by scattering in time-of-flight range cameras. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, **38(Part 5)** (2010).

Kim, Y. M., D. Chan, C. Theobalt, and S. Thrun. Design and calibration of a multi-view tof sensor fusion system. In: *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*, pp. 1–7. IEEE (2008).

Klette, R., K. Schlüns, and A. Koschan. *Computer vision: three-dimensional data from images*, volume 1. Springer Singapore (1998).

Lange, R. 3d time-of-flight distance measurement with custom solid-state image sensors in cmos/ccd-technology. *Diss., Department of Electrical Engineering and Computer Science, University of Siegen* (2000).

Lange, R. and P. Seitz. Solid-state time-of-flight range camera. *IEEE Journal of Quantum Electronics*, **37(3)**, pp. 390–397 (2001).

Lehmann, M., B. Buettgen, and T. Oggier. Reference pixel array with varying sensitivities for time of flight (TOF) sensor (2009). US Patent 7,586,077.

Lindner, M. and A. Kolb. Lateral and depth calibration of PMD-distance sensors. In: *Advances in Visual Computing*, pp. 524–533. Springer (2006).

Lindner, M. and A. Kolb. Calibration of the intensity-related distance error of the PMD ToF-camera. In: *Optics East 2007*, pp. 67640W–67640W. International Society for Optics and Photonics (2007).

Lindner, M. and A. Kolb. Compensation of motion artifacts for time-of-flight cameras. In: *Dynamic 3D Imaging*, pp. 16–27. Springer (2009).

Lottner, O., A. Sluiter, K. Hartmann, and W. Weihs. Movement artefacts in range images of time-of-flight cameras. In: *Signals, Circuits and Systems, 2007. ISSCS 2007. International Symposium on*, volume 1, pp. 1–4. IEEE (2007).

Ma, Y. *An invitation to 3-D vision: from images to geometric models*, volume 26. Springer Science & Business Media (2004).

Marr, D. and T. Poggio. Cooperative computation of stereo disparity. *Science*, **194(4262)**, pp. 283–287 (1976).

May, S., D. Dröschel, S. Fuchs, D. Holz, and A. Nuchter. Robust 3d-mapping with time-of-flight cameras. In: *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pp. 1673–1678. IEEE (2009).

May, S., B. Werner, H. Surmann, and K. Pervolz. 3D time-of-flight cameras for mobile robotics. In: *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 790–795. IEEE (2006).

Mesa Imaging. *SR4000 User Manual*. MESA Imaging, Zürich, Switzerland (2010). Rev 1.4.

Mesa Imaging. *SR4000 Data Sheet*. MESA Imaging, Zürich, Switzerland (2011). Rev 5.1.

Mure-Dubois, J. and H. Hügli. Real-time scattering compensation for time-of-flight camera. In: *Proceedings of the ICVS Workshop on Camera Calibration Methods for Computer Vision Systems* (2007).

Oggier, T., B. Buettgen, and T. Zamofing. Adaptive neighborhood filtering (ANF) system and method for 3D time of flight cameras (2012). US Patent 8,223,215.

Oprisescu, S., D. Falie, M. Ciuc, and V. Buzuloiu. Measurements with tof cameras and their necessary corrections. In: *Signals, Circuits and Systems, 2007. ISSCS 2007. International Symposium on*, volume 1. IEEE (2007).

Payne, A. D., A. A. Dorrington, M. J. Cree, and D. A. Carnegie. Improved linearity using harmonic error rejection in a full-field range imaging system. In: *3D*

Image Capture and Applications VII, volume 6805 of *Proc. SPIE*, p. 68050D. San Jose, CA, USA (2008).

Payne, A. D., A. A. Dorrington, M. J. Cree, and D. A. Carnegie. Improved measurement linearity and precision for amcw time-of-flight range imaging cameras. *Applied optics*, **49(23)**, pp. 4392–4403 (2010).

Piatti, D. and F. Rinaudo. SR-4000$^{\mathrm{TM}}$and CamCube3.0 time of flight (ToF) cameras: Tests and comparison. *Remote Sensing*, **4(4)**, pp. 1069–1089 (2012).

Plaue, M. Technical report: Analysis of the PMD imaging system. *Interdisciplinary Center for Scientific Computing, University of Heidelberg* (2006).

Radmer, J., P. M. Fusté, H. Schmidt, and J. Kruger. Incident light related distance error study and calibration of the pmd-range imaging camera. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08.*, pp. 1–6. IEEE (2008).

Rapp, H. *Experimental and theoretical investigation of correlating TOF-camera systems*. Master's thesis, University of Heidelberg, Germany (2007).

Raskar, R., A. Agrawal, and J. Tumblin. Coded exposure photography: motion deblurring using fluttered shutter. *ACM Transactions on Graphics (TOG)*, **25(3)**, pp. 795–804 (2006).

Scharstein, D. and R. Szeliski. High-accuracy stereo depth maps using structured light. In: *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pp. I–195. IEEE (2003).

Schiller, I., C. Beder, and R. Koch. Calibration of a pmd-camera using a planar calibration pattern together with a multi-camera setup. *The international archives of the photogrammetry, remote sensing and spatial information sciences*, **37**, pp. 297–302 (2008).

Schmidt, M. O. Spatiotemporal analysis of range imagery. *Diss., Combined Faculties for the Natural Sciences and for Mathematics e, Ruperto-Carola University* (2008).

Schuon, S., C. Theobalt, J. Davis, and S. Thrun. High-quality scanning using time-of-flight depth superresolution. In: *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*, pp. 1–7. IEEE (2008).

Steiger, O., J. Felder, and S. Weiss. Calibration of time-of-flight range imaging cameras. In: *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pp. 1968–1971. IEEE (2008).

Streeter, L. and A. A. Dorrington. Coded exposure correction of transverse motion in full-field range imaging. *Optical Engineering*, **53(10)**, p. 102109 (2014).

Stürmer, M., J. Penne, and J. Hornegger. Standardization of intensity-values acquired by time-of-flight-cameras. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08*, pp. 1–6. IEEE (2008).

Szeliski, R. *Computer vision: algorithms and applications*. Springer Science & Business Media (2010).

Thorlabs, Inc. Engineered diffusers^{TM}(2015). Retrieved 2015, May 1 from, `https://www.thorlabs.com/navigation.cfm?guide\_id=17`.

Tsai, R. Y. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *Robotics and Automation, IEEE Journal of*, **3(4)**, pp. 323–344 (1987).

Wehr, A. and U. Lohr. Airborne laser scanningan introduction and overview. *ISPRS Journal of Photogrammetry and Remote Sensing*, **54(2)**, pp. 68–82 (1999).

Wehrwein, S. *Computational Photography: Coded Exposure and Coded Aperture Imaging*. Master's thesis, Middlebury College, Vermont, U.S.A (2010).

Weng, J., P. Cohen, and M. Herniou. Camera calibration with distortion models and accuracy evaluation. *IEEE Transactions on pattern analysis and machine intelligence*, **14(10)**, pp. 965–980 (1992).

Wiedemann, M., M. Sauer, F. Driewer, and K. Schilling. Analysis and characterization of the pmd camera for application in mobile robotics. In: *Proceedings of the 17th IFAC World Congress*, pp. 6–11 (2008).

Xu, Z., R. Schwarte, H. Heinol, B. Buxbaum, and T. Ringbeck. Smart pixel–photonic mixer device (PMD). In: *Proc. Int. Conf. on Mechatron. & Machine Vision*, volume 113 (1998).

Zhang, Z. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22(11)**, pp. 1330–1334 (2000).

Appendices

Appendix A

Industrial Specifications and Support

List of industrial specifications:

- Working distance: 50 cm to 100 cm.
- Stationary Camera.
- Fruit grader moving at $1\,\mathrm{m\,s^{-1}}$.
- Field of View: 50 cm to 80 cm $\times$ 20 cm.
- Preferably eye safe at 10 cm to 30 cm.

29 August 2014
Re: Sean Charleston

To whom it may concern,

Compac Sorting Equipment is a global supplier of sorting technology for the fresh produce industry, and winner of the NZ Hi-Tech awards in 2010 and 2012. A key technology component in our systems is the computer vision system that optically inspects produce at high speed and makes market relevant decisions on quality and grade for every piece of produce.

As a natural product every piece of produce is different and will behave differently while travelling and rotating at 10 pieces per second through our system. Understanding the different behaviors of each piece is the biggest challenge facing this technology, the system must understand the exact position and orientation of the product being inspected in order to not mistake normal parts of the produce (eg. The stem) for a defect, and incorrectly rejecting that piece. Many years of ongoing research have shown that 3D imaging is a critical piece of the solution to this problem, and Compac is investing significantly in various 3D imaging techniques as we continuously improve this key aspect of our system and therefore the value of our solution to the market.

Time-of-flight technology represents a significant break from the traditional computer vision based 3D techniques that are appropriate to our problem, but currently the technology is hampered by our need to take four images to resolve depth. Our problem is uniquely challenging in that the product is both moving and rotating at high speed. The research project that is proposed to be undertaken by Sean Charleston would, if successful, remove a significant barrier to the integration of Time-of-flight technology into the Compac system, with a camera system that is novel in our industry whilst providing New Zealand with unique Intellectual Property in this exciting and growing field.

Sincerely,

Ken Moynihan
Vision Systems Development Manager

Compac Sorting Equipment Limited
11 Spring Street P O BOX 13516 Onehunga Auckland New Zealand
Phone +64 9 634 0088 Fax +64 9 634 4491 Web www.compacsort.com

Appendix B

Relevant Code

```
% Motion correction algorithm code. Performs the motion correction on the
% phase steps in 'raw', comparing them to the phase steps in 'original'.
%
% Sean Charleston
% 24/7/2015
%
% The University of Waikato

%% Setup

% Load workspaces containing the lookup tables for calibration
load('calibration_variables.mat');
load('calimg.mat');

% The demodulation requires motion affected data to be in the form of
%   raw(:,:,n), where n is the nth phase step.
% A variable 'original' in the same format of raw is also required, which
%   is the stationary image to compare the motion corrected object to.

[HEIGHT, WIDTH] = size(raw(:,:,1));

display = false;  % Toggle to display various plots for analysis

reverse = false;  % Toggle to determine the direction of ddemodulation.
                  % The SR4000 outputs phase steps in reverse order
                  % (0, -pi/2,...).

interp = true; % Toggle to activate interpolation. The variable
    'pixelshift'
              % is used in the case without interpolation, while the
              % variable 'increment is used with interpolation.

%% Display Raw Phase Steps

if (display)
    MIN = min(min(raw(:,:,1)));
    MAX = max(max(raw(:,:,1)));
    for i = 2:4 % Find color limits for display
        temp = min(min(raw(:,:,i)));
```

```
        if temp < MIN
            MIN = temp;
        end
        temp = max(max(raw(:,:,i)));
        if temp > MAX
            MAX = temp;
        end
    end

    figure;
    subplot(2,2,1);
    imagesc(raw(:,:,1),[MIN MAX]);
    title('Tap 1');

    subplot(2,2,2);
    imagesc(raw(:,:,2),[MIN MAX]);
    title('Tap 2');

    subplot(2,2,3);
    imagesc(raw(:,:,3),[MIN MAX]);
    title('Tap 3');

    subplot(2,2,4);
    imagesc(raw(:,:,4),[MIN MAX]);
    title('Tap 4');

    colormap(gray)
end

%% Background correction

% Subtract the dar current and background dependent amplitude offsets from
% the raw data.
for i = 1:4
    bgrm(:,:,i) = raw(:,:,i) - dark - bgamp(:,:,i);
end

%% Display Binary Images

if (display)
    B1 = abs(raw(:,:,1) - raw(:,:,2));
    B2 = abs(raw(:,:,1) - raw(:,:,3));
    B3 = abs(raw(:,:,1) - raw(:,:,4));
    figure;
    subplot(2,2,1);
    imagesc(B1);
    title('B1')
    subplot(2,2,2);
    imagesc(B2);
    title('B2')
    subplot(2,2,3);
    imagesc(B3);
    title('B3')
    colormap(gray)
end
```

```
%% Standard Demodulation

% Run a standard demodulation on the raw data (no motion correction) for
% comparison.
if(reverse)
    demod.standard = demodulate_reverse(raw);
else
    demod.standard = demodulate(raw);
end

%% Original Demodulation

% Run a standard demodulation on the 'original' data that the motion
% corrected data will be compared to.
demod.still = demodulate(original);


%% Correct Amplitude

% Apply the radial amplitude correction
for i = 1:4
    bgrm(:,:,i) = bgrm(:,:,i).*radialamp;
end

%% Phase Shift

pixelShift = [0 5 10 15]; % Vector for determining the number of pixels
                          % to shift each phase step.

% Shift each of the phase steps.
for i = 1:4
    shifted(:,:,i) = [bgrm(:,(pixelShift(i)+1):WIDTH,i)
        zeros(HEIGHT,pixelShift(i))];
end

% Demodulate the raw phase steps (this represents the naive motion
% correction technique from Hussman et al.
if(reverse)
    demod.naive = demodulate_reverse(shifted);
else
    demod.naive = demodulate(shifted);
end

% Erase edge pixels (now nonsense after shifting).
demod.naive(:,WIDTH - pixelShift(4):WIDTH) = NaN;


%% Interpolation

% This overrides the 'pixelShift' correction when interpolation is on.
if (interp)
    [X,Y] = meshgrid(1:WIDTH,1:HEIGHT);
    increment = 2.8;
    for i = 1:4
        shifted(:,:,i) = interp2(bgrm(:,:,i),X +
            increment*(i-1),Y,'linear',0);
```

```
    end
end

%% Find Phase Error

% Generate fixed pattern phase correction.
perr = (skewPattern.cam30 + 0.057 + FPNPattern.cam30);

% Interpolate the fixed pattern phase for interpolation.
if (interp)
    for i=1:4
        intperr(:,:,i) = interp2(perr,X + increment*(i-1),Y,'linear',0);
    end
end

% Generate the additional exponential phase term to add to the
% demodulation.
eperr = zeros(HEIGHT,WIDTH);
for i = 1:HEIGHT
    for j = 1:(WIDTH-pixelShift(4))
        for k = 1:4
            if (interp)
                eperr(i,j) = eperr(i,j) + exp(1i*2*intperr(i,j,k));
            else
                eperr(i,j) = eperr(i,j) + exp(1i*2*perr(i,j +
                    pixelShift(k)));
            end
        end
    end
end

%% Delta Offset Demodulation

% Apply the modified demodulation technique.
demod.complex = zeros(HEIGHT,WIDTH);
for i=1:HEIGHT
    for j=1:(WIDTH-pixelShift(4))
        for k=1:4
            if (interp)
                offset = exp(1i*intperr(i,j,k));
            else
                offset = exp(1i*perr(i,j+pixelShift(k)));
            end
            if(reverse)
                step = exp(-1i*(k-1)*(-pi/2));
            else
                step = exp(-1i*(k-1)*(pi/2));
            end
            demod.complex(i,j) = demod.complex(i,j) +
                shifted(i,j,k)*step*offset;
        end
    end
end

% Correct output for added exponential phase term.
demod.improved = demod.complex./eperr;
```

```
%% Display Images (2D)

if(display)
    temp1(:,:,1) = mod(angle(demod.naive),2*pi);
    temp1(:,:,2) = mod(angle(demod.improved),2*pi);
    temp1(:,:,3) = mod(angle(demod.still),2*pi);
    MIN = min(min(mod(angle(demod.standard),2*pi)));
    MAX = max(max(mod(angle(demod.standard),2*pi)));

    for i = 1:3
        temp2 = min(min(temp1(:,:,i)));
        if temp2 < MIN
            MIN = temp2;
        end
        temp2 = max(max(temp1(:,:,i)));
        if temp2 > MAX
            MAX = temp2;
        end
    end

    figure;
    subplot(2,2,1);
    imagesc(mod(angle(demod.standard),2*pi),[MIN MAX]);
    title('No Motion Correction')

    subplot(2,2,2);
    imagesc(mod(angle(demod.naive),2*pi),[MIN MAX]);
    title('Naive Motion Correction')

    subplot(2,2,3);
    imagesc(mod(angle(demod.improved),2*pi),[MIN MAX]);
    title('Delta Motion Corrected');

    subplot(2,2,4);
    imagesc(mod(angle(demod.still),2*pi),[MIN MAX]);
    title('Still Image');
end

%% Display Images (3D)

if(display)
    limits = [-0.08 0.1 -0.1 0.1 0.65 0.95];

    % Generate an image mask and the 3D naive correction
    figure;
    h_im = imshow(mod(angle(demod.improved),2*pi),[]);
    waitfor(msgbox('Select the sphere region'))
    e = imrect(gca);
    wait(e);
    BW = createMask(e,h_im);
    masked.naive = (mod(angle(demod.naive),2*pi)*5/(2*pi)).*BW;
    masked.naive(masked.naive == 0) = NaN;
    xyz.naive = transform_xyz(masked.naive,calimg);
    close;
```

```
    figure;
    subplot(2,2,2);
    plot3(xyz.naive.x,xyz.naive.y,xyz.naive.z,'.b','markersize',1);
    title('Naive Motion Correction');
    axis equal;
    view(180,180);
    xlim(limits(1:2));ylim(limits(3:4));zlim(limits(5:6));
    xlabel('x');ylabel('y');zlabel('z');

    % Apply the factory calibrations to the data with no motion correction
    % (and display)
    temp =
        calibrate_data(mod(angle(demod.standard),2*pi),[],'30','MESA','offset');
    temp = calibrate_data(temp.dataPhaseOut,[],'30','MESA','skew');
    temp = calibrate_data(temp.dataPhaseOut,[],'30','MESA','FPN');
    masked.standard = temp.dataPhaseOut*5/(2*pi).*BW;
    masked.standard(masked.standard == 0) = NaN;
    xyz.standard = transform_xyz(masked.standard,calimg);

    subplot(2,2,1);
    plot3(xyz.standard.x,xyz.standard.y,xyz.standard.z,'.b','markersize',1);
    title('No Motion Correction');
    axis equal;
    view(180,180);
    xlabel('x');ylabel('y');zlabel('z');

    % Generate and display 3D image for the improved motion correction
    % technique
    masked.improved = (angle(demod.improved)*5/(2*pi)).*BW;
    masked.improved(masked.improved == 0) = NaN;
    xyz.improved = transform_xyz(masked.improved,calimg);

    subplot(2,2,3);
    plot3(xyz.improved.x,xyz.improved.y,xyz.improved.z,'.b','markersize',1);
    title('Motion Correction (new)');
    axis equal;
    view(180,180);
    xlim(limits(1:2));ylim(limits(3:4));zlim(limits(5:6));
    xlabel('x');ylabel('y');zlabel('z');

    % Generate and display 3D image for the stationary reference image
    masked.still = (angle(demod.still)*5/(2*pi)).*BW;
    masked.still(masked.still == 0) = NaN;
    xyz.still = transform_xyz(masked.still,calimg);

    subplot(2,2,4);
    plot3(xyz.still.x,xyz.still.y,xyz.still.z,'.b','markersize',1);
    title('Still Image');
    axis equal;
    view(180,180);
    xlim(limits(1:2));ylim(limits(3:4));zlim(limits(5:6));
    xlabel('x');ylabel('y');zlabel('z');
end

%% Generate Statistics
```

```
projX = any(BW, 1); % projection of mask along x direction
projY = any(BW, 2); % projection of mask along y direction
fx = find(projX, 1, 'first'); % first column with non-zero val in mask
tx = find(projX, 1, 'last'); % last column with non-zero val in mask
fy = find(projY, 1, 'first'); % first row with non-zero val in mask
ty = find(projY, 1, 'last'); % last row with non-zero val in mask
cropRect = [fx, fy, tx-fx, ty-fy];

% Crop to mask
cropped.still = imcrop(masked.still, cropRect);
cropped.naive = imcrop(masked.naive, cropRect);
cropped.improved = imcrop(masked.improved, cropRect);
cropped.standard = imcrop(masked.standard, cropRect);

% Generate RMSEs
stat.rmse.improved = sqrt(mean(mean((cropped.still -
    cropped.improved).^2)));
stat.rmse.standard = sqrt(mean(mean((cropped.still -
    cropped.standard).^2)));
stat.rmse.naive = sqrt(mean(mean((cropped.still - cropped.naive).^2)));

% Generate SDs
stat.sd.improved = std2(cropped.still - cropped.improved);
stat.sd.standard = std2(cropped.still - cropped.standard);
stat.sd.naive = std2(cropped.still - cropped.naive);
```