A LINE-SCAN SYSTEM FOR THE INSPECTION AND MEASUREMENT OF CYLINDRICAL SURFACES

A. N. Zographos, B.Eng., M.Sc.

A thesis submitted in partial fulfilment of the requirements of
The Nottingham Trent University for the degree of
Doctor of Philosophy

Department of Electrical and Electronic Engineering,
The Nottingham Trent University,
Burton Street,
Nottingham.

June 1998

A Line-Scan System for the Inspection and Measurement of Cylindrical Surfaces

A. N. Zographos

ABSTRACT

This thesis describes research into a stereoscopic machine vision system specifically developed for the imaging of objects that have a high degree of cylindrical symmetry.

A preliminary investigation involved the consideration of two imaging scenarios based on the standard area array sensor. The first employed a network of such sensors in order to produce all-round observation of a cylindrical object. The second consisted of a single area array camera imaging a rotating object. It was found, however, that both schemes had a number of limitations which precluded the efficient inspection of cylindrical objects. These limitations were subsequently addressed by using a line-scan sensor and rotating the object to be inspected.

Following this initial investigation, an experimental two-dimensional (2-D) line-scan system employing rotational object motion was constructed. The imaging characteristics were analysed and experiments were conducted to evaluate the 2-D coordinate measurement capability of the technique. It was found that this system possessed the necessary attributes to be used in the extraction of coordinate information from a defined object workspace. The experimental line-scan system was utilised in the imaging of ballistics specimens, such as cartridge cases and fired rounds of ammunition.

The results obtained from this part of the investigation led to the development of a stereoscopic line-scan camera system, which could be employed to extract three-dimensional (3-D) coordinate data from an object of interest. To achieve this, a rigorous calibration technique, based on independent geometry for each camera with respect to both its platform and the object workspace, was devised. The resultant mathematical model was extended to encompass the calibration of all the critical parameters of the stereoscopic system.

Experiments were conducted to validate the system model and to determine the reliability of the analytical procedures applied when performing 3-D measurements. Further experiments were undertaken to characterise the spatial resolving properties of the stereoscopic system and evaluate its coordinate measurement accuracy. Utilising the existing hardware, the experiments indicate a spatial accuracy of 0.2mm to 0.4mm in all three coordinate axes at a range of 1.5m.

ACKNOWLEDGEMENTS

I would like to thank the following people for their help during this work.

My supervisors, Dr. Paul Evans, Professor Max Robinson and Dr. Simon Godber for their technical input and advice regarding this thesis.

Gary Griffiths for calibrating the control fields.

Katerina, my family and friends for their irreplaceable support.

TABLE OF CONTENTS

1 INTRODUCTION 1

1.1 Research Objectives 2
1.2 Previous Related Work. 2
1.3 Structure of the Thesis 3

2 CYLINDRICAL OBJECT IMAGING 5

2.1 Introduction 5
2.2 Limitations of Area Array Sensors in Cylindrical Object Imaging 5
2.3 The 'Multiple View' Area Array System 11
2.4 The Rotating Object Line-Scan System 14
2.5 The Line-Scan Camera 19
2.5.1 Principles of Operation 19
2.5.2 Spatial Oversampling and Undersampling 23
2.5.3 Image Aspect Ratio 25
2.5.4 Factors Limiting the Spatial Resolution of a Line-Scan Image. 27

3 THE TWO-DIMENSIONAL SYSTEM 30

3.1 Introduction 30
3.2 The Experimental System 30
3.2.1 The Line-Scan Cameras. 31
3.2.2 The Video Combiner Circuit 33
3.2.3 The Frame Grabbers 35
3.2.4 The Stepper Motor Controller 36
3.2.5 The Basewidth/Convergence Stage 38
3.3 Host Controller and System Operation 38

3.4 Sample Images 41
3.5 A Ballistics Application 45
3.6 Algorithms for the Two-Dimensional System 47
3.6.1 Geometry of the Two-Dimensional System 47
3.6.2 X Axis Algorithm Derivation 48
3.6.3 Y Axis Algorithm Derivation 50
3.6.4 Determining the Conditions for Correct Image Aspect Ratio 52
3.7 Two-Dimensional System Experiments 53
3.7.1 Experimental Strategy and Error Representation 54
3.7.2 Establishing the Repeatability of System Parameters 55
3.7.3 Validation of the Mathematical Model 60
3.7.3.1 . X Axis Algorithm Verification 60
3.7.3.2 Y Axis Algorithm Verification 64
3.7.4 Discussion of the Experimental Results 69

4 THE STEREOSCOPIC SYSTEM 71

4.1 Introduction 71
4.2 The Stereoscopic Region 72
4.3 The Spatial Sampling Pattern. 74
4.4 Stereoscopic Line-Scan System Algorithms 76
4.4.1 Geometry of the Stereoscopic System 76
4.4.2 The System Model 78
4.5 Implementing the System Model 96
4.5.1 Calibration Implementation Issues. 96
4.5.2 The Requirement for Initial Approximations 99
4.5.3 Space Intersection Implementation Issues 100
4.5.4 Calculating the Spatial Quantisation Error 103
4.5.5 Software Code 104
4.6 Summary 106

5 EXPERIMENTS WITH THE STEREOSCOPIC SYSTEM 107

5.1 Introduction 107
5.2 Preliminary Considerations 107
5.2.1 Feature Matching 107
5.2.2 Subpixelation 108
5.2.3 Target Location 109
5.2.4 Nominal System Alignment. 110
5.3 Control Fields. 111
5.3.1 The Cartesian Control Field 112
5.3.2 The Cylindrical Control Field 114
5.4 Test Area Illumination. 118
5.5 Coordinate Measurement: An Example 119
5.6 Imaging Characteristics 122
5.7 Experiments with Synthetic Data 134
5.7.1 Varying the Amount of Control in the Presence of Noise. 136
5.7.2 Effect of Radial Lens Distortion 138
5.7.3 Projection and Back-Projection Accuracy. 139
5.8 Experiments with Real Data 142
5.8.1 Accuracy in the Measurement of Test Points 143
5.8.2 Accuracy in the Measurement of a Generic Object 148

6 SUMMARY, CONCLUSIONS AND FUTURE WORK 151

6.1 Introduction 151
6.2 The Imaging Concept 151
6.2.1 Motivation 151
6.2.2 The Rotating Object Line-Scan System. 152
6.2.3 The Image Sensors. 153
6.3 The Experimental Two-Dimensional System 154
6.4 The Stereoscopic System 154
6.4.1 The System Model 154

6.4.2 The Experimental Results 156
6.5 Conclusions 160
6.6 Future Work 164
6.6.1 Automating the Calibration Process 164
6.6.2 Improving Accuracy 164
6.6.3 Further Investigation of the Line-Scan Sensor 165

REFERENCES 167

APPENDIX I
The Rigid Transformation and the Rotation Matrix I-1

APPENDIX II
Extension of the System Algorithms to Account for Radial Lens Distortion II-1

APPENDIX III
The Partial Derivatives of the System Algorithms III-1

APPENDIX IV
Software Code IV-1

APPENDIX V
Control Field Calibration Data V-1

LIST OF FIGURES

2-1. Area array sensor imaging a cylindrical object 5
2-2. Geometry of an area array sensor imaging a cylindrical object 7
2-3. Loss of image resolution over one quadrant due to object cylindricity 9
2-4. The area sampled by a row of photosites 10
2-5. Object and image having different aspect ratios 10
2-6. Restricted angular field of view 11
2-7. A network of area array cameras imaging a cylindrical object 12
2-8. The combined area sampled by four corresponding rows of photosites 13
2-9. An area array camera imaging a rotating object 14
2-10. A line-scan camera imaging a cylindrical object 15
2-11. Comparison of the spatial sampling patterns 16
2-12. Object and camera rotation modes 18
2-13. The line-scan camera: front and perspective views 19
2-14. Simplified line-scan timing diagram 20
2-15. Line-scan camera instantaneous field of view 21
2-16. Two-dimensional line-scan image production 22
2-17. Effective spatial width of a scan line 24
2-18. Spatial oversampling and undersampling 24
2-19. The effect of the integration period on image aspect ratio 26
2-20. The effect of increasing range on image aspect ratio 27
2-21. Spatial sampling effects 29
3-1. Block diagram of the system 31
3-2. LC1902 timing diagram 32
3-3. Schematic diagram of the video combiner circuit 34
3-4. Block diagram of the P360F frame grabbers 35
3-5. Camera positional controller 38
3-6. System operation flowchart 39
3-7. A cylindrical object as imaged by an area array camera 41
3-8. The spray can as imaged by the line-scan system 42
3-9. "Electronic zooming" in the X axis 42

3-10. Area array image of a chuck 43
3-11. All-round image produced by the line-scan system 44
3-12. Line-scan image of chuck depicting "correct" pixel aspect ratio 44
3-13. Area array image of a cartridge case 45
3-14. The image obtained by the line-scan system 46
3-15. Geometry of the two-dimensional system 47
3-16. Points symmetrically located about the Z axis 50
3-17. Block diagram of the rotational speed precision measurement system 56
3-18. Spatial angle measurements from variations of the integration period 61
3-19. Spatial angle measurements from variations of the rotational speed 62
3-20. Spatial residual error from variations of the angle ω 63
3-21. Spatial residual error from variations of Y_P 66
3-22. Spatial residual error from variations of r 68
3-23. Spatial residual error from variations of the depth Z_C 69
4-1. The stereoscopic region 73
4-2. Evaluating the 'far' boundary Z_F 74
4-3. The spatial sampling pattern produced after object rotation 75
4-4. Geometry of the stereoscopic system 77
4-5. Left and right perspective images depicting disparity 78
4-6. Detail of the left camera geometry 79
4-7. Detail of the right camera geometry 85
4-8. Space intersection 94
4-9. Determining the unknown point's radius 101
4-10. Determining the unknown point's angle 102
4-11. Flow chart of spatial quantisation error calculation 104
5-1. Nominal stereoscopic system alignment 110
5-2. The Cartesian control field 113
5-3. Side view of the cylindrical control field 115
5-4. Plan view of the cylindrical control field 116
5-5. Plan view of target distribution in the cylindrical control field 117
5-6. The test area 118
5-7. The calibration code showing sample calibration results 120

5-8. Variance of the calibration parameters 121
5-9. Other calibration data 121
5-10. Space intersection results 122
5-11. Left and right camera images having no stereoscopic parallax. 123
5-12. Left and right camera images depicting stereoscopic parallax 123
5-13. A set of collinear targets imaged by the line-scan system 125
5-14. Superimposed left and right perspective images 125
5-15. Graph of left and right image X axis frame buffer coordinates against radius for a single set of collinear points 126
5-16. X axis frame buffer coordinates against radius 127
5-17. X axis frame buffer coordinates against angle 128
5-18. Graph of parallax Δx_f against angle. 129
5-19. Y axis frame buffer coordinates against Y axis spatial distance 130
5-20. Radial and angular uncertainty 131
5-21. Y axis uncertainty 131
5-22. Graph of parallax Δx_f against radius with fitted tangents 132
5-23. Graph of estimated and calculated radial uncertainties against radius 133
5-24. Flow chart of calibration tests 136
5-25. The MNCE as a function of the number of control points 139
5-26. Projection and back-projection tests 141

1. INTRODUCTION

A commonly utilised electronic imaging device in machine vision is the area array sensor. Historically, this device was developed to produce images that were well-suited to human observation. Although the last two decades have seen the successful introduction of such cameras in industrial inspection[1,2,3,4,5,6,7], robot vision and control applications[8,9,10,11,12,13,14], area array cameras do not necessarily represent the best choice for all machine vision applications[15,16,17,18]. Indeed, sensor choice is critical in all vision systems and the suitability of different types of sensor must be thoroughly assessed according to the specific requirements of a particular application.

The imaging of cylindrical objects presents a number of problems when conventional imaging techniques, based on area array cameras, are employed. It is for this reason that techniques specifically developed for the imaging of such objects have been developed in the past. One such technique dates back to the 1940's and involves rotating the object to be inspected in front of a photographic camera employing a slit aperture[19]. The research presented in this thesis replaces the photographic camera with a line-scan camera system. In addition, the development of a stereoscopic variant of this system is presented.

The line-scan sensor consists of a single column of photosensitive elements, thus producing an essentially one-dimensional image at any instant in time. In order to obtain a two-dimensional image, relative motion between the sensor and the object to be imaged must exist. The two-dimensional images produced by a line-scan device have different characteristics from those of an area-array sensor, as their projective geometry is fundamentally different. In particular, a line-scan sensor realises the *perspective* projection in the sensor axis, but the projection in the motion axis is *orthographic*, that is, range invariant. By exploring these properties, the inspection of cylindrical objects by means of a line-scan camera offers a number of unique advantages over the area array sensor. These include homogeneously distributed spatial resolution over the whole of the object's surface, and an all-round, i.e. 360 degree, inspection of the object in a single image. In addition, the cylindrical surface of the object is 'unfolded' into a planar surface, which is significantly more efficient to process.

1.1 Research Objectives

The principal objective of this work is to develop an efficient machine vision system well-suited to the inspection of objects that have a high degree of cylindrical symmetry. In order to achieve this, the following objectives were set:

i. the investigation into the suitability of different types of sensor formats and operating modes in the context of cylindrical object imaging;

ii. the development of a rotating object two-dimensional line-scan system and the analysis of its imaging characteristics;

iii. the development of a stereoscopic variant of the two-dimensional line-scan system and the analysis of its imaging characteristics;

iv. the derivation of a rigorous calibration technique and the implementation of space intersection (triangulation) algorithms;

v. the experimental verification of the stereoscopic system's mathematical model.

1.2 Previous Related Work

Continuing work within the 3-D Imaging Group of The Nottingham Trent University has focused on the utilisation of alternative sensors to the area array imager. Initially, an investigation into the coordinate measurement potential of the position sensitive detector (PSD) and the line-scan camera was undertaken[20]. It was found that the line-scan camera possesses the necessary attributes of consistency and spatial resolving power to be used in dimensional measurement. Subsequently, a stereoscopic line-scan system employing lateral object motion was developed. The spatial resolving characteristics of this system were ascertained, leading to the derivation of algorithms for the extraction of three-dimensional coordinate information from an object workspace. This was done under the assumption of perfect system alignment regarding both the relative position of the two line-scan sensors and the stereoscopic camera arrangement to the reference object space coordinate system.

To facilitate measurement in object space, the horizontal and vertical axis object to image space scale factors were determined using simple geometrical concepts. Such a system could be employed to inspect objects in applications where linear motion is inherent, such as production lines.

The next line of research involved the development of a stereoscopic line-scan system in which an uncalibrated stereo-camera was rotated[21]. Such an arrangement could acquire omni-directional, i.e. panoramic, stereoscopic images for applications such as intruder detection, autonomous robot navigation and the production of virtual reality models of scenes. Investigation into the spatial resolving properties of this system lead to the development of coordinate measurement algorithms. These were based on Euclidean geometry and perfect alignment was assumed between the two cameras and their platform. However, a local object space coordinate system was established by implementing a coordinate systems' transformation from the stereo-camera frame to the object space coordinate system.

1.3 Structure of the Thesis

The organisation of this report is as follows.

Chapter 2 discusses the decision to utilise the line-scan sensor in this work. The limitations of conventional imaging techniques when used for the inspection and dimensional measurement of cylindrical objects are identified. The development of the rotating object line-scan system, which addresses these limitations, is subsequently presented. The chapter concludes with a description of the line-scan camera and its operating characteristics.

Chapter 3 details the design of the experimental system and presents a number of sample images that demonstrate its imaging properties. A ballistics application utilising the rotating object line-scan system is then presented, followed by the development of a coordinate measurement mathematical model for the two-dimensional line-scan system. Experimental work carried out to establish the integrity of the line-scan system and verify the validity of the mathematical model concludes this chapter.

Chapter 4 presents a theoretical appraisal of the stereoscopic line-scan system. The derivation of the calibration model is presented, followed by the application of space intersection algorithms to extract three-dimensional coordinate data. Issues pertaining to the implementation of the above processes and the quality of the produced data are considered.

The experimental strategy adopted to evaluate the coordinate measurement accuracy of the stereoscopic system and verify the validity of the system model is given in Chapter 5. Tests designed to analyse the imaging characteristics of the stereoscopic rotating object line-scan system are also presented.

Chapter 6 contains a discussion of the results and the conclusions drawn from the work presented in the previous chapters. The areas where additional work may be carried out to improve the accuracy of the system are also identified. The thesis concludes with suggestions for future research.

2. CYLINDRICAL OBJECT IMAGING

2.1 Introduction

The fundamental aim of this research is to develop an efficient machine vision system for the all-round inspection and dimensional measurement of cylindrical objects. This chapter investigates the imaging techniques that could be employed to facilitate these tasks.

Initially, the standard area array sensor is investigated in the context of cylindrical object imaging. The imaging characteristics of a vision system employing a number of such sensors to image the complete surface of a cylindrical object are then described. It is shown that this arrangement has inherent limitations that complicate the imaging of cylindrical objects and produce suboptimal results. An imaging system that addresses these limitations is subsequently presented. This utilises a line-scan sensor and rotation of the object to be inspected.

Following this, the principles of operation of the line-scan sensor are examined in more detail and the imaging characteristics of this device are discussed.

2.2 Limitations of Area Array Sensors in Cylindrical Object Imaging

A perspective view of an area array sensor imaging a cylindrical object is illustrated in Figure 2-1.

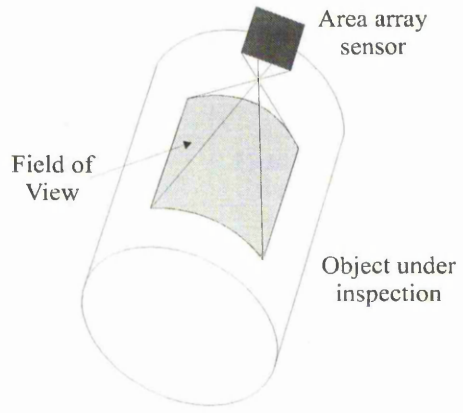

Figure 2-1 *Area array sensor imaging a cylindrical object.*

Although this discussion assumes that the image sensor consists of discrete photosensitive elements of finite size, analogous concepts are also applicable to a photographic camera imaging a cylindrical object.

With reference to Figure 2-1, the following points can be made:

i. the spatial resolution defined on the circumference of the cylindrical object, referred to as the *circumferential* resolution, decreases non-linearly;

ii. if the aspect ratios of the area array sensor and the object under inspection are different, part of the image area cannot be utilised;

iii. surfaces of high reflectivity can give rise to specular reflections that cannot be readily controlled;

iv. the maximum achievable angular field of view is restricted by the geometry of the sensor and the perspective projection realised by the camera optics.

These issues are discussed in more detail in the following text.

The non-linear decrease in the circumferencial resolution is produced by two factors:

- each picture column lies at a different range, as corresponding distances from the lens perspective centre to the surface of the object are a function of the object's curvature;
- the perspective projection through a single point in conjunction with the object's curvature gives rise to oblique views.

To illustrate the above effects, Figure 2-2 depicts a plan view of an area array camera imaging a cylindrical surface.

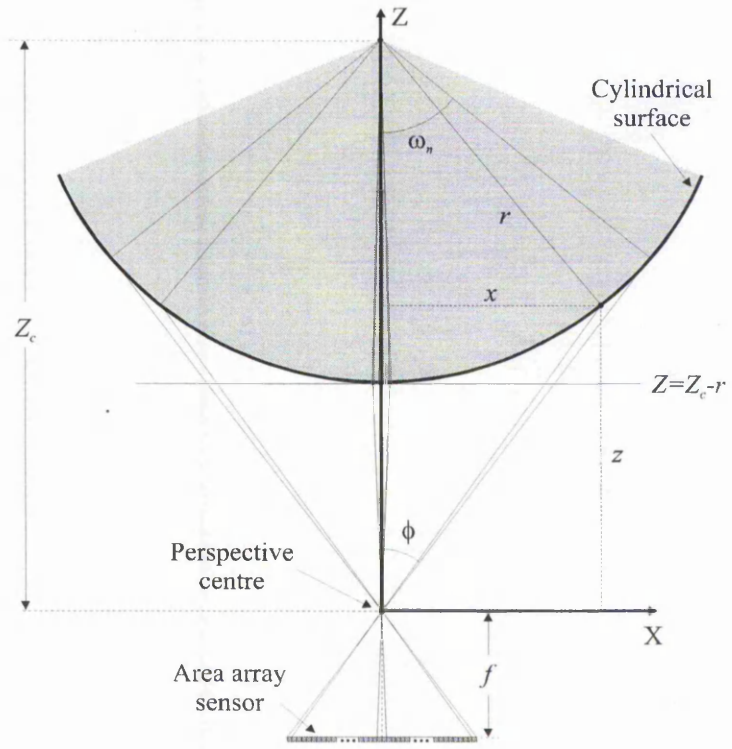

Figure 2-2 *Geometry of an area array sensor imaging a cylindrical object.*

The cylindrical object of radius r is located distance Z_c from the perspective centre. The lens principal distance, i.e. the perpendicular distance from the lens perspective centre to the image plane at a given lens focus distance, is denoted by f. The sensor consists of rows of n photosites, and each photosite has a width, defined along the X axis, of dx. The Z axis denotes range and is coincident with the lens' optical axis. The optical axis and the main cylinder axis are normal.

The spatial resolution of this system over the imaged area of the object is a maximum at a plane parallel to the sensor at a range $Z = Z_c - r$. This is because at this range the surface of the object is closest to the lens perspective centre and hence the spatial sampling interval has its smallest value. However, as the angle of subtendence φ increases tangentially with photosite separation from the optical axis, there is a corresponding symmetrical reduction in spatial resolution available. This effect can be quantified by the following analysis.

From Figure 2-2, the X axis coordinate of a point on the circumference of the circle can be expressed as -

$$x = z\tan\varphi \tag{2-1}$$

$$x = (Z_c - z)\tan\omega$$

Eliminating z produces -

$$x = UZ_c \tag{2-2}$$

where -

$$U = \left(\frac{1}{\tan\varphi} + \frac{1}{\tan\omega}\right)^{-1} \tag{2-3}$$

The equation of the circle of radius r is -

$$r^2 = x^2 + (Z_c - z)^2$$

Substituting equation 2-1 in the above expression produces -

$$r^2 = \left(1 + \frac{1}{\tan^2\varphi}\right)x^2 + \left(\frac{-2Z_c}{\tan\varphi}\right)x + Z_c^2$$

Incorporating equation 2-2 gives -

$$U^2\left[Z_c^2\left(1 + \frac{1}{\tan^2\varphi}\right)\right] + U\left(\frac{-2Z_c^2}{\tan\varphi}\right) + \left(Z_c^2 - r^2\right) = 0$$

The above quadratic is solved in U, and angle ω is obtained from equation 2-3 as -

$$\omega = \tan^{-1}\left(\frac{1}{U} - \frac{1}{\tan\varphi}\right)^{-1}$$

where the angle φ is given by -

$$\varphi = \tan^{-1}\left(\frac{n\delta x}{2f}\right)$$

The arc $\vec{a}_i$ corresponding to a back-projected photosite i, $i \to \{1\ldots n\}$, is then calculated as -

$$\vec{a}_i = r \cdot \omega_i$$

where ω_i is the angle corresponding to photosite i. A *resolution figure-of-merit* is established as -

$$f-o-m = c\left[\ddot{a}_i - \ddot{a}_{i-1}\right]^{-1}$$

where c is a normalising constant equal to -

$$c = \left[100\ddot{a}_1\right]^{-1}$$

and $\ddot{a}_1$ is the arc corresponding to the outermost photosite of the area array sensor.

The following example utilises the above procedure. A realistic projection model is adopted by assigning the following values to the geometrical parameters of Figure 2-2: a sensor of 512 photosites per row, having a width of $13\mu m$ each, a lens principal distance of 50mm and a distance from the lens perspective centre to the axis of cylindrical symmetry of approximately 1.5m. The object's radius is chosen such that the outermost vertical scan line is approximately tangential to the surface of the object. The resolution figure-of-merit, normalised to the maximum achievable resolution for the particular example above is plotted in the graph of Figure 2-3.

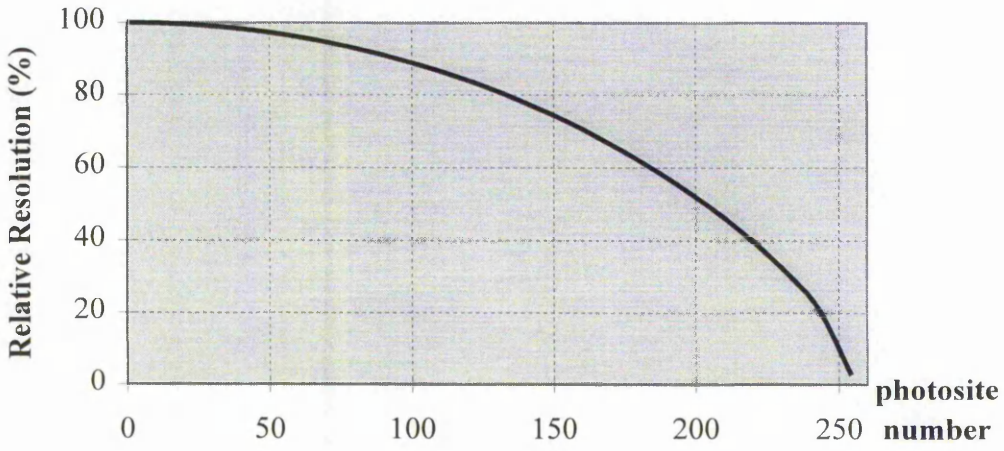

Figure 2-3 *Loss of image resolution over one quadrant due to object cylindricity.*

From the above plot it can be seen that a rapid decrease in image resolution occurs at picture columns of increasing separation from the optical axis.

Furthermore, photosite columns of increasing separation from the centre of the sensor sample parts of the object at increasing range. Consequently, the spatial sampling interval in the Y image axis (parallel to the object's axis of cylindrical symmetry) is also a function of the cylindricity of the object under inspection. The resultant area sampled by a row of photosites is illustrated in Figure 2-4.

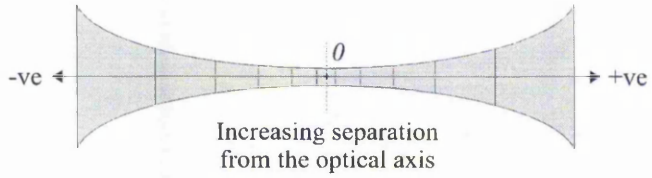

Figure 2-4 *The area sampled by a row of photosites.*

From this figure it can be appreciated that the size of the photosites back-projected on the surface of the object increases in *both* imaging axes with increasing separation from the optical axis.

A further limitation can be observed in this system. The aspect ratio of the area array sensor is predefined by the camera manufacturer and it usually conforms to either the television imaging standard of 4:3 or the metric standard of 1:1. However, when the aspect ratio of the object to be imaged is dissimilar to that of the sensor, a significant part of the image area cannot be utilised. This is illustrated in Figure 2-5, where the sensor, and hence the image, have an aspect ratio $X_{img}:Y_{img}$ of unity, but that of the object $X_{obj}:Y_{obj}$ is much smaller. Under such conditions, the fixed geometry of the area array sensor becomes a limiting factor.

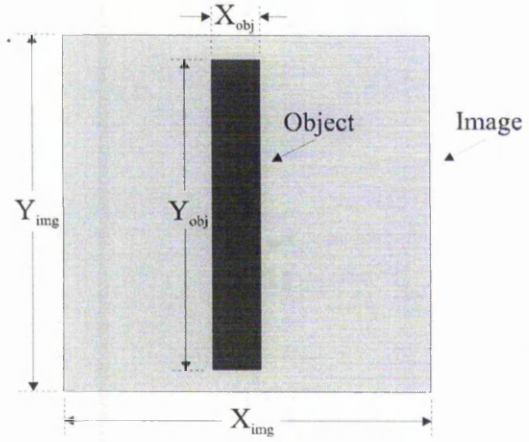

Figure 2-5 *Object and image having different aspect ratios.*

Illumination is of critical importance in machine vision applications as image information is conveyed by the brightness level of individual pixels. Considering that for a highly reflective surface the cylindricity of the object will result in dispersed reflected light rays, potentially uncontrollable light reflections can occur. Such specular reflections are difficult to control and can lead to loss of image information. If, on the other hand, highly diffused light is used, image contrast will suffer.

An all-round observation of a cylindrical object requires a 360^0 angular field of view. However, an area array camera can only image part of the object's surface. Figure 2-6 shows that the maximum angular field of view φ_{max} is restricted by the geometry of the sensor and the perspective projection through a single point.

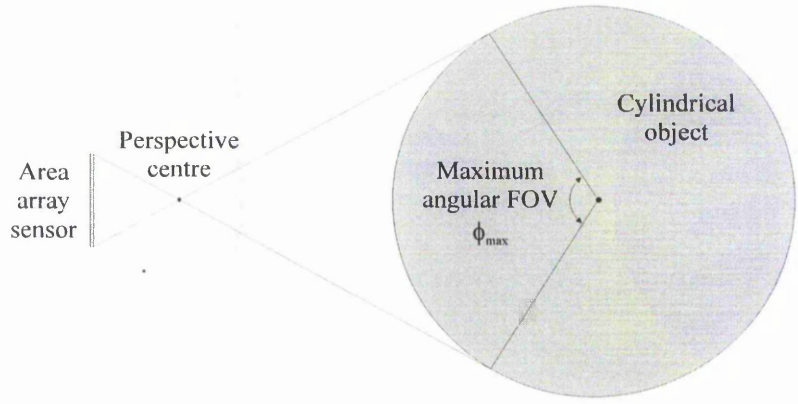

Figure 2-6 *Restricted angular field of view.*

Hence, an all-round observation of the object is not possible unless multiple views, acquired from different perspectives, are employed. This can be achieved by utilising either a network of area array cameras distributed around the periphery of the object or by multiple view acquisition from a single camera synchronised with object rotation.

2.3 The 'Multiple View' Area Array System

Figure 2-7 illustrates a network of four area array cameras imaging a cylindrical object. The cameras are symmetrically distributed at right angles around the periphery of the object. The

field of view of each camera in the horizontal plane, i.e. normal to the object's axis of cylindrical symmetry, is approximately tangential to the surface of the object.

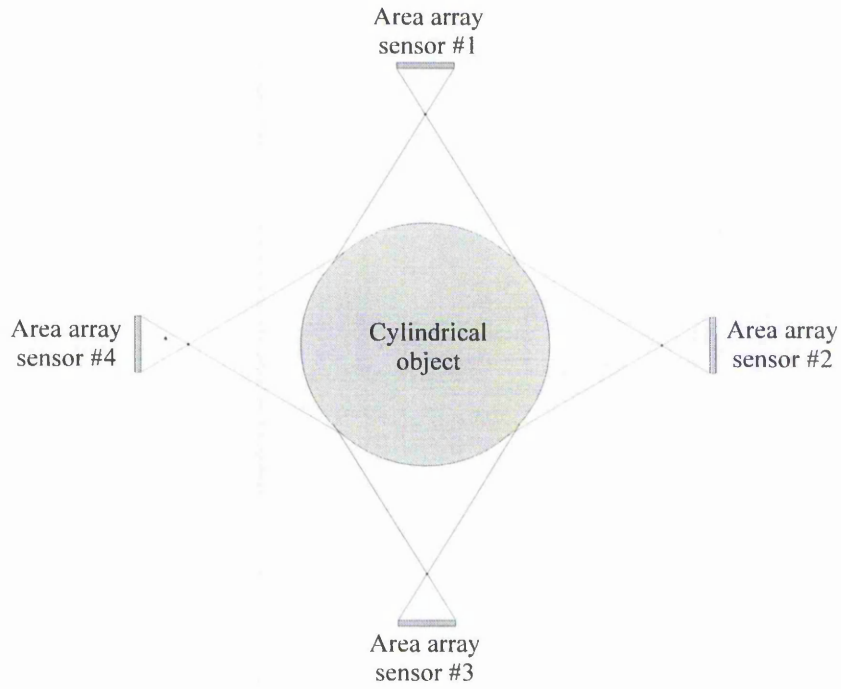

Figure 2-7 *A network of area array cameras imaging a cylindrical object.*

Although such an arrangement can be used to inspect the whole surface of the object, it suffers from the following limitations:

i. cylindrical objects of different diameter require changes in the system set-up;

ii. inspection of the complete surface of the object from a single image requires the production of an image *mosaic*, whereby the separate images are combined together. It can be appreciated that the accurate registration of the individual images will be non-trivial[22];

iii. the circumferential resolution offered by each camera decreases non-linearly (see Figure 2-3).

From Figure 2-7 it can be seen that the optical and geometrical parameters of the multiple view area array system, such as the lens principal distance and the camera to object range, have to be set in accordance with the diameter of the object. However, if an object of larger diameter is imaged, adjacent fields of view will not overlap and part of the object's surface will not be viewed. Similarly, an object of smaller diameter will result in loss of resolution, as the extent of the individual fields of view will be larger than their optimum value.

Figure 2-8 depicts the spatial sampling pattern produced by the same row of photosites in the four area array sensors. The loss in circumferential resolution is indicated by the increasing area of the spatial samples produced by photosites of increasing separation from the optical axis. The horizontal scan line produced by the four sensors suffers from a 'ripple' effect.

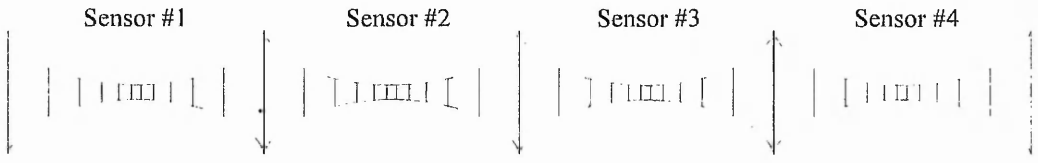

Figure 2-8 *The combined area sampled by four corresponding rows of photosites.*

In order to decrease the ripple and increase the circumferential resolution, the number of views have to be increased. However, this is uneconomical in terms of both the large number of cameras and digital storage memory required. In addition, the production of the image mosaic will require more processing power and time.

A more economical and efficient solution would be to use a single area array camera and rotate the object under inspection. This would allow the inspection of the complete surface of the object in a series of consecutive images acquired in synchronism with object rotation. An example of such a system is illustrated in Figure 2-9.

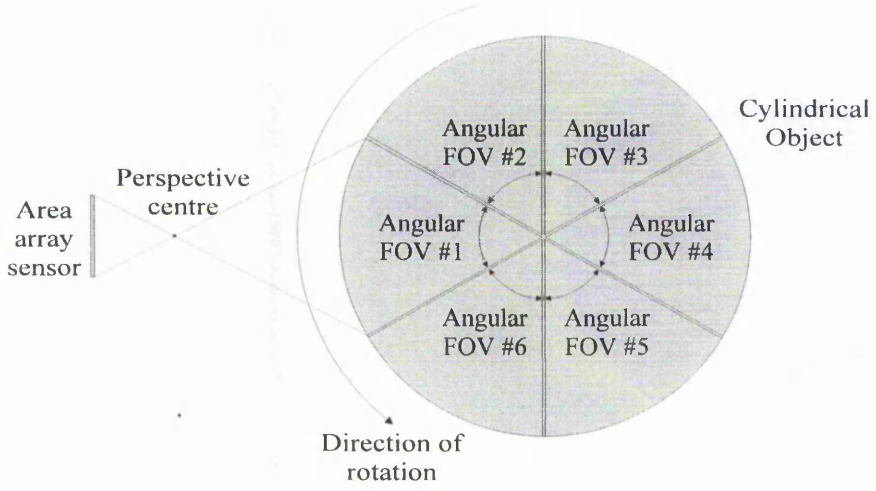

Figure 2-9 *An area array camera imaging a rotating object.*

In the above example, because the acquisition of a 360^0 view of the object is divided into six images having smaller angular fields of view, the ripple in horizontal scan lines is reduced and the circumferential resolution is increased. Although this system can be used to inspect the whole surface of a cylindrical object and partially solves the problem of decreasing circumferential resolution, it still suffers from the remaining complexities discussed in section 2.2. Furthermore, the integration of the large number of individual images produced is expected to be a complicated process. Therefore, an alternative technique is required for the efficient imaging of cylindrical objects.

2.4 The Rotating Object Line-Scan System

In order to address the above limitations, a line-scan system employing rotational object motion is proposed. Although for the purposes of analysis the object under inspection is assumed to be cylindrical, any object that has a degree of cylindrical symmetry is well suited to this system. In the context of this work, 'cylindrical symmetry' refers to any object which is symmetrical about a central axis, but may have unequal diameters at different heights.

The principles of operation of the line-scan camera are discussed in section 2-5. Briefly, the line-scan sensor consists of single column of photosites, thus requiring relative motion

between the sensor and the object to produce a two-dimensional image. The projection properties of this device are different from those of an area array sensor (see section 2.5.1).

The rotating object line-scan system is depicted in Figure 2-10. This system produces a two-dimensional image by a 'scanning' process: consecutive columns of picture information are acquired and stored in a frame buffer whilst the object is being rotated. Both the *instantaneous field of view* (IFOV) of the line-scan camera and the axis of cylindrical symmetry of the object are aligned to be coincident with the rotation axis. The origin of the camera-centred coordinate system has been translated along the optical axis from the perspective centre *0* to the rotation axis, to aid understanding of the drawing's perspective.

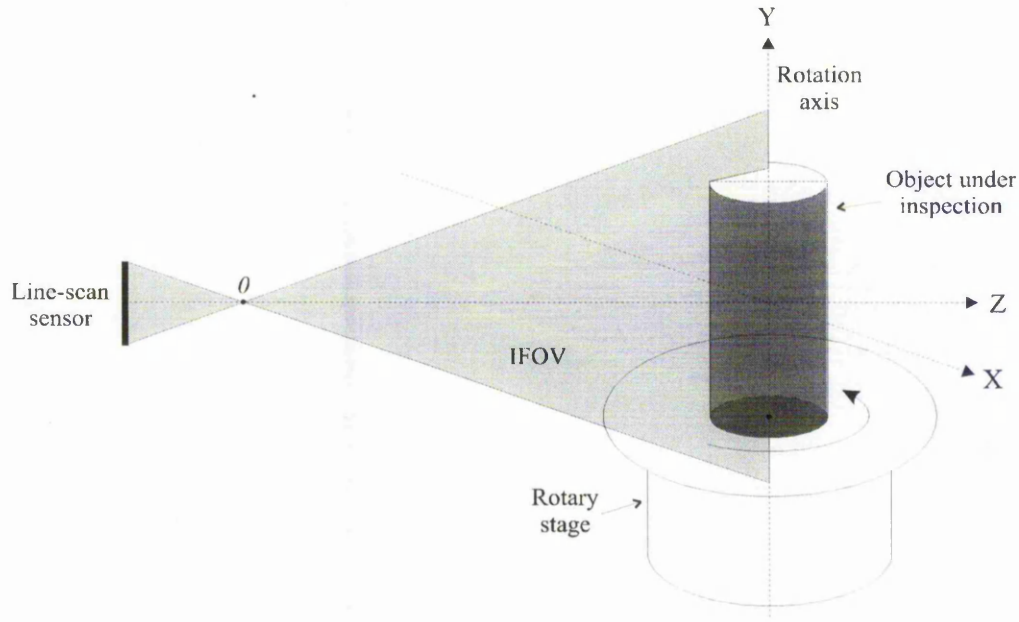

Figure 2-10 *A line-scan camera imaging a cylindrical object.*

The following points pertaining to this system can be made:

i. an all-round view of the object can be readily obtained in a single image;

ii. the circumferential resolution is constant over the inspected surface;

iii. the Y axis field of view can be optimised for a given object height without affecting the angular field of view;

iv. lighting issues may be more easily addressed than in an area array system.

These points are discussed in the following text. From Figure 2-10 it can be seen that, for a nominally cylindrical object, the distance from the lens perspective centre to the object's surface remains constant throughout the object rotation. Hence, assuming that the width of the scan line is much smaller than the radius of the object, for a particular combination of rotational speed and integration period (see section 2.5.3), the back-projected photosites will retain their rectangular shape. As a result, there is no loss of circumferential resolution produced by the cylindricity of the object. To illustrate this point, Figure 2-11 depicts a comparison of the spatial sampling pattern of the multiple view area array and the rotating object line-scan systems.

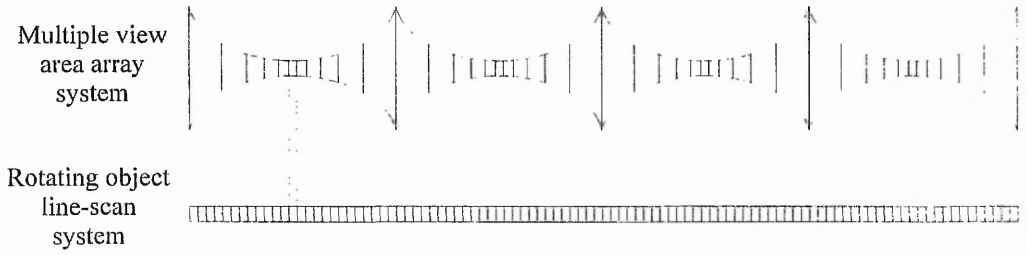

Figure 2-11 *Comparison of the spatial sampling patterns.*

The spatial sampling pattern of the area array system is produced by corresponding rows of photosites in the four sensors (see section 2.3), whereas that of the line-scan system is produced by a single photosite consecutively sampling the object. It can be appreciated that the circumferential resolution of the line-scan system is constant and equal to the maximum resolution offered by the area array system. In other words, the cylindricity of the object does not adversely affect the spatial sampling characteristics of the line-scan system. The resultant image is a *planar view* of the object, in which the whole of the cylindrical surface appears "unfolded" into a flat surface.

As the angular field of view of the line-scan system exists only in the frame buffer memory, enough consecutive scan lines to image the complete surface of the object can be produced if the necessary amount of frame buffer memory exists. This attribute of the line-scan

system is of critical significance in that it solves the potentially involved problem of obtaining a single, continuous image depicting the complete surface of the object.

In section 2.2 it was shown that the fixed geometry of the area array sensor may not suit all applications: efficient imaging of a cylindrical object requires that the angular field of view can be controlled to any value up to 360^0 and that the Y axis field of view is set in accordance with the object's height. As in the line-scan system these two parameters can be controlled independently, maximum utilisation of the system resources can be achieved.

The line-scan system simplifies lighting issues by requiring that lighting of the object under inspection is optimised only along the instantaneous field of view. This is in contrast to the area array camera scenario, where lighting over a whole area on the surface of the object must be optimised.

Having identified the advantages of the rotating object line-scan system, it is important to consider the potential disadvantages of this system. This is because the successful integration of the line-scan system to an application may be inhibited by certain limitations of this system. These limitations can be summarised as follows:

i. due to the mechanical motion, the production of a two-dimensional image, in general, requires a longer duration than that required by the multiple view area array system;

ii. the line-scan system may not be suitable for the inspection of bulky or heavyweight objects;

iii. the higher the dimensional measurement accuracy and the lower the imaging distortion required, the more consistent the rotational speed must be.

The minimum integration period (see section 2.5.1) for line-scan cameras can be as low as $50\mu S$ [23], which is faster than many high-speed area array cameras. Furthermore, a one-dimensional image can be transferred and processed faster than the two-dimensional frames produced by area array cameras. In certain applications, the picture information produced by a line-scan sensor may be processed on a line-by-line basis[24,25,26,27,28,29,30] or in

small two-dimensional frames[31,32,33,34,35,36]. In such applications, due to the reduced amount of picture information, a machine vision system utilising line-scan cameras has a potential speed advantage over an area array camera system. However, other applications, including the rotating object line-scan system and satellite photogrammetry[37], require two-dimensional images consisting of hundreds or thousands of scan lines. As mechanical motion is involved in such applications, the time required to produce the two-dimensional images will, in general, be longer than that required by an area array camera system.

The rotating object line-scan system may not be suitable for the inspection of heavyweight or bulky objects. Although in this research programme the object rather than the camera is rotated, the two modes of operation are interchangeable as far as the resultant images are concerned. This is evident from Figure 2-12, where the circumferential fields of view produced by (a) object rotation and (b) concentric camera rotation are compared.

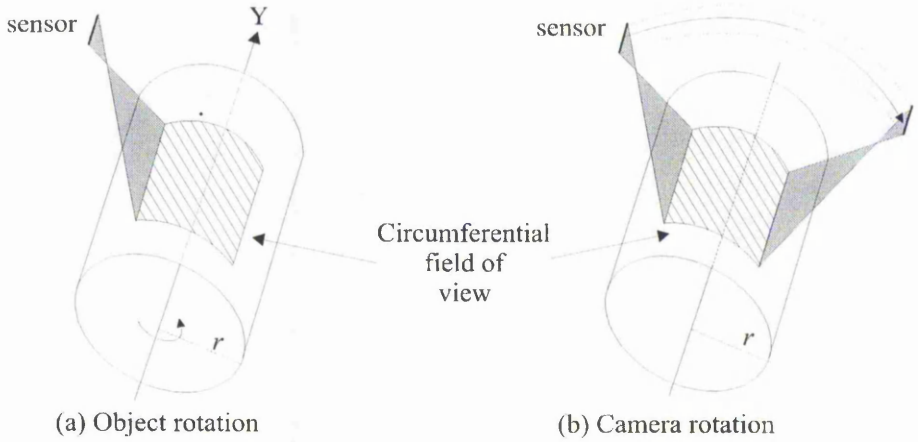

Figure 2-12 *(a) object and (b) camera rotation modes.*

Thus, the arrangement shown in Figure 2-12(b) may be more suitable for imaging of bulky or heavyweight objects, or objects which cannot be moved.

If the rotational speed of the object under inspection is not constant, imaging distortion will occur. Additionally, any variations in the rotational speed will have an impact on the coordinate measurement accuracy and precision of the line-scan system. The experimental work carried out in this work shows that the precision of the rotational speed can be controlled to a high degree. Specifically, in Chapter 5 it is shown that the line-scan system

can produce three-dimensional coordinate measurement to an accuracy of a few hundred microns. Nevertheless, it is accepted that a much higher spatial accuracy may be difficult to obtain due to technical limitations in controlling the rotational speed.

The following section presents the principles of operation of the line-scan sensor. A more detailed description of line-scan camera technology can be found in the manufacturers' literature[38,39,40,41].

2.5 The Line-Scan Camera

2.5.1 Principles of Operation

Figure 2-13 depicts a line-scan camera. The camera's sensor consists of a single column of photosensitive elements. At present, the typical number of photosites found in line-scan sensors is 1024 and 2048 [42,43]. However, sensors with as many as 7,926 photosites have been produced[44], mainly for astronomical applications.

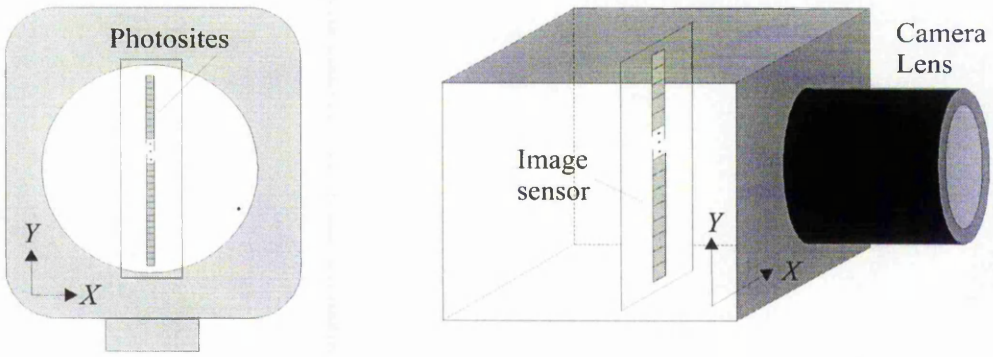

Figure 2-13 *The line-scan camera: front and perspective views.*

Although the sensor technology of line-scan and area array cameras can be identical as, for instance, both are often based on the *charged-coupled device*[45,46,47], the fundamentally different sensor geometry gives rise to a number of different operating and imaging properties.

A line-scan camera can constantly produce successive columns of picture information. The exposure time of each column, during which the photosensitive elements collect incident illumination from a scene, is termed the *integration period.* It is set by the frequency of the *line-transfer* signal, initiating transfer of photosite charge into the camera's shift registers. This is illustrated in the timing diagram of Figure 2-14. At the positive edge of the line transfer pulse the shift registers 'clock out' the video data collected by *n* photosites during the previous integration period. The time available for photon collection by the photosites in each picture column is determined by the integration period. Thus, the longer this period the brighter the resultant image and vice-versa.

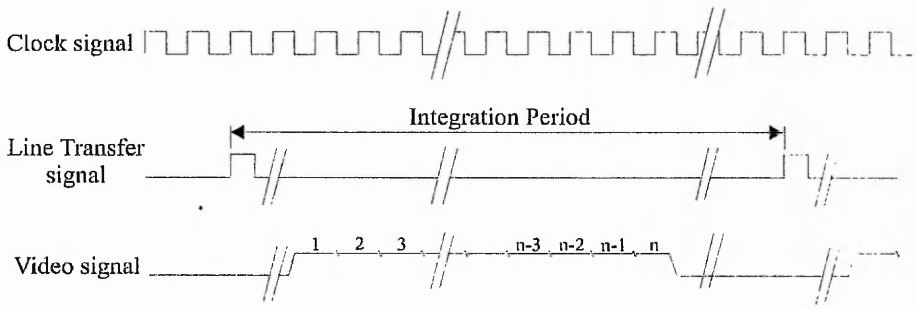

Figure 2-14 *Simplified line-scan timing diagram.*

Image production by a line-scan camera is now discussed. Due to the fundamentally different imaging properties in the X and Y image axes, each axis may be discussed independently.

Y IMAGE AXIS

Figure 2-15 depicts the instantaneous field of view of a line-scan camera in the Y image axis. This axis is also termed the *main axis* and is parallel to the line defined by the sensor.

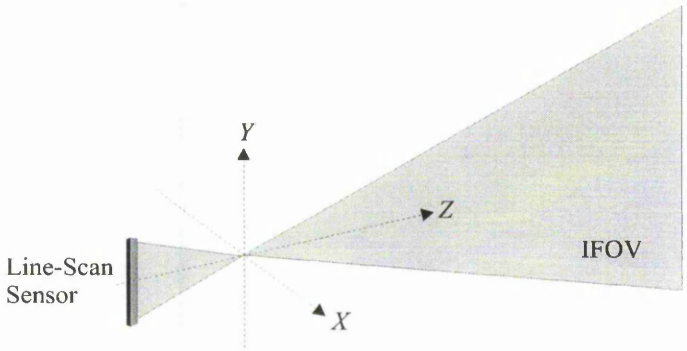

Figure 2-15 *Line-scan camera instantaneous field of view.*

The main axis acts in a similar fashion to a conventional two-dimensional image sensor or a photographic camera. The field of view in this axis depends on:

i. the lens principal distance;

ii. the sensor length;

iii. the sensor to object range.

That is, the Y axis field of view is defined by the perspective projection model.

X IMAGE AXIS

The object to image space projection in the X axis is *orthographic*, that is, invariant of the sensor to object range. In an orthographic projection there is no change in scale. The absence of a second imaging axis in the line-scan sensor necessitates the introduction of a temporal parameter to the imaging process to produce a two-dimensional image. This is achieved by the application of relative motion between the sensor and the imaged object.

To illustrate this process, Figure 2-16 depicts a line-scan sensor being laterally translated past an object. The motion direction is normal to the main axis and parallel to the camera face, that is, it takes place along the X_c camera axis.

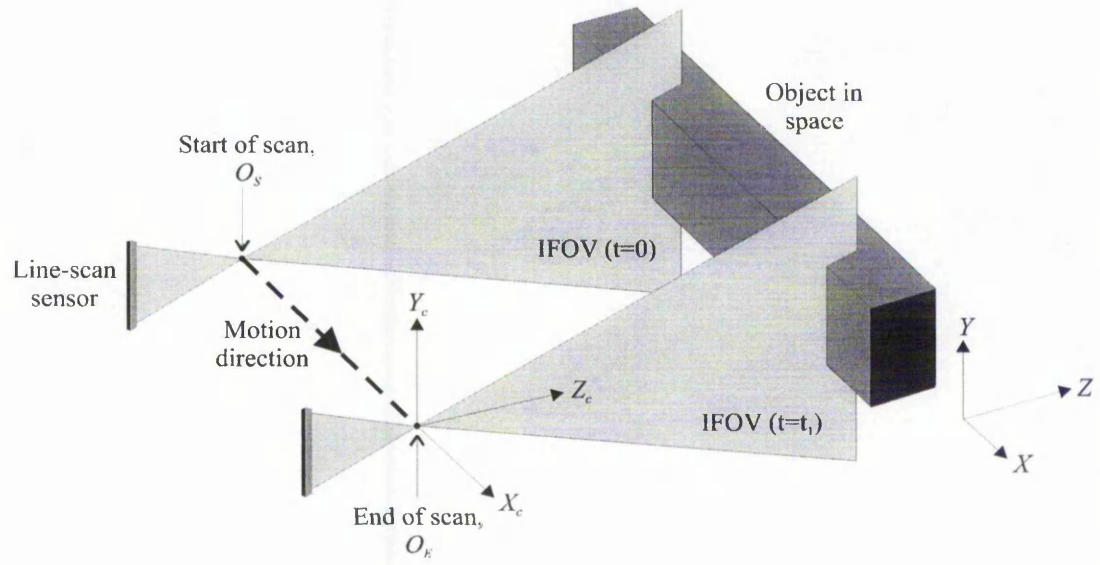

Figure 2-16 *Two-dimensional line-scan image production.*

At the start of the scanning process, t=0, the perspective centre of the lens is at location O_S, while at the end of it, $t=t_1$, the perspective centre will lie at location O_E. Assume that object motion is not continuous, but rather consisting of discrete translation steps, so that the sensor is stationary during the interval the photosites collect photons from light incident on them (the reasoning behind this will be promptly explained). After exposure equal to the integration period, the column of generated picture information is stored in a suitable medium. The object is then advanced by the width of the *scan line*, that is, the picture column at the range of the object. If this process is repeated a number of times and successive columns of picture information are stored alongside each other, a two-dimensional image will be produced.

The X axis field of view of a two-dimensional line-scan image is proportional to:

i. the integration period;

ii. the relative speed of the motion between the camera and the object;

iii. the number of scan lines stored in the image buffer memory.

Since a two-dimensional line-scan image is produced by storing successive columns of picture information, the time required for image acquisition depends on the integration period. Assuming that the relative motion speed between the sensor and the object remains constant and the image consists of a predetermined number of lines, decreasing the integration period results in less time required for the production of a two-dimensional image. This reduces the translation distance during which the sensor produces image information, thus reducing the X axis field of view.

A similar effect is produced when the integration period is kept constant but the speed of the motion is decreased: the reduced sensor translation distance results in reduced X axis field of view.

An increase of the integration period and/or the speed of motion has the opposite effect, increasing the X axis field of view accordingly.

Theoretically, the X axis field of view can be infinitely large, as new scan lines can be continuously acquired at a given rate by the line-scan device. However, in practice, the available frame buffer memory limits the maximum X axis field of view. Thus, the field of view in this axis is also proportional to the number of scan lines stored in the frame buffer memory.

So far, the relative motion between the sensor and the object has been assumed to consist of discrete translation steps, so that the sensor is stationary during each integration period. In practice, the motion is continuous, resulting in image *smearing* that degrades image quality.

2.5.2 Spatial Oversampling and Undersampling

The production of a two-dimensional line-scan image can be regarded as a discrete spatial sampling process as the image sensor consists of discrete photosensitive elements. Each column of picture information is acquired over an interval of one integration period, during which the sensor is displaced by a distance δx. Therefore, the X axis spatial sampling interval, which is equal to the width of the scan line, is a function of the integration period

and the relative speed of motion between the sensor and the object. In addition, it is independent of the photosite dimensions. This is illustrated in Figure 2-17.

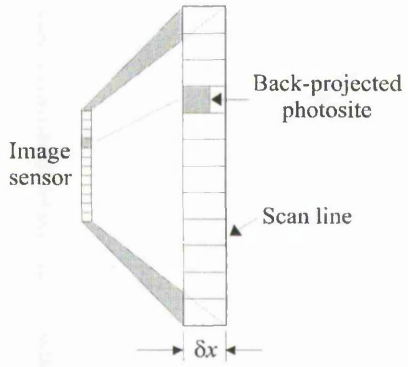

Figure 2-17 *Effective spatial width of a scan line.*

In the context of this discussion, the condition for "correct" image aspect ratio (see next section) can be expressed as: "the value of the product of the integration period and the relative speed of motion between the sensor and the object which yields an X axis spatial sampling interval equal to the width of the back-projected photosite at a given range". Any deviation from this value not only changes the image aspect ratio, but also introduces image smearing due to *spatial undersampling* or *oversampling*.

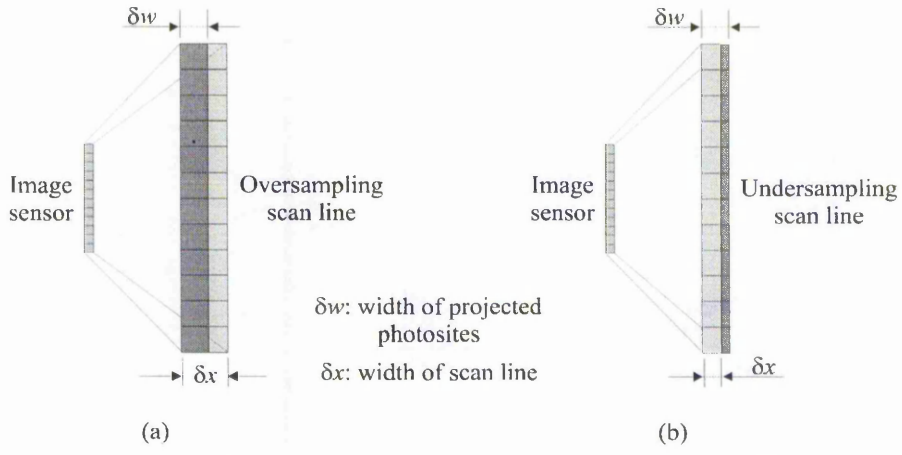

Figure 2-18 *(a) Spatial oversampling, (b) undersampling.*

Spatial oversampling occurs when the X axis sampling interval δx is greater than the width δw of the back-projected photosite at a given sensor to object range. This condition is

illustrated in Figure 2-18(a). The larger the amount of oversampling the worse the spatial resolution (see section 2.5.4), as each scan line occupies a larger spatial area. However, the field of view is increased accordingly and this can be important in some applications.

The opposite effect, spatial undersampling, occurs when the width of the scan line is smaller than the back-projected photosite width, i.e. $\delta x < \delta w$, and is illustrated in Figure 2-18(b). In this case, adjacent spatial samples overlap, giving rise to multiple exposure at the overlapping regions. Although this deteriorates the image quality, spatial resolution is increased in comparison to that offered by an area array sensor having equal photosite dimensions (equal lens principal distance and sensor to object range is also assumed). It should be noted that the increased spatial resolution can be utilised only if feature identification is not impeded by the reduced image quality. Hence, it may be concluded that the Y image axis resolution of a line-scan system is equal to that of a system utilising area array sensors, but the X image axis resolution is potentially higher. In recent work carried out by Kaftandjian[48] on the contrast transfer function of linear detectors, albeit of the X-ray variant, the same conclusion is reached.

The effects of spatial undersampling and oversampling described here are peculiar to line-scan camera operation and should not be confused with the same terminology used in general sampling theory. Indeed, in sampling theory the conditions of undersampling and oversampling are linked to the frequency spectrum of the sampled signal, whereas in a line-scan system they are only related to the physical and system operating parameters which determine the spatial size of the back-projected photosites and the sampling interval in the motion axis.

2.5.3 Image Aspect Ratio

The aspect ratio of an object 'seen' from a given perspective can be defined as the ratio of its width to height. However, the aspect ratio of an image is determined by the ratio of the X to the Y axis spatial sampling interval. Therefore, the aspect ratio of a line-scan image depends on:

i. the lens principal distance;

ii. the size of the photosites in the main axis;

iii. the sensor to object range;

iv. the integration period;

v. the relative speed of motion between the camera and the object.

As the above five parameters are independent, the line-scan camera can produce *affine* images. Consider a line-scan system where the interaction of the above five parameters is such that the X axis spatial sampling interval is smaller than that of the Y axis. Then, a circular object, that is, of 1:1 aspect ratio, occupies a larger portion in the X image axis than in the Y image axis, hence appearing *expanded*. This is shown in Figure 2-19(a).

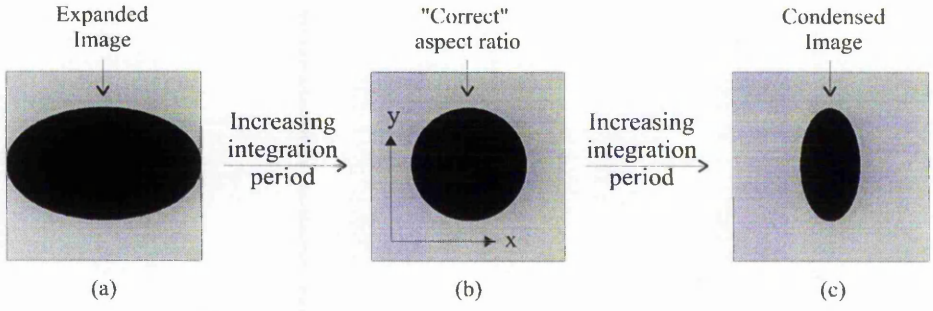

Figure 2-19 *The effect of the integration period on image aspect ratio.*

In Figure 2-19(b) however, the integration period has been increased while the remaining four parameters have been kept constant. This increases the X axis spatial sampling interval and, for a particular combination of the five system parameters determining the X and Y axis spatial sampling intervals, the line-scan image will have the same aspect ratio as the object. This condition of "correct" aspect ratio is illustrated in Figure 2-19(b).

Increasing the integration period further, increases the X axis spatial sampling interval relative to that of the Y axis and the object appears *condensed*, as shown in Figure 2-19(c).

A similar effect is produced by altering the speed of motion whilst keeping the integration period constant: increased speed of motion results in image condensing, as the object is translated over a larger distance during the integration period and vice-versa. Since the integration period is controlled electronically, the aspect ratio of a line-scan image can be readily controlled to suit the requirements of a given application. From this discussion it may also be concluded that, for a constant Y image axis spatial sampling interval, there is a unique value for the product of the integration period and the translation speed for which the image has a "correct" aspect ratio.

The effect of image condensing or expanding can be also achieved along the Y image axis. In Figure 2-20 the camera to object range is progressively increased while the lens principal distance, the integration period and the relative speed are kept constant. This results in constant X axis spatial sampling interval, but that of the Y axis is increasing with range. In Figure 2-20(a) the interaction of the above parameters is such that X axis spatial sampling interval is larger than that of the Y axis, hence the image appears expanded in the Y axis.

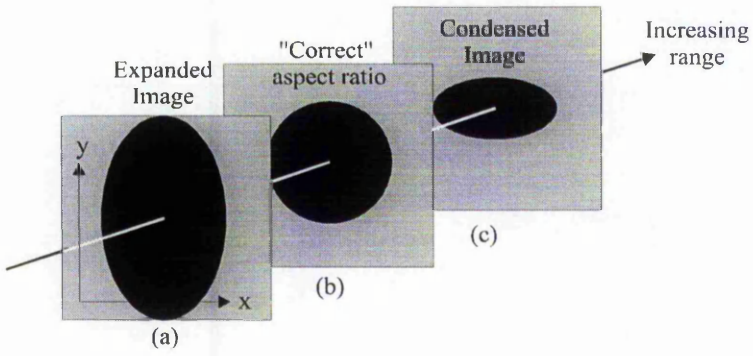

Figure 2-20 *The effect of increasing range on image aspect ratio.*

Progressively increasing range yields image (b), where the aspect ratio of the image is equal to that of the object, and image (c) where the image appears condensed in the Y image axis.

2.5.4 Factors Limiting the Spatial Resolution of a Line-Scan Image

The general definition of the term "resolution" given in the Manual of Photogrammetry[49] is:

"The minimum distance between two adjacent features, or the minimum size of a feature, which can be detected by remote sensing".

This definition is general and implies that both the geometric and the radiometric properties of the imaging system are taken into account. Indeed, the resolution of a machine vision system can be limited by its radiometric properties, i.e. its noise, dynamic range, linearity, number of quantisation levels, etc. However, an investigation into the radiometric properties of a line-scan system is beyond the scope of this research. Rather, the objective of the work undertaken here is to characterise the geometric properties of a line-scan system. Hence, in the context of this work, the spatial resolution of a line-scan system in each imaging axis can be defined by the parameters that determine the size of the spatial sample.

The Y image axis resolution of a line-scan system depends on:

i. the lens principal distance;

ii. the size of the photosites in the main axis;

iii. the sensor to object range.

Assuming that spatial undersampling or oversampling does not affect the identification of a feature, the X image axis resolution depends on:

i. the integration period;

ii. the relative speed of motion.

Note that this definition encompasses only the geometric properties of the line-scan system. Furthermore, it should be noted that a given combination of the above five parameters determines the *maximum achievable* resolution and, unless certain conditions are met, the actual spatial resolution will be less than this maximum value. To illustrate this point, Figure 2-21(a) shows a white planar object containing a black feature of square shape and of size equal to the spatial sample at a given range.

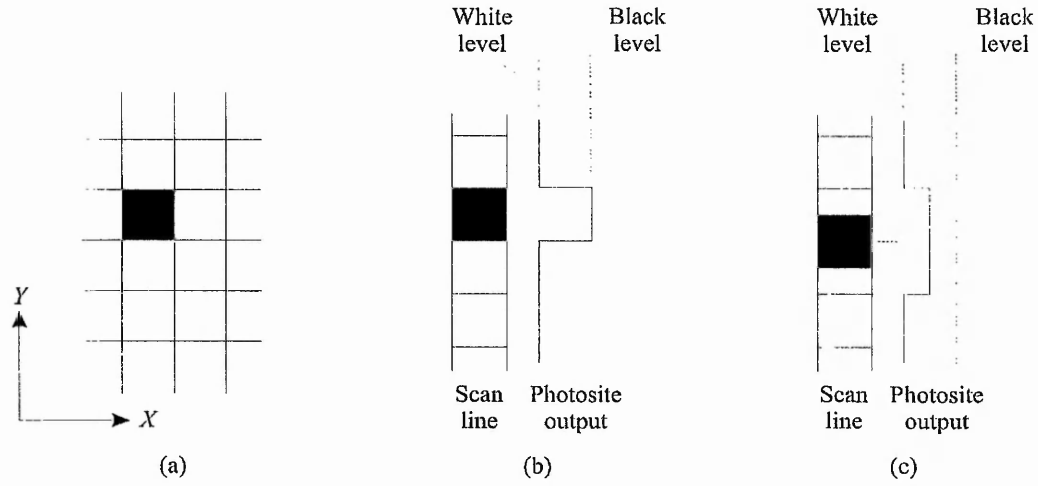

Figure 2-21 *Spatial sampling effects.*

Both the camera and the object are static, and the object is sampled by a particular photosite in a scan line such that the corresponding pixel is exactly filled, as shown in Figure 2-21(b). Then, the electrical output of the sensor is ideally a step function indicating maximum contrast at the boundaries of that pixel. In this case, the frame buffer coordinate of the feature can be determined to an accuracy of one pixel.

In Figure 2-21(c) however, the spatial relationship of the scan line and the feature is such that half the area of two adjacent pixels is covered, giving rise to an electrical output level representing mid-grey (assuming perfect sensor linearity) from the two neighbouring photosites. It can be appreciated that, unless *subpixelation*[50,51,52] techniques are employed, determination of the image location of the target is limited by an uncertainty of ± 1 pixel.

Indeed, only for the condition shown in Figure 2-21(b) will the measurement uncertainty of the target be equal to one pixel. Any relative shift between the feature and the scan line up to the size of the spatial sample produces an electrical output from two adjacent photosites, increasing the uncertainty to ± 1 pixel.

3. THE TWO-DIMENSIONAL SYSTEM

3.1 Introduction

This chapter presents the development and implementation of the two-dimensional rotating object line-scan system. In order to illustrate the imaging characteristics of the rotating object line-scan system detailed in the previous chapter, a number of sample images are presented, following the description of the experimental system.

The rotating object line-scan system has been used in a ballistics application to image cartridge cases. One of the requirements of this application is the measurement of identification marks present on the surface of the cartridge cases, such as scratches and marks. This requires the development of a mathematical model for the two-dimensional system, which is also presented in this chapter.

The chapter concludes with the results of the experimental work undertaken to evaluate the integrity of the experimental system and the accuracy of the mathematical model.

3.2 The Experimental System

The block diagram of the hardware constituting the rotating object line-scan system is depicted in Figure 3-1. This diagram refers to the stereoscopic system, hence the two cameras and two frame grabbers. However, this system also serves as a two-dimensional measurement platform by utilising a single camera. The experimental system consists of the following sub-systems:

i. a rotary stage;

ii. a camera basewidth/convergence angle control table;

iii. two line-scan cameras;

v. two frame grabbers;

vi. an external frequency generator;

vii. a stepper motor controller;

viii. a host computer.

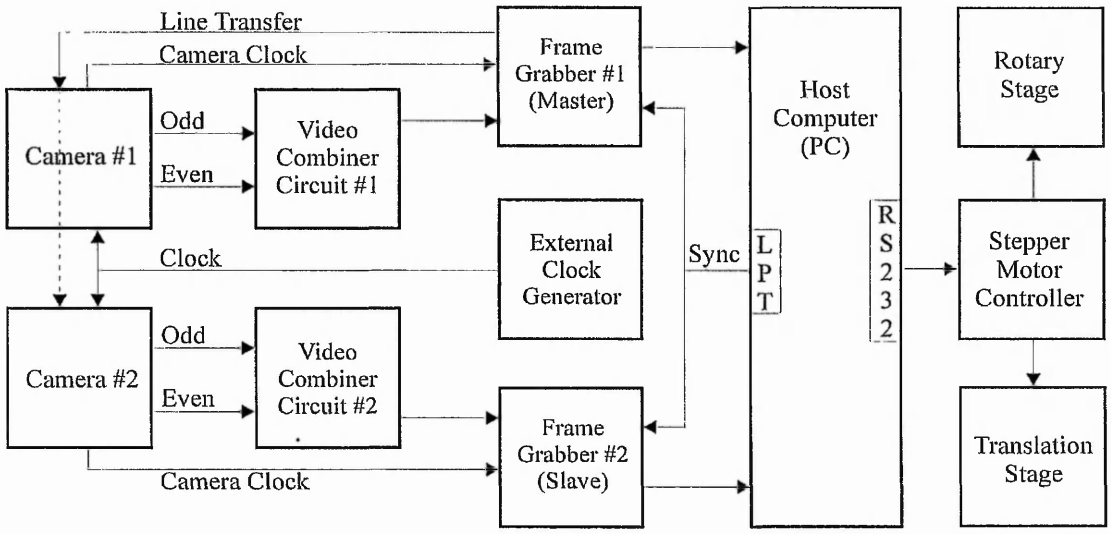

Figure 3-1 *Block diagram of the system.*

Details of each of the above subsystems are given below, whilst the process of image capture is described in section 3-3.

3.2.1 The Line-Scan Cameras

The cameras used are EG&G Reticon's LC1902 [53], consisting of a linear array of 512 square photosites having a width of $13\mu m$ each. The photosite centre-to-centre spacing is also $13\mu m$ and the total sensor length is 6.656mm. The photosites convert incident light into discrete charge packets. After a user-definable integration period, the charge packets are transferred into two shift registers where they are time-division multiplexed. The shift registers carry the data streams from the odd and even numbered pixels, which are then combined externally into a single video data stream.

Camera operation is controlled by two externally generated signals: the *line-transfer* (LT) signal that determines the integration period and the *master clock* (CLK) that determines the video data rate. The timing of these signals is shown in Figure 3-2.

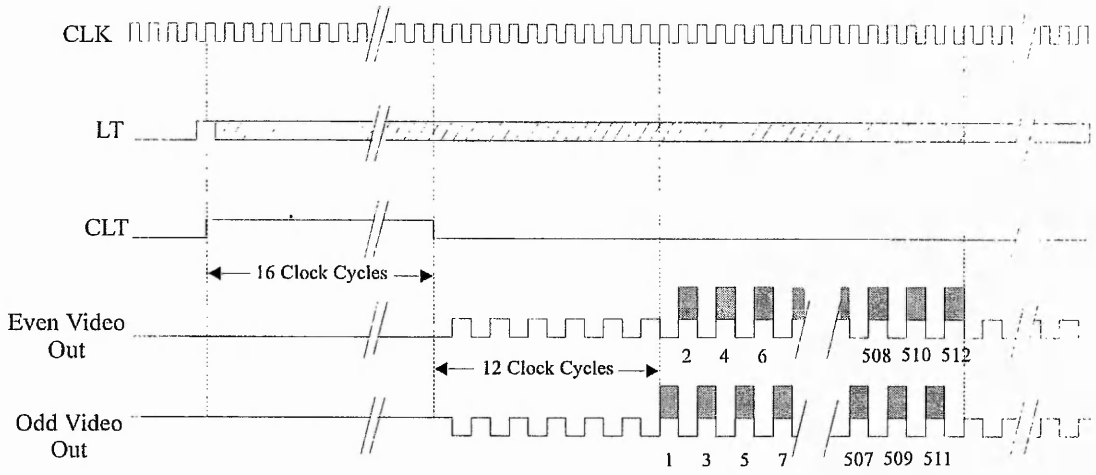

Figure 3-2 *LC1902 timing diagram.*

Half a clock cycle after the line-transfer pulse goes high, the timing circuit of the LC1902 outputs a *camera line-transfer* (CLT) signal. This signal is an internal re-transmission of the line-transfer pulse and is used to synchronise the video combiner circuits and the frame grabbers. During the interval the CLT signal is logic high, charge from each photosite is transferred to the internal shift registers.

Immediately after the falling edge of a CLT logic level transition, the video output lines become active. During the first 12 clock cycles of this phase, the dark diode time occurs where image information is blocked, but the dark video level is present. This allows the effects of the dark current, present in all photodiodes, to be removed thus improving the video dynamic range. In the next phase, active video is available from both video output lines. Video data from each line is output at a rate of half the clock frequency. Following the transfer of one video line to the frame buffer, the video outputs are re-referenced to the dark video level. A new cycle can then begin with a transition of the LT signal from low to high.

3.2.2 The Video Combiner Circuit

This circuit performs the necessary signal conditioning functions to allow successful interfacing of the cameras with the frame grabbers. These include:

i. impedance termination of each video output line;

ii. DC level restoration of each video output line;

iii. gain and offset adjustment of the two video lines;

iv. elimination of dark current effects;

v. recombination of the odd and even video signals.

The schematic diagram depicted in Figure 3-3 is the circuit recommended by EG&G Reticon[54] and was thus chosen to perform the above functions. A printed circuit board utilising a ground plane to preserve the integrity of the high frequency signals was designed by the author, and two such circuits were built.

With reference to the schematic diagram, each video line is terminated in 75 Ohms to -5Volts in order to prevent signal reflections. The odd and even video outputs have a negative, temperature sensitive DC offset, which is removed by C1 and Q2. The latter is an electronic switch operating at half the clock frequency. Q1 references the video signal to a positive value before it is buffered and adjusted for variations in gain by Q3.

Control signals *C* and *D* are active only during the dark diode time and are used to clock Q4 and Q5 in order to remove the effects of dark current from the video signals. The output of this clamp circuit is buffered by Q6 and the video output is sampled by Q7 and Q8 at half the pixel rate. The even video output is conditioned in a similar way, but this time the signal is sampled 180 degrees out of phase. The combined video signal is created on the hold capacitor C7 and at the output of buffer Q9. Operational amplifier U1 applies an offset DC voltage so that the dark output level is at zero. U2 adds gain to the circuit and buffer/amp U3 provides a low impedance output to drive the input of the frame grabber.

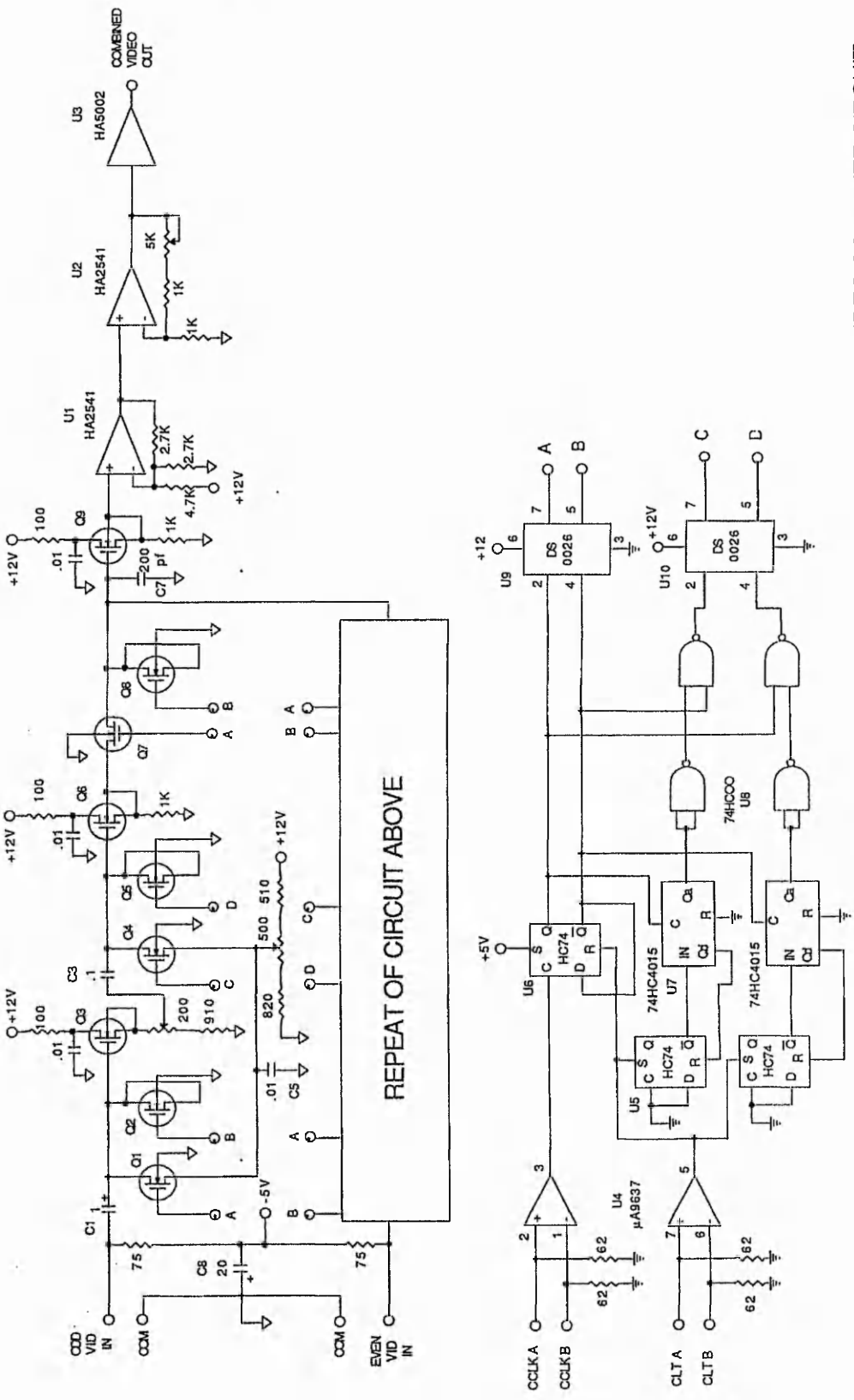

Figure 3-3 *Schematic diagram of the video combiner circuit.*

3.2.3 The Frame Grabbers

The frame grabbers are the P360Fs manufactured by Dipix[55]. These offer the following functions:

i. programmable gain and offset adjustment;

ii. video signal sampling and grey level discretisation;

iii. general purpose programmable I/O control lines;

iv. 4Mb of on-board frame buffer memory;

v. implementation of image processing functions, using an on-board digital signal processor (DSP);

vi. interfacing circuitry for transfer of image data to the host PC.

Figure 3-4 shows the block diagram[56] of the P360F.

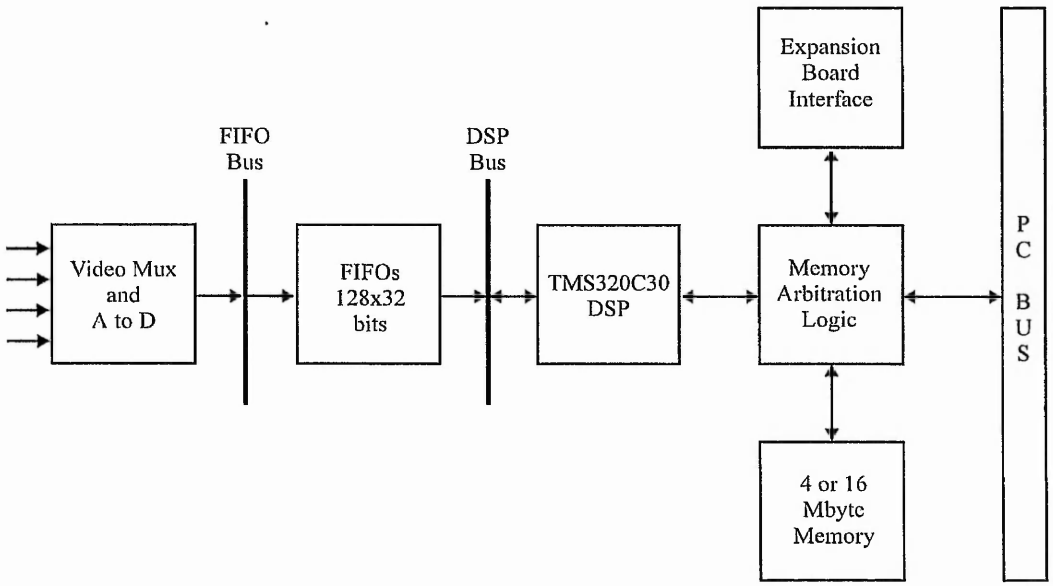

Figure 3-4 *Block diagram of the P360F frame grabbers.*

An input multiplexer allows up to four analogue inputs to be switched into the digitising section. After software-controllable gain and offset adjustment, the video signal is sampled. The sampling process is synchronised with the camera clock and utilises the Bt252 analogue-to-digital converter (ADC). The sampled video signal is discretised into 8 bits per pixel.

The ADC is connected through a 16 bit wide bus to a *first-in-first-out* (FIFO) module. This is used to provide a temporary buffer for the incoming data and is required for two reasons: firstly it allows memory refresh and host PC or DSP access to the memory of the P360F and secondly, it allows the DSP to operate at a different clock frequency from the ADC. Data is packed into 32 bit words by the FIFO control logic, which is capable of storing 64 such words at any one time. The data is then transferred either directly or after processing through the DSP to the on-board memory. The DSP is Texas Instruments' TMS320C30, a 32-bit floating point/integer RISC processor running at 32MHz.

Dipix provides 'C' language callable object modules that perform various image processing tasks within the TMS320C30, as well as a number of utility functions[57], such as the transfer of data to and from the P360F and the host PC. A number of these functions are utilised in the code developed by the author, described in section 3.3.

3.2.4 The Stepper Motor Controller

The stepper motor controller is the MC3E, manufactured by North East Electronics[58]. This can control up to three stepper motors individually and, through the complementing mechanical equipment, provides object rotation, camera basewidth and convergence angle control.

As equation 3-6 of section 3.6.2 indicates, coordinate values in the X image axis are a function of the rotational speed of the object. It is therefore critical that the rotational speed is kept constant during the image capture interval. However, due to the discrete rotor positioning characteristics of stepper motors operating in full-step systems, their low-speed operation can be notchy. In particular, when the pulse-drive frequency is close to the stepper motor's natural frequency, each step will suffer from excessive overshoot and the resulting

motion will be erratic. To overcome this deficiency, *microstepping* is employed. This entails supplying a precisely controlled, variable proportion of the full step current to each phase of the stepper motor. Between two full steps, current flowing through one phase is slowly increased whilst current flowing through the next phase is slowly decreased, ensuring smooth transition between consecutive steps. The MC3E allows the user to set the number of microsteps per full step to 5, 10, 25, 45, 90, 100 or 127. A larger number provides smoother operation at the expense of reduced maximum obtainable speed. For the rotary stage used, a value of ten microsteps per full step was chosen, yielding a positional angular resolution of 14.4 seconds of arc, or 0.004 degrees, and a maximum rotational speed of 14 revolutions per minute. In practice, the actual maximum positional resolution that can be obtained is determined also by the mechanical integrity of the drive system, i.e. the worm gear wear and the pitch of the ballscrew.

In applications where stepper motors are employed to provide accurate positional control, another factor regarding the motor driving scheme must be taken into account: when the rotor is at standstill and a particular speed built-up is required, even if the motor load is below the maximum permissible limits, the load inertia may force the stepper motor to skip some of the first steps. The higher the required speed, the greater the possibility of step skipping. To solve this problem, *ramping* is introduced, whereby the motor drive system limits the permissible maximum acceleration to a suitable value.

The MC3E communicates with the host PC via an RS232 serial port link and uses a custom control script language[59] consisting of strings of hexadecimal characters for each command. The controller has an extensive set of commands build-in, providing flexible motion control capabilities. In order to enable command checking before execution, commands are followed by two data checksum characters. Checksum generation and calls to MC3E commands are accomplished via the host computer. The relevant 'C' language code also caters for the synchronised operation of the stepper motor controller and the frame grabbers.

North East Electronics would not supply the source code for their scripting language and the user's manual was found lacking in accurate information. As a result, the MC3E scripting language was reverse engineered using an RS232 line monitor terminal and the supplied executable code.

3.2.5 The Basewidth/Convergence Stage

A translation stage provides an accurate mechanical platform for camera position control. It is manufactured by McLennan Servo Supplies and consists of two linear axis tables, each supporting a rotary stage (Figure 3-5). The two translation tables are mechanically coupled by a gearbox arrangement such that a 'move' command results in a combined movement of both tables either towards or away from each other, depending on the instructed direction. The positional resolution of this platform is 0.004mm per step.

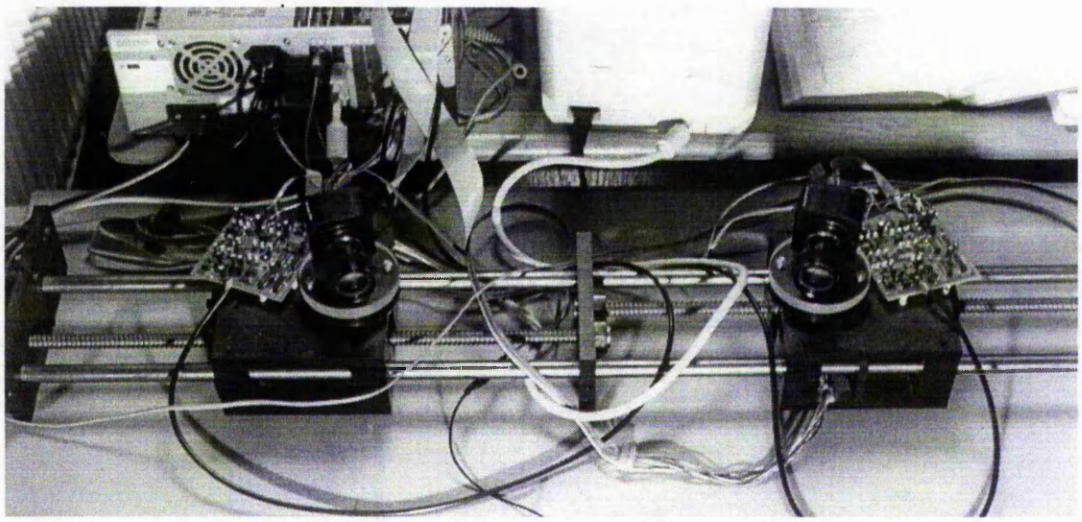

Figure 3-5 *Camera positional controller.*

Each translation table hosts a rotary stage, which is controlled individually by a stepper motor. However, each stepper motor is connected in anti-phase so that the cameras are either diverging or converging dependent on the instructed direction. The angular position resolution is 0.02 degrees per step for each camera.

3.3 Host Controller and System Operation

An IBM PC compatible host controller provides the following functions:

i. control over the operation of all the sub-systems mentioned above;

ii. synchronisation of the image capture process with object rotation;

iii. a platform for image storage and processing.

Software code was written to control the operating parameters of both the frame grabbers and the stepper motor controller. This allows the operator to control the convergence angle, the camera basewidth, the rotational speed, the integration period and the image generation start point. The software flowchart is depicted in Figure 3-6.

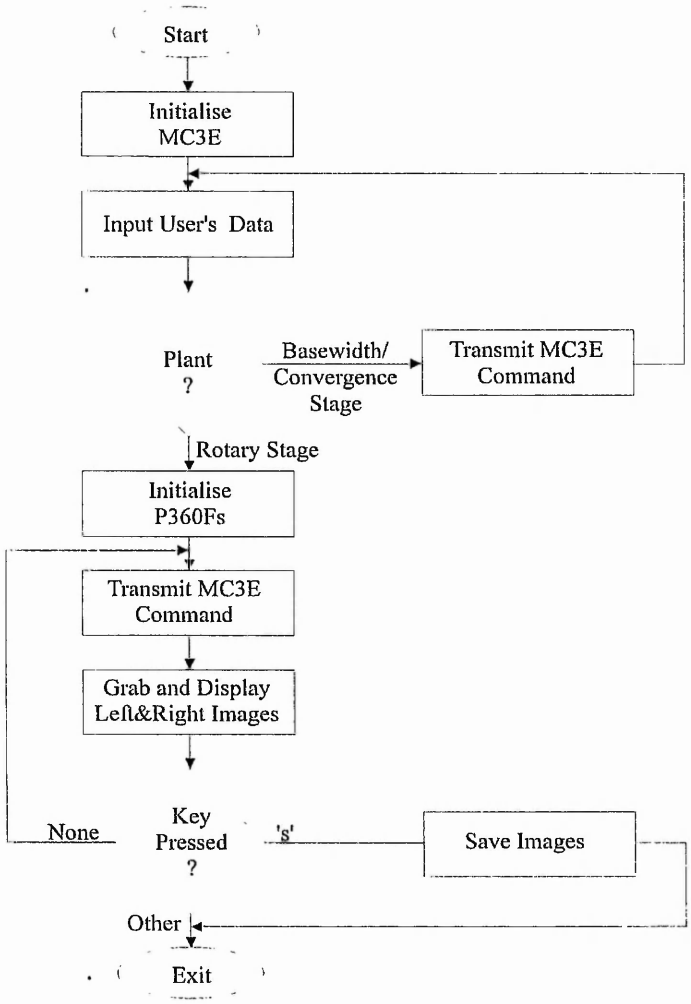

Figure 3-6 *System operation flowchart.*

An image capture cycle begins with the initialisation of the stepper motor controller. This involves setting the PC-to-MC3E serial communication parameters, such as the Baud rate,

parity usage and the datum position of the three mechanical stages. The operator is then requested to input the required process, i.e. object rotation, camera convergence or camera basewidth control. Motion direction and rotational speed can then be entered.

The next phase is the initialisation of the frame grabbers. This entails setting the camera interface parameters and writing the appropriate values in the P360F registers. During the initialisation process, frame grabber memory for image storage is allocated. Following this, image capturing can begin. However, to avoid image distortion from the speed built-up (see section 3.2.4), the image capture is delayed until the rotating stage has acquired a constant speed.

To ensure that the image generation start point is precisely the same for both the left and the right cameras, a means of simultaneously initialising both frame grabbers had to be found. It was decided that a hardware pulse, generated by the parallel port of the PC, be used so that software control over the frame grabbing process is maintained.

The rate of video data is controlled by a Hewlett Packard 8116A frequency generator. In order to ensure consistent timing, this instrument clocks both cameras. In addition, one of the frame grabbers operates in 'master mode', generating the line-transfer pulse for both cameras. However, each video combiner circuit and frame grabber is synchronised by the corresponding camera, by utilising the camera's re-transmission of the clock and line-transfer signals. This ensures correct synchronisation between each camera and corresponding frame grabber regardless of any minor timing differences that may exist between the two cameras.

The sequential image capture process is repeated until the operator directs otherwise, as a number of consecutive image captures are required to yield a satisfactory image. This is because an amount of initial alias charge is present in the CCD arrays and must be removed to allow maximum dynamic range. Also, lens focusing or scene illumination adjustment may be required.

Images stored in the frame grabber memory can be transferred to the host computer memory for displaying or further processing. The images can also be saved on disk in the *tagged image file* (TIFF) image format, which can be read by most image processing packages.

3.4 Sample Images

In order to illustrate the imaging characteristics of the rotating object line-scan system, this section presents sample images of a spray can and a drill chuck. The spray can was chosen because its purely cylindrical and contains text which aids the intuitive interpretation of the imaging properties of the system. The drill chuck has a high degree of cylindrical symmetry and its metallic surface is highly reflective. Images produced by an area array camera are also presented to allow comparison between the two systems.

Figure 3-7 shows the image of the spray can produced by an area array camera. A significant part of the image area is not utilised because of the unequal image and object aspect ratios. The loss of image resolution due to the cylindricity of the object is also apparent.

Figure 3-7 *A cylindrical object as imaged by an area array camera.*

Figure 3-8 shows the image of the spray can as viewed by the line-scan system. The integration period and the rotational speed of the object have been set such that the image has a 1:1 pixel aspect ratio, according to the method described in section 3.6.4. The homogeneously distributed resolution and the planar view of the object are apparent in this

image. In addition, the whole of the image area contains object information. This image depicts an all-round view of the object.

Figure 3-8 *The spray can as imaged by the line-scan system.*

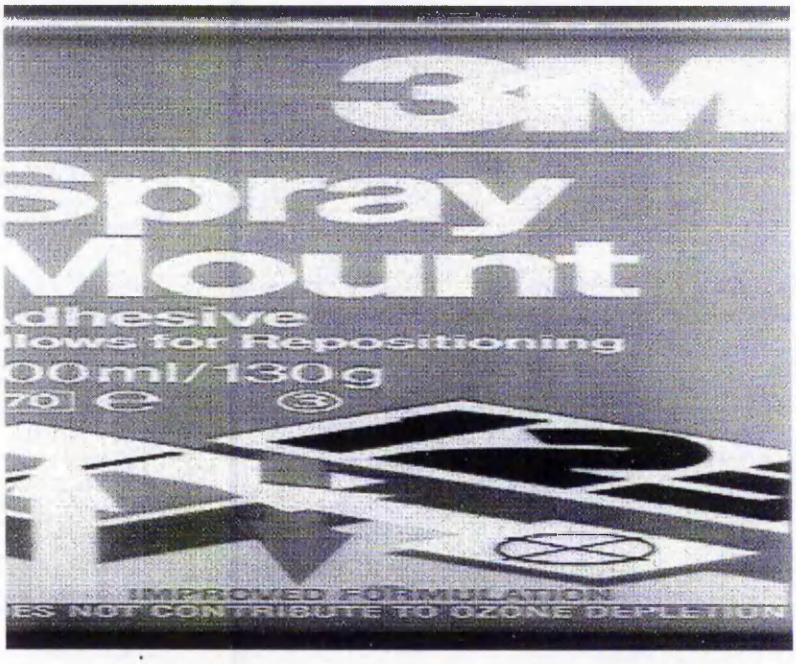

Figure 3-9 *"Electronic zooming" in the X axis.*

In Figure 3-9, the angular field of view has been decreased by decreasing the integration period. This results in increased angular resolution over the imaged portion of the object, as the number of scan lines is the same as that of Figure 3-8. This effect may be thought of as "electronic zooming" since the camera optics have not been altered in any way, yet the spatial resolution has increased. Note that the effects of spatial oversampling (see section 2.5.2) have not produced any noticeable deterioration in the quality of the image.

Area array and line-scan images of the drill chuck are shown in the next three figures. Figure 3-10 depicts the image obtained by an area array camera. Resolution loss due to object cylindricity, poor utilisation of the image area and loss of image information due to excessive scene contrast are apparent in this image.

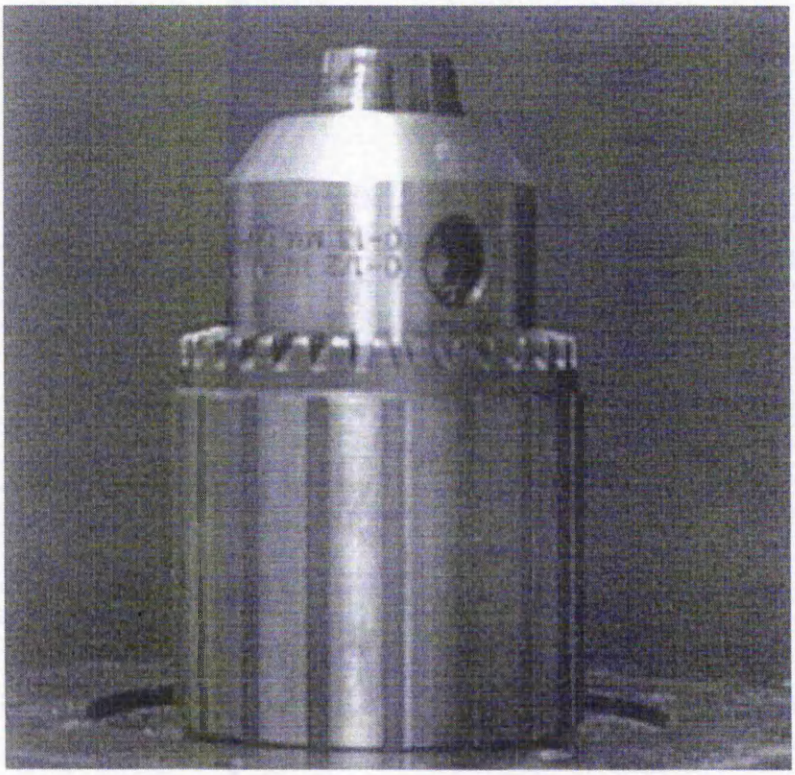

Figure 3-10 *Area array image of a chuck.*

A 360^0 view of this object, obtained by the line-scan system, is shown in Figure 3-11. This planar view can be expected to be significantly more efficient to process. For the image shown in Figure 3-12, the operating parameters of the system have been set such that a 1:1 pixel aspect ratio is obtained (see section 3.6.4).

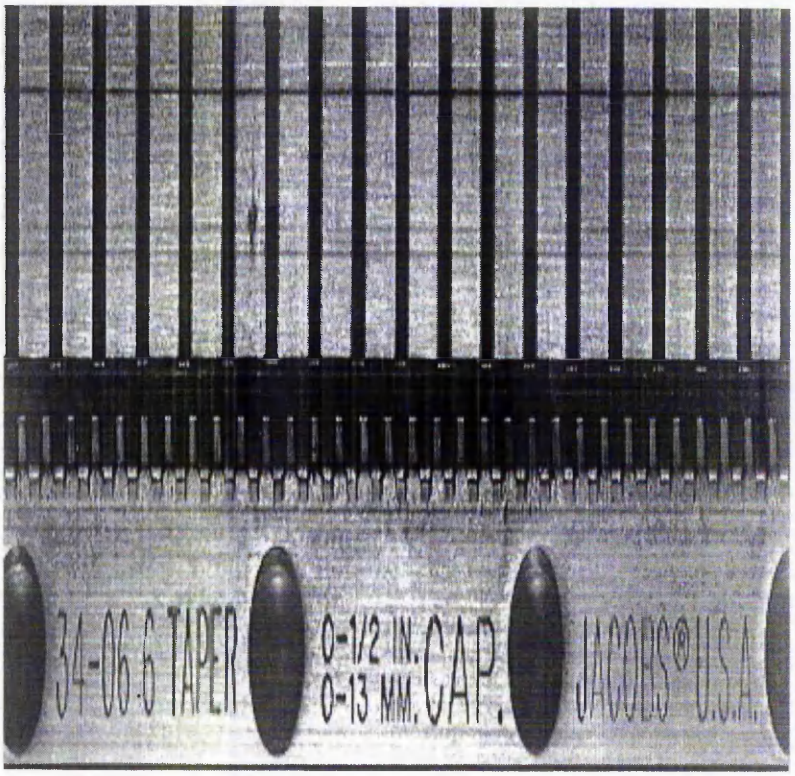

Figure 3-11 *All-round image produced by the line-scan system.*

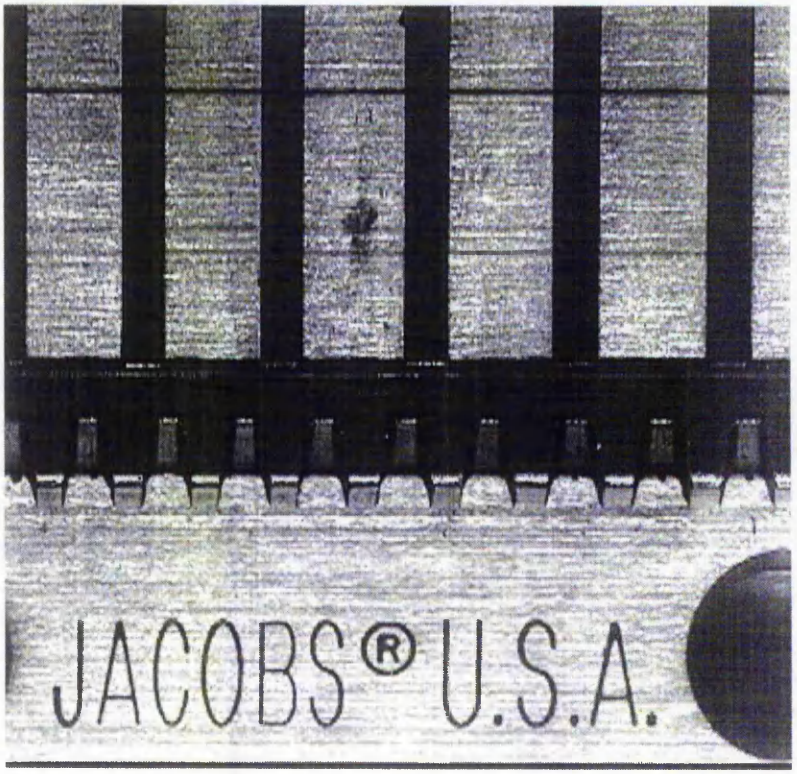

Figure 3-12 *Line-scan image of chuck depicting "correct" pixel aspect ratio.*

3.5 A Ballistics Application

The rotating object line-scan system has been employed in the inspection of fired cartridge cases[60] in a collaborative programme with the 'Edith Cowan University' at Perth, Australia.

Up to date, the analysis of marks, scratches and other identification features on fired cartridge cases are being carried out with the aid of comparison microscopes, where examination of markings on different cartridge cases is performed. This is a significantly labour-intensive and time-consuming process, and can prohibit the routine checking of catalogued exhibits against those obtained from weapons that come into the possession of the police. Moreover, it is difficult to routinely circulate such exhibits around a country. Consequently, the need for an improved cartridge case inspection system is double-fold; first in terms of greater inspection efficiency combined with reduced process times, and second in terms of image storage and transmission technology, where a digital technique is required.

Since cartridge cases are nominally cylindrical, the limitations associated with area-array sensors when used to image cylindrical objects, detailed in section 2.2, are present. The effect of these limitations can be seen in Figure 3-13, which shows the image of a cartridge case obtained by an area array camera.

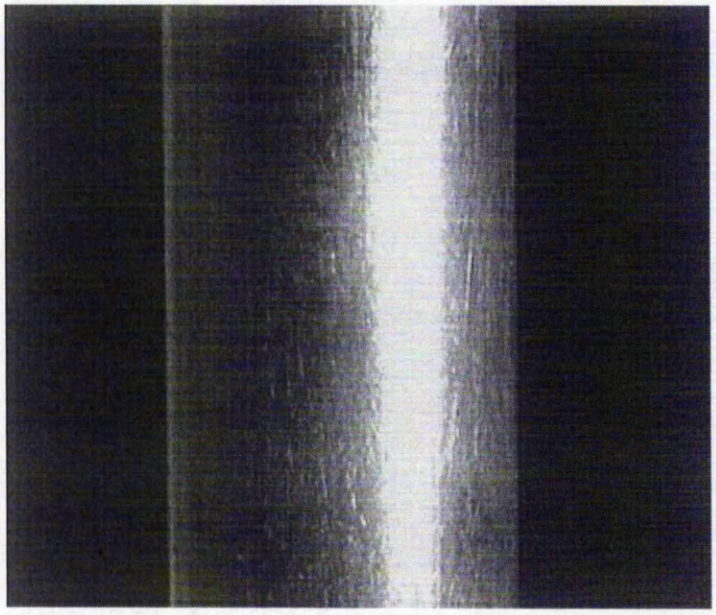

Figure 3-13 *Area array image of a cartridge case.*

During preliminary experiments with the rotating object line-scan system, it was found that the angle of the light source to the cartridge case has a major effect on image quality. Consequently, an array of high-power, narrow-angle LEDs (Light-Emitting Diodes) was used to illuminate the cartridge cases. Because the sensitivity of the line-scan cameras used peaks in the near-infrared part of the spectrum, infrared LEDs were chosen. The experiments indicated that there exists an optimal angle of approximately collimated, i.e. narrow angle, light incident on the surface of the cartridge cases.

A sample line-scan image of the cartridge case used to produce Figure 3-13 is depicted in Figure 3-14.

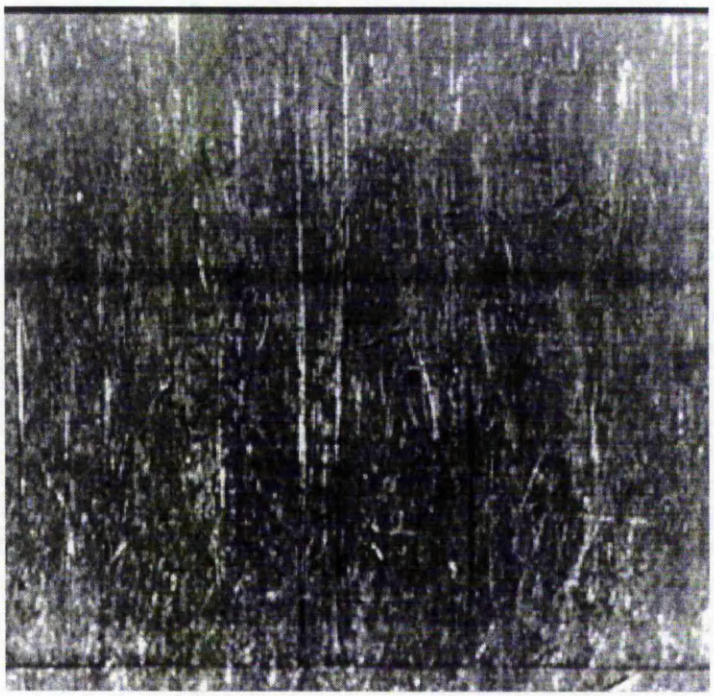

Figure 3-14 *The image obtained by the line-scan system.*

From the detail in the above figure, it can be seen that the rotating object line-scan system produces efficient images that convey much higher surface detail than the area array camera system. This application has set the foundations for a new research programme to be carried out at The Nottingham Trent University. A dedicated hardware system along with the necessary image processing support is currently under development.

To identify whether two different cartridges have been fired by the same weapon, twelve matching identification marks are required. This matching process requires the ability to measure the location of marks on the cartridge cases. The following section presents the development of a mathematical model for the two-dimensional system that can be used to perform such measurements.

3.6 Algorithms for the Two-Dimensional System

The mathematical algorithms describing the two-dimensional system relate the Cartesian coordinates (X, Y, Z) of a point in object space with its corresponding frame buffer coordinates (x_f, y_f). This transformation expresses the projection of three-dimensional space to the two-dimensional computer image as performed by the system. Since this transformation is singular, depth is assumed to be known both for the purposes of analysis and the relevant experiments carried out with the two-dimensional system.

3.6.1 Geometry of the Two-Dimensional System

Figure 3-15 depicts the geometry of the two-dimensional rotating object line-scan system.

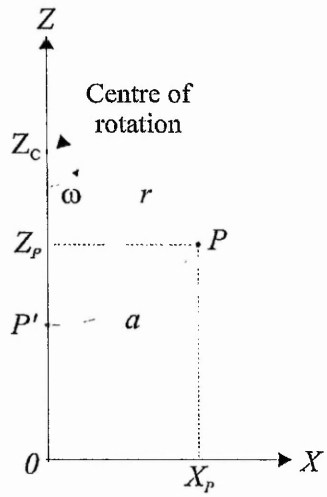

Figure 3-15 *Geometry of the two-dimensional system.*

The reference coordinate system is camera-centred, with its origin 0 located at the perspective centre of the lens. The Z axis is coincident with the optical axis, and the Y axis is parallel to the sensor. The X axis is normal to both the Y and the Z axes, and the XYZ coordinate system is orthogonal. The centre of rotation has coordinates $(0,\ Z_c)$.

A point P at distance r from the rotation axis has coordinates $(X_P,\ Y_P,\ Z_P)$. This point is subjected to clockwise rotation and it is imaged by the line-scan camera at spatial location P' (see Figure 3-15). The Cartesian coordinate components X_P and Z_P of P are expressed as a function of the radius r and the angle ω by -

$$X_P = r \sin\omega \tag{3-1}$$

$$Z_P = Z_C - r\cos\omega \tag{3-2}$$

where angle ω is the counterclockwise angle defined from the Z axis to point P and is always positive.

3.6.2 X Axis Algorithm Derivation

Let the arc defined by points P and P' be denoted by $\vec{a}$. From Figure 3-15, arc $\vec{a}$ can be expressed as a function of angle ω by -

$$\vec{a} = r\omega \tag{3-3}$$

where -

$$0 \le \omega < 2\pi$$

If point P is rotating with rotational speed U_r, expressed in revolutions per minute (rpm), its angular speed U_A is given by -

$$U_A = 2\pi r \frac{U_r}{60} \tag{3-4}$$

Let x_f be the number of scan lines generated during image capture over distance $\ddot{a}$, and t_{IP} be the integration period. Then, capturing of distance equal to arc $\ddot{a}$ requires $t_{IP}x_f$ time, hence -

$$\ddot{a} = U_A t_{IP} x_f \tag{3-5}$$

By combining equations 3-3, 3-4 and 3-5, x_f can be expressed as -

$$x_f = \frac{30}{\pi U_r t_{IP}} \omega \tag{3-6}$$

In equation 3-6, U_r is the rotational speed of point P expressed in rpm, t_{IP} is the integration period in seconds and x_f is the X axis frame buffer coordinate of point P, referenced to the start of scan. From equation 3-6 it can be seen that the projection in the X image axis is linear in angle and independent of the lens perspective centre to the rotation axis range. The product of the integration period and the rotational speed determine the X axis scaling factor in the transformation from the object space to the computer frame buffer. This scaling factor, denoted by s_x, is equal to -

$$s_x = \frac{30}{\pi U_r t_{IP}} \tag{3-7}$$

and equation 3-6 is simplified as -

$$x_f = s_x \omega \tag{3-8}$$

The X and Z axis object space coordinates of point P can be related to the X axis frame buffer coordinate from equations 3-1, 3-2 and 3-8 as -

$$X_P = r \sin(s_x^{-1} x_f) \tag{3-9}$$

$$Z_P = Z_C - r \cos(s_x^{-1} x_f) \tag{3-10}$$

For a given set of values at the right-hand-side of equations 3-9 and 3-10, either a negative or a positive solution for X_p and Z_p can be obtained. For example, points A_1 and A_2 of Figure 3-16 have the same Z axis coordinate, but their X axis coordinates have opposite signs.

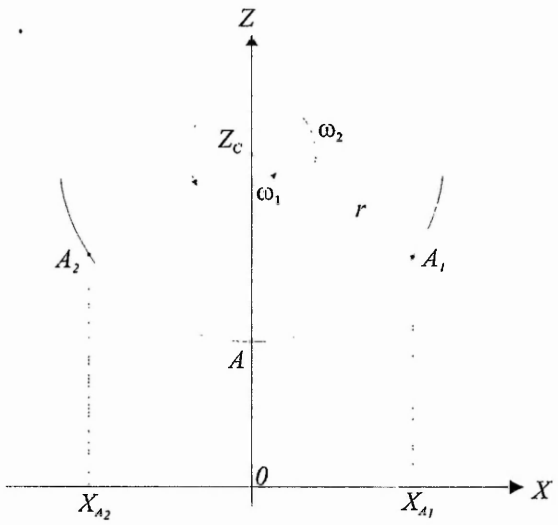

Figure 3-16 *Points symmetrically located about the Z axis.*

The correct solution can be determined only with prior knowledge of the quadrant that point P occupied at the start of scan. This issue is addressed in the fourth chapter, where the mathematical model of the stereoscopic system is developed.

3.6.3 Y Axis Algorithm Derivation

From the perspective projection model, the Y axis object space coordinate of point P can be expressed as -

$$Y_p = \frac{Z_{p'}}{f} y_p \tag{3-11}$$

where f is the lens principal distance, y_p is the image space coordinate of the point in the camera-centred coordinate system and $Z_{p'}$ is the distance from the perspective centre to

point *P'* in Figure 3-15, that is, the range from the perspective centre to point *P* at the instant of image capture. This range is -

$$Z_{P'} = Z_c - r \tag{3-12}$$

where Z_c is the distance from the rotation axis to the perspective centre, and r is the radial displacement of point *P* from the rotation axis.

In equation 3-11, y_p is measured in the camera-centred coordinate system. This coordinate is transformed by the system into y_f, the Y axis frame buffer coordinate, which is referenced to the top-left corner of the stored image -

$$y_p = s_y(y_f - C_y) \tag{3-13}$$

where y_f is the Y axis frame buffer coordinate of point *P*, and s_y is a scale factor introduced to adjust the image space scale to that of the frame buffer system and transform from metric units to frame buffer coordinates. Since image space values can be both positive and negative, a positive offset must be applied to produce positive only frame buffer coordinates. This offset is introduced by the term C_y, which is the Y axis coordinate of the image centre expressed in the frame buffer coordinate system.

Substituting equations 3-12 and 3-13 into 3-11 produces -

$$Y_p = \frac{s_y(Z_c - r)}{f}(y_f - C_y) \tag{3-14}$$

The scale factor s_y may be determined as follows. Let n be the total number of photosites on the sensor and l be the physical length of the sensor. The scaling factor can then be expressed as -

$$s_y = \frac{l}{n} \tag{3-15}$$

If the distance between adjacent photosites of the sensor is zero, as is the case for the cameras used in the experimental system, the sensor length l is -

$$l = n \cdot \delta y \tag{3-16}$$

where δy is the photosite width. Equations 3-15 and 3-16 produce -

$$s_y = \delta y$$

Therefore, for the cameras used in this work, the Y axis scale factor is equal to the photosite width.

3.6.4 Determining the Conditions for Correct Image Aspect Ratio

The X and Y axis algorithms derived in the preceding section allow the determination of the system operating parameters that yield a correct, i.e. 1:1, image aspect ratio. This requires that an object having a 1:1 aspect ratio produces an image of equal aspect ratio. Since the object is nominally cylindrical, its aspect ratio can be defined as the ratio of its circumference to its height. Let these be denoted by $\ddot{C}$ and H, respectively. Combining equations 3-3 and 3-8 produces -

$$\ddot{C} = \frac{r}{s_x} x_f \tag{3-17}$$

From equation 3-14 -

$$H = \frac{s_y (Z_c - r)}{f} (y_f - C_y) \tag{3-18}$$

Combining equations 3-17 and 3-18 under the condition that $\ddot{C} = H$ produces -

$$r = Z_c \left(\frac{f}{s_x s_y} \cdot \frac{x_f}{y_f - C_y} + 1 \right)^{-1} \tag{3-19}$$

But the image also has a 1:1 aspect ratio, so that $x_f = y_f - C_y$, and equation 3-19 becomes -

$$r_{1:1} = Z_c\left(\frac{f}{s_x s_y}+1\right)^{-1} \tag{3-20}$$

where $r_{1:1}$ denotes the radius of the object for a correct image aspect ratio, f is the lens principal distance, s_x and s_y are the X and Y scaling factors, respectively, and Z_c is the distance from the perspective centre of the lens to the rotation axis.

If the dimensions of the object are known, equation 3-20 can be solved for the product of the integration period and the rotational speed -

$$U_r t_{IP} = \frac{30 s_y}{\pi f}\cdot\frac{Z_C - r}{r} \tag{3-21}$$

Because the image aspect ratio is determined by the spatial sampling properties of the system, the aspect ratio of the raw image, as opposed to that obtained from an object of a given aspect ratio, can be considered. Indeed, it is apparent that the above discussion pertains to any part of an object that has an arc length to height ratio equal to unity. Thus, to explicitly express that the image aspect ratio is independent of the object aspect ratio, the term *pixel aspect ratio* will be adopted.

3.7 Two-Dimensional System Experiments

The purpose of the following experiments is two-fold: first, to evaluate the integrity of the experimental system and second, to establish that the experimental system can be successfully modelled using the algorithms developed in the previous section. The first task requires that the precision of certain system parameters, such as the rotational speed and the integration period, is adequately quantified. Here, the term "precision" refers to the consistency of these parameters over the course of individual experiments as well as to their

long-term repeatability, defined over a number of experiments undertaken at different instants.

The coordinate measurement ability of any metrological system can be assessed by its accuracy, i.e. the difference between the true and the measured value of a parameter. To alleviate the effects of random errors, the mean of the measured values over a sufficient number of observations is taken[61]. Note that the absolute coordinate measurement accuracy of the two-dimensional system is of limited consequence as the system cannot derive depth autonomously anyway, and it is not calibrated. The latter indicates that the precise location and orientation of the camera system with respect to the object space coordinate system, as well as the interior orientation and temporal parameters cannot be determined accurately. Thus, it is anticipated that relatively large systematic errors will be present in the relevant experimental results. Regardless of absolute accuracy, however, results obtained from the accuracy experiments will indicate whether the experimental system obeys the derived analytical model or not. Indeed, the presence of systematic errors can only compromise the degree of confidence pertaining to the validity of the mathematical model, but not to such a degree as to render the results inconclusive.

3.7.1 Experimental Strategy and Error Representation

Due to the different underlying nature of the precision and accuracy experiments, their respective outcome has to be specified in different forms. Specifically, repeatability experiments are based on multiple observations of the value of the same physical parameter. Hence, the results of these experiments are specified in terms of a *mean value* and *standard deviation*. Experiments to characterise the validity of the system model, on the other hand, require that the repeatability of the system parameters has been previously established to a given degree of confidence. Consequently, the outcome of these experiments is specified in terms of *residual error* and *standard deviation*. The statistical significance of the mean value, residual error and standard deviation is summarised below, from Mikhail[62].

Let l_n be the individual observations in a set of measurements, n being the sample size. The sample mean $\overline{L}$ is the most probable value and is equal to -

$$\overline{L} = \frac{1}{n} \sum_{i=1}^{n} l_i \tag{3-22}$$

The deviation of each sample from the mean value is called the residual error V_i and is given by -

$$V_i = l_i - \overline{L} \tag{3-23}$$

The standard deviation m is defined as -

$$m = \pm \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} V_i^2} \tag{3-24}$$

This parameter is also referred to as the *rms* (root-mean-square) error.

3.7.2 Establishing the Repeatability of System Parameters

The integration period is determined by a periodic electronic signal. The rotational speed can also be observed as an electronic signal using a suitable transducer. However, the electronic noise present in these signals as well as any spurious pickup from the environment can render their observation in the time domain, for instance by means of a storage oscilloscope, inconclusive. The preferred method is to observe their spectra (in the frequency domain) on a spectrum analyser. High frequency noise then manifests itself as spurious modulation of the signal envelope, whereas low frequency permutations shift the whole envelope of the signal, indicating poor consistency in the measured parameter.

EVALUATION OF THE ROTATIONAL SPEED PRECISION:

In order to evaluate the consistency of the rotational speed independently of other system parameters, an optical encoder is coupled to the stepper motor shaft. This is manufactured by Hewlett Packard[63], type HEDS-5600A, and features a resolution of 500 pulses per revolution. The transducer incorporates the required signal-conditioning and output drive electronics making interfacing with measurement instruments straightforward. The use of a storage oscilloscope was found to be inadequate, as, besides the aforementioned deficiencies, reliable synchronisation could not be achieved. Subsequently, it emerged that the most reliable observation method comprised the concurrent use of a frequency counter and a spectrum analyser. The former requires a time-base of one second to yield a precision better than 1 part in 10^4. Therefore, the measurement is a good indicator of consistency. The spectrum analyser, on the other hand, has a minimum scan width of 20Hz, which may not be sufficient to quantify consistency, but provides a good visual indication of high frequency spurious components.

To ensure that the measurements are reliable, three different frequency counters were utilised, namely a Thandar TF200, a Farnell FM600 and a Beckman Industrial DM27XL multimeter. The highest resolution at a gate time of one second is offered by the Farnell counter and the measurements reported in Table 3-1 were obtained by this instrument. Visual inspection of the signal envelope is carried out on a Hewlett Packard HP8556A spectrum analyser.

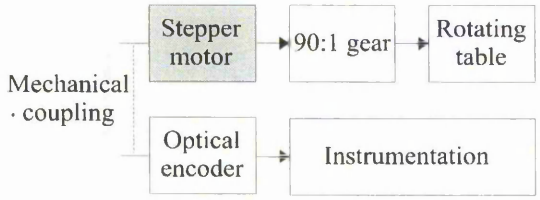

Figure 3-17 *Block diagram of the rotational speed precision measurement system.*

The optical encoder produces 500 pulses per motor shaft revolution, thus the transducer's output frequency is equal to 500/60 Hz for a rotational speed of 1rpm. However, the rotary stage incorporates a gearing of 90:1 (Figure 3-17) so that one revolution of the rotary stage

at 1rpm yields 45000/60 Hz. Therefore, if the stage rotates with a speed of U_r, the encoder's output signal has a frequency f_{en} of -

$$f_{en} = \frac{4500}{6} U_r \tag{3-25}$$

Table 3-1 presents results of the rotational speed precision experiments. The first column shows the *nominal* rotational speed in rpm and the remaining columns show the results of ten experiments with a sampling interval of one second.

TABLE 3-1 *Output of optical encoder in ten experiments.*

U_{nom} (rpm)	f_{en}#1 (Hz)	f_{en}#2 (Hz)	f_{en}#3 (Hz)	f_{en}#4 (Hz)	f_{en}#5 (Hz)	f_{en}#6 (Hz)	f_{en}#7 (Hz)	f_{en}#8 (Hz)	f_{en}#9 (Hz)	f_{en}#10 (Hz)
0.5	-	-	-	-	-	-	-	-	-	-
1.0	753.14	753.18	752.94	753.11	753.47	753.51	753.40	753.28	752.96	753.36
1.5	1130.02	1129.90	1130.11	1129.68	1129.73	1130.05	1129.72	1129.68	1129.59	1130.04
2.0	1512.62	1512.50	1512.39	1512.44	1512.40	1512.34	1512.57	1512.57	1512.38	1512.43
2.5	1882.96	1882.80	1883.07	1883.27	1882.90	1883.04	1883.38	1883.44	1883.26	1882.92
3.0	2259.51	2259.47	2259.61	2259.88	2259.49	2259.67	2259.40	2259.49	2259.36	2259.55
3.5	2631.30	2631.51	2631.44	2631.24	2631.00	2631.18	2630.99	2631.33	2631.35	2631.41
4.0	3001.19	3001.16	3001.26	3001.43	3001.20	3001.15	3001.21	3001.18	3001.25	3001.23
4.5	3369.70	3369.80	3370.01	3369.67	3369.51	3369.42	3369.49	3369.70	3369.93	3369.71
5.0	3766.21	3766.14	3766.09	3766.00	3766.38	3766.36	3766.48	3766.38	3766.17	3766.30
5.5	4175.49	4175.40	4175.62	4175.44	4175.66	4175.75	4175.78	4175.57	4175.35	4175.52
6.0	4465.95	4465.91	4465.85	4465.86	4465.92	4466.15	4465.96	4465.89	4465.91	4466.07

Readings for a nominal speed of 0.5rpm could not be obtained due to limitations in the optical encoder. Table 3-2 presents the mean value $\overline{U}$ of the rotational speed obtained from the output of the encoder, the residual error V and the sample standard deviation m obtained by applying equations 3-22 to 3-25. The last column indicates the precision of the rotational speed in parts per 10^4.

TABLE 3-2 *Results of the rotational speed precision experiments.*

U_{nom} (rpm)	$\overline{U}$ (rpm)	V (rpm)	m (rpm)	p.p.10^4
1.0	1.00431	0.00431	±0.00026	±2.60
1.5	1.50647	0.00647	±0.00024	±1.60
2.0	2.01662	0.01662	±0.00012	±0.60
2.5	2.51081	0.01081	±0.00028	±1.12
3.0	3.01272	0.01272	±0.00019	±0.63
3.5	3.50837	0.00837	±0.00022	±0.63
4.0	4.00163	0.08017	±0.00010	±0.25
4.5	4.49293	-0.00710	±0.00024	±0.53
5.0	5.02167	0.02167	±0.00019	±0.38
5.5	5.56741	0.06741	±0.00018	±0.33
6.0	5.95460	-0.04540	±0.00012	±0.20

From the fourth column of this table it can be seen that there is no apparent deterioration of the standard deviation at increasing speeds. Note that a standard deviation of ±0.00026 in 6rpm corresponds to a precision of ±0.43 parts in 10^4 whereas the same value in 1rpm yields ±2.60 parts in 10^4. This would indicate that the measurements are bound by random noise, frequency modulating the encoder's output signal. This noise is attributed, in part, to subsequent modules of the measurement chain, further contaminating the encoder's output signal.

A maximum standard deviation of ±0.00028 rpm was measured at a rotational speed of 2.5rpm. If this worst case value, bound by random noise rather than actual speed inconsistency, is referred to the maximum rotational speed of 6rpm, a speed precision of ±0.47 parts in 10^4 is obtained. This value represents an inferred worst case speed precision and is therefore regarded as the worst case precision of the system at all speeds.

Since the two-dimensional images produced by the line-scan system consist of 508 scan lines, for an image having a 360^0 angular field of view the angular resolution of the system is -

$$\frac{360^0}{508} \approx 0.709^0 \ / \ pixel$$

The worst case speed precision of ±0.47 parts in 10^4 can be expressed as an angular precision of -

$$\pm 0.47 \frac{360^0}{10^4} \approx \pm 0.017^0 \text{ in } 360^0$$

which is less than an order of magnitude the angular resolution of the system. Consequently, for the purposes of this work, the rotational speed is regarded to be precise enough.

EVALUATION OF THE INTEGRATION PERIOD PRECISION:

The arguments outlined above rendering the evaluation of precision more accurate in the frequency domain than in the time domain hold true also for the integration period experiments. However, experiments indicated that the very small duty-cycle of the line-transfer pulse (active-high for 10nS in each integration period) is not sufficient to trigger any of the three frequency counters used. Evaluation in the time domain by means of a storage oscilloscope proved also inconclusive because of limited resolution, synchronisation problems and the presence of noise. However, the line-transfer signal is produced by a digital input/output port of the frame grabbers and is therefore expected to be highly precise. This is verified experimentally in section 3.7.3.1.

EVALUATION OF THE REPEATABILITY OF START OF SCAN:

In certain cases, absolute distances of spatial points referenced to the same coordinate system may be required over more than one pass of an object. For example, an experiment presented in Chapter 5 requires one pass for system calibration and another for measurement. Thus, the start of scan must be repeatable to within one pixel. To assess this, the frame buffer coordinates of an arbitrary spatial point were measured ten times. Between each run the rotary stage was returned to its datum position and the values of all other system operating parameters were retained. Identical frame buffer coordinates to within one pixel were recorded in all ten runs.

3.7.3 Validation of the Mathematical Model

3.7.3.1 X Axis Algorithm Verification

In the following set of experiments, each one of the three unknowns that determine the X axis frame buffer coordinates x_f, namely the rotational speed U_r, the integration period t_{IP} and the angle ω, are varied individually within the physical limits set by the experimental system. During each experiment, it is assumed that all variables apart from that which is stepped have *constant* and *known* values. The validity of the first assumption is established in the previous section, whereas that of the second can only affect the accuracy of the results. Thus, in accordance with the arguments presented at the beginning of section 3.7, the above assumption is justifiable for the purposes of these experiments. Tables 3-3, 3-4 and 3-5 in conjunction with the respective graphs present the results. The notation used in presenting the results is as follows:

- column $L\{x_f\}$ shows the calculated frame buffer coordinates in the X axis;
- column $l\{x_f\}$ shows the corresponding measured values;
- column $V\{x_f\}$ shows the resultant error residual;
- column $l\{\omega\}$ presents spatial radial values extrapolated from corresponding frame buffer coordinate measurements;
- column $V\{\omega\}$ presents the associated object space radial residual error.

Where necessary, calculated values have been rounded off to a precision of three or more decimal points, dependent on the required resolution.

VARIATION OF THE INTEGRATION PERIOD:

Experimental conditions: $U_r = 2.5rpm$, $\omega = 14^0$.

TABLE 3-3 *Accuracy in variations of the integration period.*

t_{IP} (mS)	$L\{x_f\}$	$l\{x_f\}$	$V\{x_f\}$	$l\{\omega\}$ (deg)	$V\{\omega\}$ (deg)
2	466.667	464	2.667	13.920	0.080
3	311.111	310	1.111	13.950	0.050
4	233.333	232	1.333	13.920	0.080
5	186.667	185	1.667	13.875	0.125
6	155.556	155	0.556	13.950	0.050
7	133.333	133	0.333	13.965	0.035
8	116.667	115	1.667	13.800	0.200
9	103.704	103	0.704	13.905	0.095
10	93.333	94	-0.667	14.100	-0.100
11	84.848	85	-0.152	14.025	-0.025
12	77.778	77	0.778	13.860	0.140

The mean value of angle ω is $\overline{\omega} = 13.934^0$ with a standard deviation of $m(\omega) = \pm 0.107^0$. The spatial residual error has a mean value of $V = 0.066^0$. Object space angular values, corresponding to measured frame buffer coordinates, are plotted in Figure 3-18 as a function of the integration period.

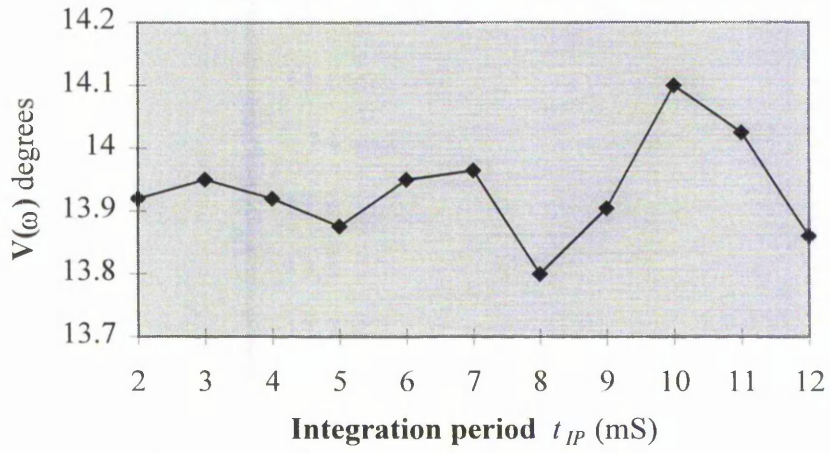

Figure 3-18 *Spatial angle measurements from variations of the integration period.*

A maximum absolute frame buffer coordinate error of less than three pixels was recorded. The increasing spatial angular error trend is attributed to the combination of the discrepancy between the nominal and actual values of the rotational speed and the inverse (1/x) rule indicated by the X axis algorithm.

VARIATION OF THE ROTATIONAL SPEED:

Experimental conditions: $t_{IP} = 10mS$, $\omega = 14^0$.

TABLE 3-4 *Accuracy in variations of the rotational speed.*

U_r (mS)	$L\{x_f\}$	$l\{x_f\}$	$V\{x_f\}$	$l\{\omega\}$ (deg)	$V\{\omega\}$ (deg)
0.5	466.667	466	0.667	13.980	0.020
1.0	233.333	232	1.333	13.920	0.080
1.5	155.556	155	0.556	13.950	0.050
2.0	116.667	116	0.667	13.920	0.080
2.5	93.334	94	-0.667	14.100	-0.100
3.0	77.778	77	0.778	13.860	0.140
3.5	66.667	67	-0.333	14.070	-0.070
4.0	58.334	58	0.333	13.920	0.080
4.5	51.852	51	0.852	13.770	0.230
5.0	46.667	46	0.667	13.800	0.200
5.5	42.424	43	-0.576	14.190	-0.190
6.0	38.889	39	-0.111	14.040	-0.040

The mean value of angle ω calculates to $\overline{\omega} = 13.960^0$ and the standard deviation is $m(\omega) = \pm 0.130^0$. The mean spatial residual error is $V = 0.040^0$. Object space angle measurements, calculated from frame buffer coordinate measurements, are plotted in Figure 3-19 as a function of the rotational speed.

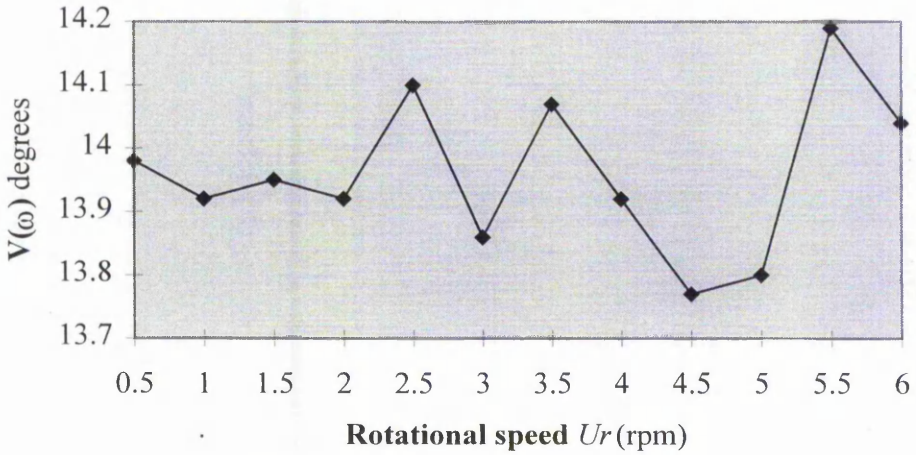

Figure 3-19 *Spatial angle measurements from variations of the rotational speed.*

A maximum absolute frame buffer coordinate error of less than two pixels was recorded. The non-linear increase in spatial angular error with increasing rotational speed is again attributed to the inverse rule governing X axis frame buffer coordinates.

VARIATION OF THE ANGLE ω:

Experimental conditions: $U_r = 5rpm$, $t_{IP} = 10mS$.

TABLE 3-5 *Accuracy in variations of the angle* ω.

ω (deg)	$L\{x_f\}$	$l\{x_f\}$	$V\{x_f\}$	$l\{\omega\}$ (deg)	$V\{\omega\}$ (deg)
15	50.0	49	1	14.7	0.3
30	100.0	100	0	30.0	0.0
45	150.0	149	1	44.7	0.3
60	200.0	200	0	60.0	0.0
75	250.0	248	2	74.4	0.6
90	300.0	299	1	89.7	0.3
105	350.0	348	2	104.4	0.6
120	400.0	399	1	119.7	0.3
135	450.0	448	2	134.4	0.6

The mean spatial residual error is $V = 0.333^0$ and the standard deviation is $m(\omega) = \pm 0.424^0$. Figure 3-20 plots the spatial residual error as a function of angle ω.

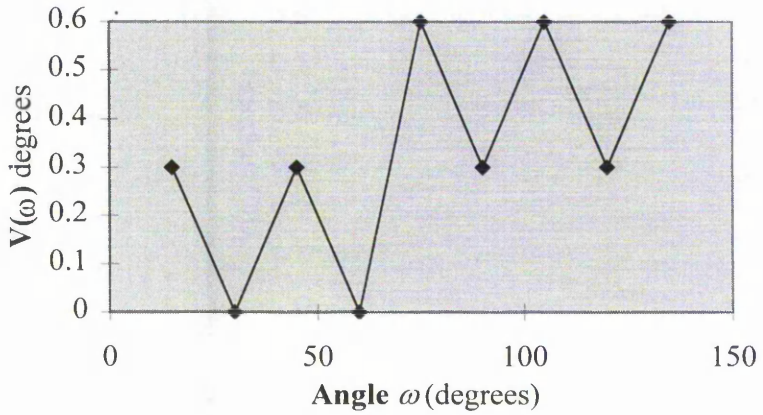

Figure 3-20 *Spatial residual error from variations of the angle* ω.

A maximum absolute frame buffer coordinate error of exactly two pixels was recorded. Due to the presence of spatial quantisation noise, the effects of the inaccuracy of the rotational speed can just be identified.

3.7.3.2 Y Axis Algorithm Verification

To verify the Y axis algorithm, equation 3-14 of section 3.6.3 is used. Solving for Y axis frame buffer coordinates y_f gives -

$$y_f = C_y + \frac{f}{s_y(Z_c - r)} Y_p$$

Since the interior orientation parameters of the camera are not known, nominal values for the Y axis image centre and the lens principal distance are assumed. Therefore, the Y axis image centre C_y is assigned a value of 256 and, unless otherwise stated, the 50mm nominal focal length lens is used. Also, from section 3.6.3 the Y axis scale factor s_y is 13μm. The notation used in presenting the results is as follows:

- column $L\{y_f\}$ shows the calculated frame buffer coordinates in the Y axis;
- column $l\{y_f\}$ shows the corresponding measured coordinates;
- column $V\{y_f\}$ shows the resultant error residual;
- column $l\{Y_p\}$ presents Y axis spatial distances extrapolated from corresponding frame buffer coordinate measurements;
- column $V\{Y_p\}$ presents the associated spatial residual error.

VARIATION OF OBJECT SPACE Y AXIS DISTANCE:

Experimental conditions: $r = 0.05m$, $Z_c = 1.25m$.

TABLE 3-6 *Accuracy in variations of* Y_P.

Y_P (m)	$L\{y_f\}$	$l\{y_f\}$	$V\{y_f\}$	$l\{Y_P\}$ (m)	$V\{Y_P\}$ (mm)
-0.060	63.692	56	7.692	-0.06240	2.40
-0.055	79.718	73	6.718	-0.05710	2.10
-0.050	95.744	89	6.744	-0.05210	2.10
-0.045	111.769	105	6.769	-0.04711	2.11
-0.040	127.795	121	6.795	-0.04212	2.12
-0.035	143.820	138	5.820	-0.03682	1.82
-0.030	159.846	154	5.846	-0.03182	1.82
-0.025	175.872	170	5.872	-0.02683	1.83
-0.020	191.897	186	5.897	-0.02184	1.84
-0.015	207.923	203	4.923	-0.01654	1.54
-0.010	223.949	218	5.949	-0.01186	1.86
-0.005	239.974	235	4.974	-0.00655	1.55
0	256.000	251	5.000	-0.00156	1.56
0.005	272.026	268	4.026	0.00374	1.26
0.010	288.051	284	4.051	0.00874	1.26
0.015	304.077	300	4.077	0.01373	1.27
0.020	320.103	316	4.103	0.01872	1.28
0.025	336.128	333	3.128	0.02402	0.98
0.030	352.154	349	3.154	0.02902	0.98
0.035	368.179	365	3.179	0.03401	0.99
0.040	384.205	381	3.205	0.03900	1.00
0.045	400.231	397	3.231	0.04399	1.01
0.050	416.256	413	3.256	0.04898	1.02
0.055	432.282	430	2.282	0.05429	0.71
0.060	448.308	446	2.308	0.05928	0.72

From Table 3-6 the mean residual error has a value of $V = 1.48$ mm and the standard deviation is $m(Y_P) = \pm 1.56$ mm. The object space residual error as a function of Y axis spatial distance is shown in Figure 3-21.

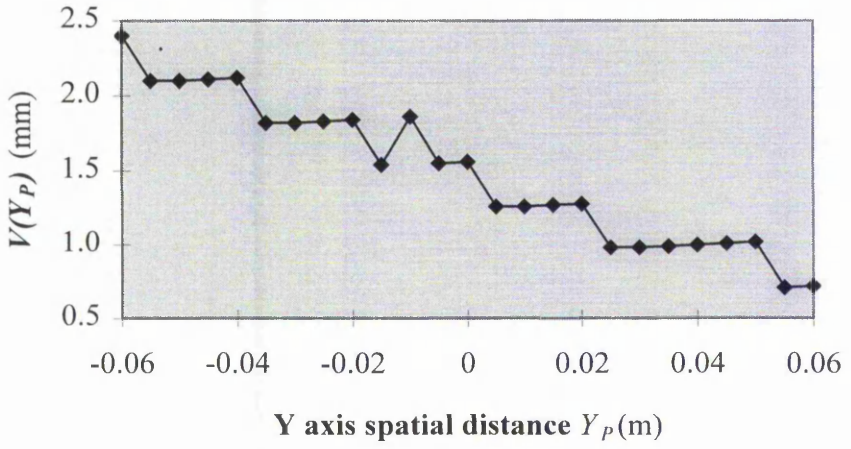

Figure 3-21 *Spatial residual error from variations of* Y_P.

A maximum absolute frame buffer coordinate error of eight pixels was recorded. Evidently, this error is more significant than that present in the X axis results. This is attributed to inaccuracies in the interior orientation parameters of the camera: Table 3-6 shows that the Y axis image centre occurs at pixel 251 instead of the nominal 256. In addition, the linear trend in error decrease indicates that the principal distance of the lens is larger than its nominal value. Indeed, extrapolation yields a true principal distance value of approximately 52mm.

VARIATION OF THE PRINCIPAL DISTANCE:

Experimental conditions: $r = 0.05m$, $Z_c = 1.25m$, $Y_P = 0.05m$.

TABLE 3-7 *Accuracy in variations of the principal distance.*

f_{nom} (mm)	$L\{y_f\}$	$l\{y_f\}$	$V\{y_f\}$	$l\{Y_P\}$ (m)	$V\{Y_P\}$ (mm)
25	336.128	331	5.128	0.0468	3.2
50	416.256	413	3.256	0.0490	1.0
75	496.384	488	8.384	0.0483	1.7

The mean spatial error and the standard deviation are of limited consequence in this experiment since the deviation of the lens principal distance from the nominal focal length value is significant.

The maximum frame buffer coordinate error is nine pixels. It is concluded that, principally, the discrepancy between nominal and actual lens principal distance gives rise to this error.

VARIATION OF THE RADIAL DISTANCE:

Experimental conditions: $Z_c = 1.25m$, $Y_P = 0.05m$.

TABLE 3-8 *Accuracy in variations of the radial distance.*

r (m)	$L\{y_f\}$	$l\{y_f\}$	$V\{y_f\}$	$l\{Y_P\}$ (m)	$V\{Y_P\}$ (mm)
0.025	412.986	414	-1.014	0.04839	1.61
0.050	416.256	416	0.256	0.04800	2.00
0.075	419.666	420	-0.334	0.04817	1.83
0.100	423.224	425	-1.776	0.04859	1.41
0.125	426.940	427	-0.060	0.04809	1.91
0.150	430.825	432	-1.175	0.04840	1.60
0.175	434.891	435	-0.109	0.04811	1.89
0.200	439.150	440	-0.850	0.04830	1.70

The residual error has a mean value of $V = 1.74$ mm and the standard deviation is $m(Y_P) = \pm 1.87$ mm. Figure 3-22 plots the spatial residual error as a function of the radial distance r.

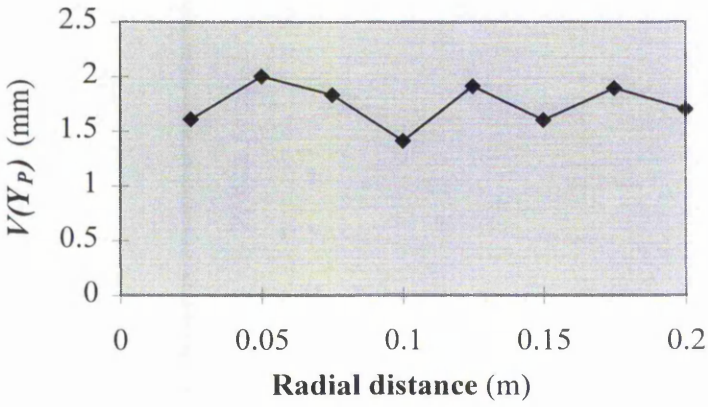

Figure 3-22 *Spatial residual error from variations of r .*

The absolute frame buffer coordinate error is within two pixels, and its linear and constant trend indicates that the image centre error is the cause.

VARIATION OF PRINCIPAL POINT TO ROTATION AXIS DISTANCE:

Experimental conditions: $Y_p = 0.05m$, $r = 0.05m$.

TABLE 3-9 *Accuracy in variations of the depth* Z_C.

Z_C (m)	$L\{y_f\}$	$l\{y_f\}$	$V\{y_f\}$	$l\{Y_p\}$ (m)	$V\{Y_p\}$ (mm)
1.0	458.429	462	-3.571	0.05088	-0.88
1.2	423.224	427	-3.776	0.05113	-1.13
1.4	398.450	400	-1.550	0.05054	-0.54
1.6	380.069	381	-0.931	0.05037	-0.37
1.8	365.890	366	-0.110	0.05005	-0.05
2.0	354.619	354	0.619	0.04969	0.31
2.2	345.445	346	-0.555	0.05031	-0.31
2.4	337.833	337	0.833	0.04949	0.51

From the above table, the mean residual error is $V = -0.31$ mm with a standard deviation of $m(Y_p) = \pm\ 0.65$ mm. Figure 3-23 shows the plot of the spatial residual error as a function of depth.

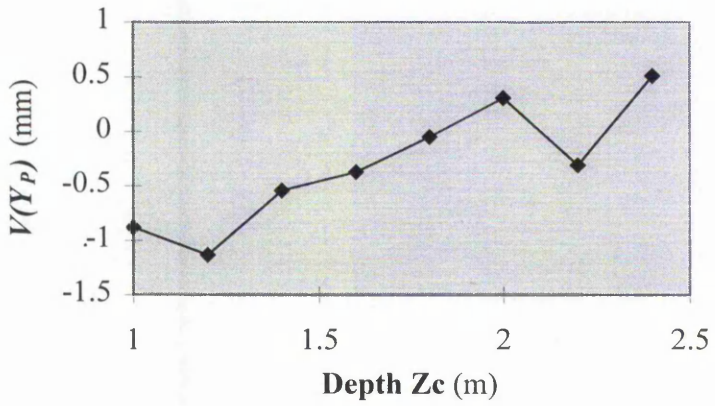

Figure 3-23 *Spatial residual error from variations of the depth* Z_C.

A maximum absolute frame buffer coordinate error of four pixels was recorded. Inaccuracies in the lens principal distance and the measurement of the radial distance r and/or in the exterior orientation parameters of the camera are thought to be the cause of the recorded errors.

3.7.4 Discussion of the Experimental Results

The results presented in section 3.7.2 indicate that imaging errors produced by inconsistency in the rotational speed are indiscernible. Although the consistency of the integration period proved more difficult to assess, the fact that it is produced by a digital port clocked by a crystal oscillator relaxes the uncertainty associated with its precision. Furthermore, since all X axis accuracy experiments rely on the consistency of the integration period and no erratic behaviour was recorded in any of them, it may be concluded that the line-transfer signal is sufficiently precise. More rigorous exercise of the precision of this signal is undertaken in the relevant experiments with the three-dimensional system.

Reasonable results regarding the verification of the two-dimensional system model were obtained through the accuracy experiments. In examining these results, it is imperative to note that the system calibration parameters are not known; rather, the experimental system is assumed to be perfectly aligned, i.e. strictly conforming to the derived geometrical model. Furthermore, the values of space resolving critical parameters, such as the lens principal

distance, the rotational speed and radial and angular distances, have been assumed to be exactly equal to their nominal or measured values. Both of these assumptions cannot be met in practice, thus compromising the spatial resolving power of the two-dimensional system.

4. THE STEREOSCOPIC SYSTEM

4.1 Introduction

This chapter presents the analysis of the stereoscopic line-scan system. The principal aim of this analysis is to derive an analytical model for the line-scan system that will allow the measurement of a three-dimensional object workspace from a pair of perspective images. The discussion starts with the introduction of the stereoscopic region and the identification of the parameters determining its spatial extent. To ascertain the spatial resolving properties of the stereoscopic line-scan system, the spatial sampling pattern is presented and the parameters affecting the voxel size are discussed.

In order to produce object space metric information from the line-scan system, the development of an analytical model is required. A primary consideration regarding the development of such a model in this work is that the geometry of each camera is considered individually and no dependency between them is assumed. This rigorous approach is consistent with the majority of existing photogrammetric and machine vision research work based on area-array cameras and has been demonstrated by numerous researchers, e.g.[64,65]. However, to the best of the author's knowledge, no publications addressing the issues of both rigorous calibration and space intersection on close-range stereoscopic line-scan systems exist.

The topic of camera calibration has attracted considerable attention from scientists and researchers of different disciplines. This is because of its extended field of practical applications and significance towards improving the metrological accuracy of photogrammetric and machine vision systems. Traditionally, calibration was exploited by photogrammetrists in the refinement of systems for the production of topographic maps, but calibration techniques have been employed in such diverse fields as autonomous robot forklifts[66] and satellite photogrammetry[67], to stereometric microscopy[68] and neurosurgery[69]. More recently, following substantial cost reductions in computing power and quality improvements in electronic image sensors[70,71,72,73], the topic of calibration has received renewed attention through the research efforts of the machine, robot and computer vision communities[74,75,76,77,78,79,80,81,82,83,84,85,86].

One rigorous resection method employed to compute the camera calibration parameters entails the mathematical modelling of the object to image space projective transformation through the *collinearity equations*[87,88]. Other calibration methods include traditional photogrammetric interior orientation carried out by goniometers combined with Euclidean geometry space resection, non-parametric calibration[89], calibration without point feature extraction[78,90], self-calibration[91], the Direct Linear Transformation[92,93] (DLT), and the bundle adjustment[94] (this method utilises the collinearity equations).

Although numerous approaches to the calibration of photographic and electronic area array cameras are available, the established base of research publications addressing the calibration of line-scan sensors is comparatively limited. To the author's knowledge, the most comprehensive calibration method specifically developed for line-scan cameras is that of Horaud *et al*[95]. However, Horaud's method does not take the temporal parameters of the two-dimensional line-scan image production into account. This is because, in many applications, the image information produced by the line-scan sensor is processed on a line-by-line basis (see section 2.4), rather that in two-dimensional frames. The calibration model developed in this research addresses this issue in the context of rotational object motion. In addition, the camera parameters are derived explicitly.

4.2 The Stereoscopic Region

Figure 4-1 depicts a plan view of the stereoscopic camera arrangement. The object of interest is assumed to be nominally cylindrical, with its axis of cylindrical symmetry coincident with the axis of rotation. The slit fields of view of both line-scan cameras are coincident at a range Z_T from the rotation axis, that is, on the surface of the cylindrical object. Further, it is assumed that the distance between each camera and the rotation axis is equal.

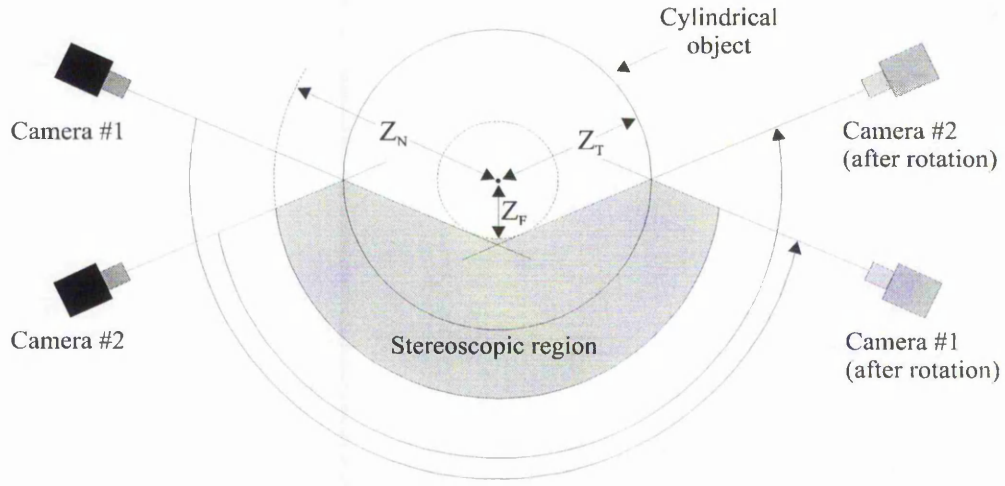

Figure 4-1 *The stereoscopic region.*

From sections 2.4 and 2.5.1, the angular field of view of the rotating object line-scan system is produced only when relative motion between the sensor and the object exists and is limited only by the available frame buffer memory. However, the extent of the stereoscopic region, formed at the mutual volume of the left and right camera fields of view, depends also on geometric parameters of the line-scan arrangement, such as the camera basewidth and convergence angles. This is illustrated in Figure 4-1 where it can be seen that the stereoscopic region, shown as the shaded area, is bound by the *near* (Z_N) and *far* (Z_F) ranges. It will become apparent that, in the context of the stereoscopic system, the depth planes are arranged concentrically with the rotation axis. Consequently, the near and far boundaries of the stereoscopic region also take the form of concentric cylinders of radius Z_N and Z_F, respectively.

The stereoscopic region extends from the cylinder of radius Z_F towards the cameras and the near boundary Z_N is imposed by physical constraints, namely the smallest range from the lens at which acceptable focus can be achieved.

To evaluate the far boundary, consider Figure 4-2. The camera basewidth is signified by B, the equal camera convergence angles by φ, and t_Z denotes the distance from the lens perspective centre to the rotation axis. The cameras converge at distance Z_T from the rotation axis.

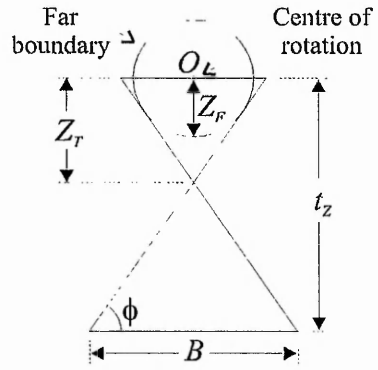

Figure 4-2 *Evaluating the 'far' boundary* Z_F.

It can be shown from Figure 4-2 that the radius of the circle which defines the far boundary is -

$$Z_F = \frac{BZ_T \sin\varphi}{2(t_z - Z_T)} \tag{4-1}$$

Therefore, the far boundary of the stereoscopic region depends on the camera basewidth, the camera convergence angles and the distance from the cameras to the rotation axis. It can also be appreciated that the volume enclosed by the cylinder of radius Z_F and the extent of the Y axis field of view does not form part of the stereoscopic region. This gives rise to a cylindrical *volume of occlusion* which is coaxial with the rotation axis. The radius of this cylindrical volume is given by equation 4-1.

4.3 The Spatial Sampling Pattern

The spatial sampling properties of the stereoscopic line-scan system can be identified by consideration of the sampling pattern, depicted in Figure 4-3. This pattern stems from the discrete nature of the sensor's elements. The whole of the stereoscopic region is made up of individual volume elements, or *voxels*[96], each corresponding to the mutual volume of back-projected photosite elements from the left and right camera through the camera optics to object space. Unless subpixelation techniques are employed, the location of a point in object space that falls anywhere within the volume of a voxel cannot be determined to an accuracy better than the spatial dimensions of that voxel. Since this translates to an

uncertainty of ± 1 pixel, or two pixels in absolute value, individual voxels stretch between two consecutive depth planes. The instantaneous fields of view of both cameras are nominally parallel to the rotation axis, therefore the depth planes in Figure 4-3 (denoted r_1, r_2, etc.) are of cylindrical shape and coaxial to the rotation axis.

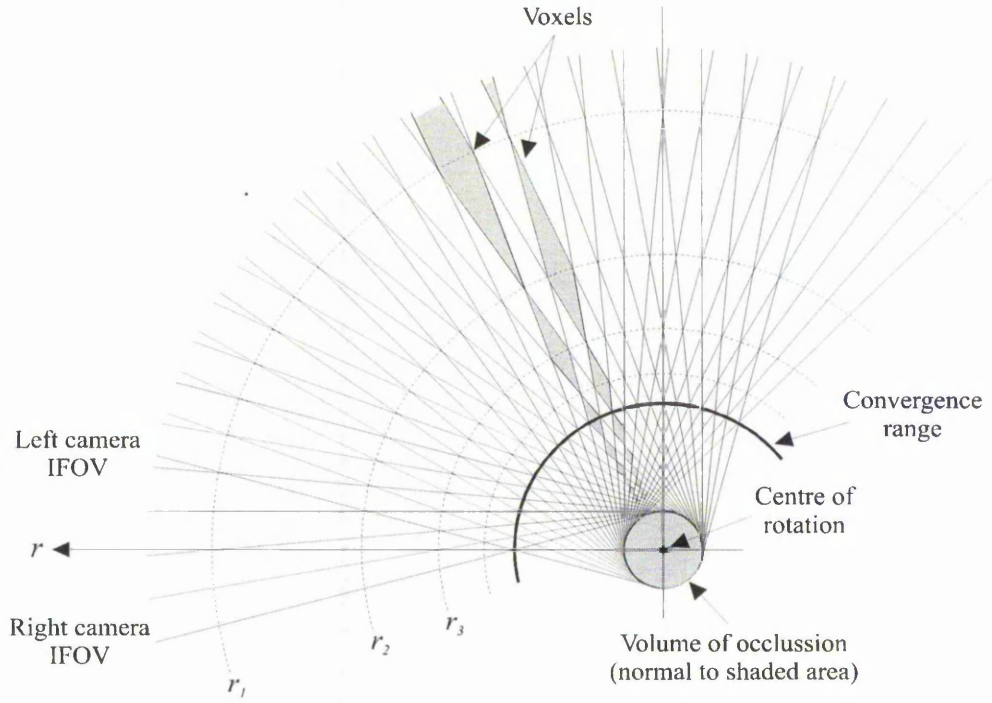

Figure 4-3 *The spatial sampling pattern produced after object rotation (plan view).*

From this diagram it is also apparent that the spatial resolution of the line-scan system is highest at the minimum resolvable distance from the rotation axis, i.e. closest to the cylinder of occlusion, and gets progressively lower as this distance is increased. In addition, the spatial resolution is constant over the circumference of any cylinder coaxial to the rotation axis lying within the stereoscopic region.

As voxels represent the way a digital imaging system samples space, all of the system parameters governing image production affect their size. With reference to Figure 4-3, voxel dimensions depend on -

Main (Y) axis:

i. the lens principal distance;

ii. the photosite dimensions;

iii. the camera to object range.

Any radial axis (normal to the main axis):

i. the integration period;

ii. the rotational speed;

iii. the radial separation from the rotation axis;

iv. the camera convergence angles;

v. the camera basewidth.

The number of variables involved in the determination of the voxel size precludes the practical derivation of a closed-form solution. Thus, in order to quantify the voxel dimensions, the three-dimensional space resolving algorithms must first be introduced. Section 4.5.4 details the calculation of the voxel size.

4.4 Stereoscopic Line-Scan System Algorithms

4.4.1 Geometry of the Stereoscopic System

The geometry of the stereoscopic system is illustrated in Figure 4-4. The coordinates of points in object space are defined in the three-dimensional Cartesian object space coordinate system (X, Y, Z). The purpose of this coordinate system is to map the object workspace and it is required to be independent from the camera coordinate systems.

A three-dimensional rectangular coordinate system is established for each camera, denoted by (X_{cL}, Y_{cL}, Z_{cL}) and (X_{cR}, Y_{cR}, Z_{cR}) for the left and right camera, respectively. Both cameras are rotated inwards about the Y axis of the left and right camera-centred coordinate systems by angles φ_L and φ_R such that their optical axes converge at point C. Both Z_c axes are coincident with the respective optical axis, and the Y_c axes are parallel to the line defined by the sensor's photosites. The X_c axes are normal to the respective Y_cZ_c plane, and both camera coordinate systems are centred on the lenses' perspective centre.

The origin of the object space coordinate system (X, Y, Z) lies on the rotation axis, its exact location determined by the Y axis translation component between the camera and object space systems. In the nominally aligned system depicted in Figure 4-4, the X_cZ_c planes of both cameras and the object space coordinate systems are coplanar and normal to the rotation axis. However, for the rigid transformation (see Appendix I) from the object space to either camera coordinate system to be valid, all three coordinate systems must be right-handed; hence the opposite direction of the Y_{cL} and Y_{cR} axes with respect to the Y axis.

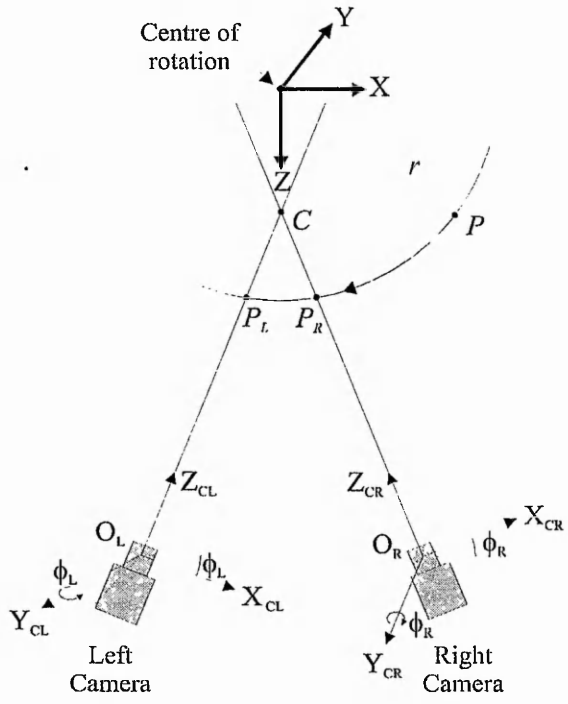

Figure 4-4 *Geometry of the stereoscopic system.*

An arbitrary point P, located distance r from the rotation axis, is subjected to a clockwise rotation such that it is imaged first by the right camera at position P_R and then by the left camera at position P_L. Provided that the cameras' optical axes do not converge on the rotation axis, stereoscopic parallax information will exist in the left and right perspective images. This can be seen in Figure 4-5 where the left and right camera perspective images of a cylindrical structure are reproduced.

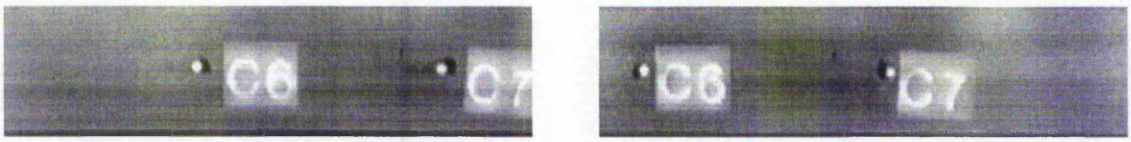

Figure 4-5 *(a) Left and (b) right perspective images depicting disparity.*

If the convergence point is located on the rotation axis there will still be a lateral shift due to the time delay between the capture of the spatial point by the left and right cameras, but no stereoscopic parallax since the left and right perspectives are essentially the same. This is discussed in more detail in Chapter 5.

4.4.2 The System Model

The development of a suitable model for the line-scan system begins with the derivation of a *functional* model. The functional model consists of a set of mathematical algorithms that describe the operation of the line-scan system without taking stochastic effects into account.

Since the left and right cameras are geometrically independent, it is not valid to derive the system algorithms on the basis of perfect relative alignment of the two sensors since these would break down under real, non-perfect alignment conditions. The derivation of a suitable model, whereby each camera has six degrees of freedom[97] (three for position and three for orientation), is presented in the following text. The discussion is divided into the following four phases -

I. *Phase one* details the derivation of the two-dimensional X and Y axis algorithms in a camera-centred coordinate system assuming perfect system alignment.

II. *Phase two* extends the two-dimensional X and Y axis algorithms to encompass an arbitrary object space coordinate system, hence modelling the non-perfectly aligned system.

III. *Phase three* details the calibration of the line-scan system and introduces the stochastic model.

IV. *Phase four* addresses space intersection.

PHASE 1: THE CAMERA-CENTRED ALGORITHMS

The geometrical model of the left camera system introduced in Figure 4-4 is illustrated in more detail in Figure 4-6. It is subsequently demonstrated that the algorithms pertaining to this model are also valid for the right camera system.

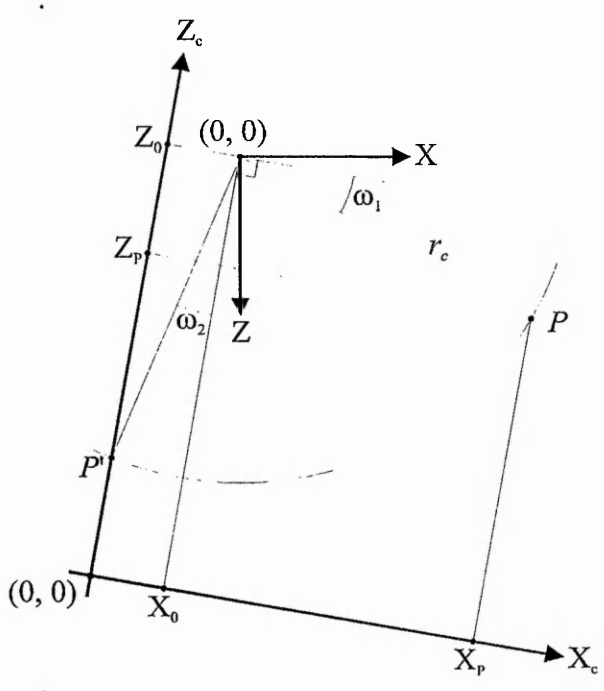

Figure 4-6 *Detail of the left camera geometry.*

Let the coordinate systems' alignment details be identical to those introduced in section 4.4.1, that is, a perfect system alignment. In addition, let the origin of the object space coordinate system have coordinates $(X_0,\ Y_0,\ Z_0)$ in the camera coordinate system and those of an arbitrary point P be $(X_P,\ Y_P,\ Z_P)$.

X Axis Algorithm Derivation

With reference to Figure 4-6, angle ω may be expressed as -

$$\omega = \frac{\pi}{2} - \omega_1 + \omega_2, \quad \text{if} \quad -\frac{\pi}{2} < \omega_1 \leq \frac{\pi}{2}$$
$$\omega = \frac{3\pi}{2} + \omega_1 + \omega_2, \quad \text{if} \quad \frac{\pi}{2} < \omega_1 \leq \frac{3\pi}{2} \tag{4-2}$$

Further, angles ω_1 and ω_2 are given by -

$$\omega_1 = sin^{-1}\left(\frac{Z_P - Z_0}{r_P}\right)$$
$$\omega_2 = sin^{-1}\left(\frac{X_0}{r_P}\right) \tag{4-3}$$

where the radius of point P in the camera coordinate system is -

$$r_P = \sqrt{\left(X_P - X_0\right)^2 + \left(Z_P - Z_0\right)^2} \tag{4-4}$$

Combining equations 3-8, 4-2 and 4-3 produces -

$$x_f = s_x\left[\frac{\pi}{2} - sin^{-1}\left(\frac{Z_P - Z_0}{r_P}\right) + sin^{-1}\left(\frac{X_0}{r_P}\right)\right], \quad \text{if} \quad -\frac{\pi}{2} < \omega_1 \leq \frac{\pi}{2}$$

$$x_f = s_x\left[\frac{3\pi}{2} + sin^{-1}\left(\frac{Z_P - Z_0}{r_P}\right) + sin^{-1}\left(\frac{X_0}{r_P}\right)\right], \quad \text{if} \quad \frac{\pi}{2} < \omega_1 \leq \frac{3\pi}{2} \tag{4-5}$$

Equation 4-5 expresses the transformation from the camera to the frame buffer coordinate system in the X axis.

Y Axis Algorithm Derivation

The analysis of the two-dimensional system in chapter 3 derived a Y axis algorithm for the rotating object line-scan system. However, it is noted from Figure 4-6 that, at the instant of image capture, the spatial point P will have assumed location P'. In addition, the camera and object space Y axes have opposite direction necessitating a sign inversion in equation 3-14.

Hence, equations 3-12 and 3-14 give -

$$y_f = C_y - \frac{f}{s_y Z_{P'}} Y_P \tag{4-6}$$

where y_f is the Y axis frame buffer coordinate of point P, C_y is the Y axis image centre, f is the lens principal distance, s_y is the Y axis scale factor and $Z_{P'}$ is the range from the lens perspective centre to point P'. With reference to Figure 4-6 the depth $Z_{P'}$ is -

$$Z_{P'} = Z_0 - \sqrt{r_P^2 - X_0^2} \tag{4-7}$$

Hence, combining equations 4-6 and 4-7, the transformation from the camera to the frame buffer coordinate system in the Y axis may be expressed as -

$$y_f = C_y - \frac{f}{s_y} \cdot \frac{Y_P}{Z_0 - \sqrt{r_P^2 - X_0^2}} \tag{4-8}$$

The topic of lens non-linearity and their modelling is well-documented[98,99,100,101]. Appendix II extends the Y axis algorithm to correct for up to fourth order radial lens distortion components. However, practical constraints have precluded the use of a lens distortion model in this work. This is discussed in more detail in section 4.5.1.

PHASE 2: REAL SYSTEM

A rigid transformation, described in Appendix I, is implemented in order to model the arbitrary pose and orientation of the camera in the reference object space coordinate system. This is given by -

$$\boldsymbol{p}_c = \boldsymbol{R}\boldsymbol{p} + \boldsymbol{t} \tag{4-9}$$

where $\boldsymbol{p}_c = \begin{bmatrix} X_P & Y_P & Z_P \end{bmatrix}^{\mathrm{T}}$ and $\boldsymbol{p} = \begin{bmatrix} X & Y & Z \end{bmatrix}^{\mathrm{T}}$ denote the camera and object space coordinates of point *P*, respectively.

From Figure 4-6, the translation vector $\boldsymbol{t}$ is -

$$\boldsymbol{t} = \begin{bmatrix} -X_0 & -Y_0 & Z_0 \end{bmatrix}^{\mathrm{T}}$$

Combining 4-3, 4-4 and 4-9 -

$$\omega_1 = \sin^{-1}\left(\frac{r_{31}X + r_{32}Y + r_{33}Z}{\sqrt{\left(r_{11}X + r_{12}Y + r_{13}Z\right)^2 + \left(r_{31}X + r_{32}Y + r_{33}Z\right)^2}}\right)$$

(4-10)

$$\omega_2 = -\sin^{-1}\left(\frac{t_x}{\sqrt{\left(r_{11}X + r_{12}Y + r_{13}Z\right)^2 + \left(r_{31}X + r_{32}Y + r_{33}Z\right)^2}}\right)$$

Substituting 4-10 into 4-5 the X axis system algorithm is obtained as -

PTO.

$$x_f = s_x\left[\frac{\pi}{2} - sin^{-1}\left(\frac{r_{31}.X + r_{32}.Y + r_{33}.Z}{\sqrt{\left(r_{11}.X + r_{12}.Y + r_{13}.Z\right)^2 + \left(r_{31}.X + r_{32}.Y + r_{33}.Z\right)^2}}\right) - sin^{-1}\left(\frac{t_x}{\sqrt{\left(r_{11}.X + r_{12}.Y + r_{13}.Z\right)^2 + \left(r_{31}.X + r_{32}.Y + r_{33}.Z\right)^2}}\right)\right]$$

if $-\frac{\pi}{2} < \omega_1 \leq \frac{\pi}{2}$ (4-11a)

and -

$$x_f = s_x\left[\frac{3\pi}{2} + sin^{-1}\left(\frac{r_{31}.X + r_{32}.Y + r_{33}.Z}{\sqrt{\left(r_{11}.X + r_{12}.Y + r_{13}.Z\right)^2 + \left(r_{31}.X + r_{32}.Y + r_{33}.Z\right)^2}}\right) - sin^{-1}\left(\frac{t_x}{\sqrt{\left(r_{11}.X + r_{12}.Y + r_{13}.Z\right)^2 + \left(r_{31}.X + r_{32}.Y + r_{33}.Z\right)^2}}\right)\right]$$

if $\frac{\pi}{2} < \omega_1 \leq \frac{3\pi}{2}$ (4-11b)

Writing the Y axis system algorithm (equation 4-8) in consistent form -

$$y_f = C_y - \frac{f}{s_y}\left(\frac{r_{21}.X + r_{22}.Y + r_{23}.Z - t_y}{t_z - \sqrt{\left(r_{11}.X + r_{12}.Y + r_{13}.Z\right)^2 + \left(r_{31}.X + r_{32}.Y + r_{33}.Z\right)^2 - t_x^2}}\right) \quad (4\text{-}12)$$

The derivation of the right camera system algorithms is now considered. The right camera geometry is illustrated in Figure 4-7.

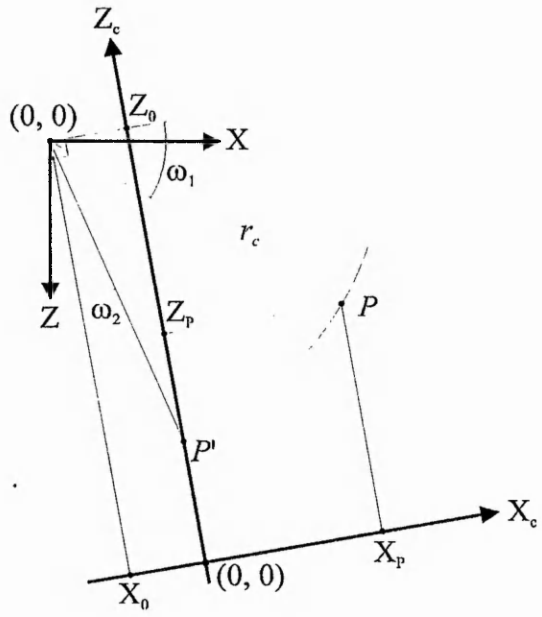

Figure 4-7 *Detail of the right camera geometry.*

From this diagram it can be seen that -

$$\omega = \frac{\pi}{2} - \omega_1 - \omega_2, \quad \text{if} \quad -\frac{\pi}{2} < \omega_1 \leq \frac{\pi}{2}$$
$$\omega = \frac{3\pi}{2} + \omega_1 - \omega_2, \quad \text{if} \quad \frac{\pi}{2} < \omega_1 \leq \frac{3\pi}{2} \tag{4-13}$$

where the angles ω_1 and ω_2 are given by equation 4-3. However, the translation vector for the right camera system is -

$$\boldsymbol{t} = \begin{bmatrix} X_0 & -Y_0 & Z_0 \end{bmatrix}^{\mathrm{T}} \tag{4-14}$$

so that the right camera X and Y axis algorithms are notationally identical to those of the left camera system.

PHASE 3: SYSTEM CALIBRATION

The aim of the calibration process is to estimate the interior, exterior and temporal parameters of the line-scan system, given the spatial coordinates of a number of control points and their corresponding frame buffer coordinates. The set of the control points and their corresponding frame buffer coordinates is referred to as the *control correspondence*, or just *correspondence*. The camera interior, exterior and temporal calibration parameters are embedded in the functional model algorithms, that is, equations 4-11 and 4-12. These parameters are summarised in Table 4-1.

TABLE 4-1 *The calibration parameters.*

Parameter	Type	Description
f	Interior	Lens principal distance
C_y	Interior	Y axis frame buffer image centre
ω, φ, κ	Exterior	Euler camera orientation angles
t_x, t_y, t_z	Exterior	The components of the translation vector
U_r	Temporal	Rotational speed
t_{IP}	Temporal	Integration period

Recalling equation 3-7 of the X axis scale factor, it is not possible to treat the rotational speed and the integration period as two independent unknowns; it is their combined effect that manifests itself in the images, requiring that the product of these two variables is treated as a single calibration unknown. This constraint is imposed by the operation of the system and does not, in any way, compromise the accuracy of the derived functional model.

As numerous researchers have shown, the estimation of the camera parameters is not very sensitive to the value of the image centre C_y. For instance, Tsai[75] has shown that three-dimensional coordinate measurement (using passive stereo triangulation and off-the-shelf cameras and lenses) with an accuracy of one part in 4,000 can be achieved assuming that the image centre occurs at its nominal value. Hence, in order to reduce the parameter space of the calibration model, it was decided not to incorporate the image centre in the set of the calibration unknowns.

Therefore, the calibration model employs eight unknowns for each camera and the stereoscopic system can be modelled using sixteen calibration parameters. Each control point contributes two equations from each image, thus the minimum number of control points required for calibration is four. This compares favourably with the calibration of an area-array camera, which, using a linear method and assuming that the camera lens is distortionless, requires a minimum of six control points for calibration. However, it should be noted that, in these conditions, the calibration of an area-array camera is a linear problem, whereas that of the line-scan system is non-linear. This non-linearity is a consequence of the rotational motion employed in the line-scan system.

In this work, calibration is performed using a *full-scale* iterative process. This method has been traditionally employed in photogrammetry, and has two important properties:

- all the calibration unknowns are calculated in one step;
- the parametrisation of the camera is not constrained by the process, i.e. changes in the functional model can be readily accommodated;

However, the method is computationally intensive and requires initial approximations for each unknown (see section 4.5.2).

Let $\boldsymbol{W}$ denote the set of the calibration parameters -

$$\boldsymbol{W} = \left\{ U_r t_{IPL},\ \omega_L,\ \varphi_L,\ \kappa_L,\ f_L,\ t_{xL},\ t_{yL},\ t_{zL},\ U_r t_{IPR},\ \omega_R,\ \varphi_R,\ \kappa_R,\ f_R,\ t_{xR},\ t_{yR},\ t_{zR} \right\}$$

Further, consider a point in object space having spatial coordinates (X, Y, Z) and corresponding frame buffer coordinates $(x_{fL},\ y_{fL})$ and $(x_{fR},\ y_{fR})$ produced by the left and right cameras, respectively. The four simultaneous equations resulting from the system algorithms can be expressed as -

$$\begin{aligned} x_{fL} - f_{xL}(X, Y, Z, \boldsymbol{W}_L) &= 0 \\ y_{fL} - f_{yL}(X, Y, Z, \boldsymbol{W}_L) &= 0 \\ x_{fR} - f_{xR}(X, Y, Z, \boldsymbol{W}_R) &= 0 \\ y_{fR} - f_{yR}(X, Y, Z, \boldsymbol{W}_R) &= 0 \end{aligned} \tag{4-15}$$

where $\boldsymbol{W}_L$ and $\boldsymbol{W}_R$ denote the set of the left and right camera calibration parameters, respectively. The above set of equations can be extended to account for n points imaged by both cameras -

$$\begin{aligned} x_{fLi} - f_{xL}(X_i, Y_i, Z_i, \boldsymbol{W}_L) &= 0 \\ y_{fLi} - f_{yL}(X_i, Y_i, Z_i, \boldsymbol{W}_L) &= 0 \\ x_{fRi} - f_{xR}(X_i, Y_i, Z_i, \boldsymbol{W}_R) &= 0 \\ y_{fRi} - f_{yR}(X_i, Y_i, Z_i, \boldsymbol{W}_R) &= 0 \end{aligned} \tag{4-16}$$

where $i \rightarrow \{1 \ldots n\}$.

In order to compute a solution for the calibration parameters, it is required to express the set of simultaneous equations 4-16 in matrix notation as -

$$\underset{(4n,16)}{\boldsymbol{A}} \; \underset{(16)}{\boldsymbol{x}} = \underset{(4n)}{\boldsymbol{0}}$$

where $\boldsymbol{x}$ is the column vector having the elements of $\boldsymbol{W}$ as its components, that is, the sixteen calibration unknowns, and $\boldsymbol{A}$ is a $4n \times 16$ matrix. However, due to the non-linearity of the system algorithms, a direct formulation of matrix $\boldsymbol{A}$ is not possible. To solve this problem, a *Newton-Raphson* iterative technique is employed. The formulation starts with the decomposition of the system algorithms into a suitable mathematical series. In particular, a *Taylor's series* expansion is utilised as follows.

Let the vector of the set of functions in equation 4-16 be denoted by $\boldsymbol{F}$, so that equation 4-16 is expressed as -

$$\boldsymbol{F(x) = 0}$$

Taylor's series expansion of the above expression in the local region of $\boldsymbol{x}$ gives -

$$F(x+dx) = F(x) + \frac{\partial F}{\partial x}dx + \frac{\partial^2 F}{\partial x^2}\frac{dx^2}{2!} + \ldots + \frac{\partial^n F}{\partial x^n}\frac{dx^n}{n!} \qquad (4\text{-}17)$$

where dx is a small increment in x, $n \to \infty$ and the terms inside the brackets are first and higher order partial derivatives of F with respect to x. In Euler's method the above series is truncated after second and higher order derivative terms giving -

$$F(x+dx) \approx F(x) + \frac{\partial F}{\partial x}dx \qquad (4\text{-}18)$$

But $F(x+dx) = 0$, so that equation 4-18 produces -

$$Jdx \approx -F(x) \qquad (4\text{-}19)$$

where - J is the Jacobian matrix of the system algorithms, i.e. the matrix of the first partial derivatives of F with respect to each calibration unknown, evaluated either at the initial approximations of the calibration unknowns or at the previous iterative step;

dx is the column vector of the corrections to be added to each calibration unknown after completion of the current iterative step;

$F(x)$ is the column vector of the values of each function in the set calculated at the initial approximations of the unknowns or at the previous iterative step.

Equation 4-19 is solved using linear matrix algebra, but the truncation of higher order terms in equation 4-17 requires that an iterative method is used.

Consider the evaluation of the elements constituting matrix J with reference to the system algorithms. The X and Y algorithms are first expressed as per equation 4-15. Subsequently, Taylor's series expansion is applied as shown in equation 4-19 giving -

$$j_{11}dU_r t_{IP} + j_{12}d\omega + j_{13}d\varphi + j_{14}d\kappa + j_{15}df + j_{16}dt_x + j_{17}dt_y + j_{18}dt_z \approx x_f - (x_f)_0 \quad (4\text{-}20)$$

$$j_{21}dU_r t_{IP} + j_{22}d\omega + j_{23}d\varphi + j_{24}d\kappa + j_{25}df + j_{26}dt_x + j_{27}dt_y + j_{28}dt_z \approx y_f - (y_f)_0 \quad (4\text{-}21)$$

where - j_{1i} and j_{2i}, $i \rightarrow \{1...8\}$, denote the first partial derivatives of the system algorithms with respect to each calibration parameter, provided in Appendix III;

x_f and y_f are the observed (measured) X and Y axis frame buffer coordinates of a point, respectively;

$(x_f)_0$ and $(y_f)_0$ are the X and Y axis frame buffer coordinates evaluated at the initial approximations of the calibration unknowns or at the solution of the previous iterative step, respectively;

Note that terms involving the spatial coordinates (X, Y, Z) of control points do not appear in equations 4-20 and 4-21 as the spatial location of control points is known and therefore constant. Furthermore, the system algorithms suggest that a number of partial derivative terms are equal to zero -

$$j_{15} = j_{17} = j_{18} = j_{21} = 0 \quad (4\text{-}22)$$

The following equations 4-23 to 4-26 are derived from 4-20, 4-21 and 4-22 with subscripts (L) and (R) indicating the camera they refer to. They are the four simultaneous equations that result from the system algorithms for each point imaged by both cameras.

$$j_{11L}dU_{rL}t_{IPL} + j_{12L}d\omega_L + j_{13L}d\varphi_L + j_{14L}d\kappa_L + j_{16L}dt_{xL} \approx x_{fL} - (x_{fL})_0 \quad (4\text{-}23)$$

$$j_{22L}d\omega_L + j_{23L}d\varphi_L + j_{24L}d\kappa_L + j_{25L}df_L + j_{26L}dt_{xL} + j_{27L}dt_{yL} + j_{28L}dt_{zL} \approx y_{fL} - (y_{fL})_0 \quad (4\text{-}24)$$

$$j_{11R}dU_{rR}t_{IPR} + j_{12R}d\omega_R + j_{13R}d\varphi_R + j_{14R}d\kappa_R + j_{16R}dt_{xR} \approx x_{fR} - (x_{fR})_0 \quad (4\text{-}25)$$

$$j_{22R}d\omega_R + j_{23R}d\varphi_R + j_{24R}d\kappa_R + j_{25R}df_R + j_{26R}dt_{xR} + j_{27R}dt_{yR} + j_{28R}dt_{zR} \approx y_{fR} - (y_{fR})_0 \quad (4\text{-}26)$$

Since the above equations are linear, a solution for the calibration unknowns is obtained through linear matrix algebra. If n control points are used -

$$\underset{(4n,16)}{\boldsymbol{J}}\ \underset{(16)}{\boldsymbol{dx}} \approx -\underset{(4n)}{\boldsymbol{F(x)}} \tag{4-27}$$

The Jacobian matrix $\boldsymbol{J}$ and column vectors $\boldsymbol{dx}$ and $\boldsymbol{F(x)}$ are partitioned as -

$$\underset{(4n,16)}{\boldsymbol{J}} = \begin{bmatrix} \underset{(2n,8)}{\boldsymbol{J}_L} & \underset{(2n,8)}{\boldsymbol{0}} \\ \underset{(2n,8)}{\boldsymbol{0}} & \underset{(2n,8)}{\boldsymbol{J}_R} \end{bmatrix} \qquad \underset{(16)}{\boldsymbol{dx}} = \begin{bmatrix} \underset{(8)}{\boldsymbol{dx}_L} \\ \underset{(8)}{\boldsymbol{dx}_R} \end{bmatrix} \qquad \underset{(4n)}{\boldsymbol{F(x)}} = \begin{bmatrix} \underset{(2n)}{\boldsymbol{F(x)}_L} \\ \underset{(2n)}{\boldsymbol{F(x)}_R} \end{bmatrix}$$

where subscripts (L) and right (R) denote the left and right camera, respectively. Only the left camera vectors and the Jacobian matrix are given below. The formulation for the right camera is identical, except that the subscripts (L) are changed with (R). From equations 4-23 and 4-24, matrix $\boldsymbol{J}_L$ is given by -

$$\underset{(2n,8)}{\boldsymbol{J}_L} = \begin{bmatrix} j_{11L1} & j_{12L1} & j_{13L1} & j_{14L1} & 0 & j_{16L1} & 0 & 0 \\ 0 & j_{22L1} & j_{23L1} & j_{24L1} & j_{25L1} & j_{26L1} & j_{27L1} & j_{28L1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ j_{11Ln} & j_{12Ln} & j_{13Ln} & j_{14Ln} & 0 & j_{16Ln} & 0 & 0 \\ 0 & j_{22Ln} & j_{23Ln} & j_{24Ln} & j_{25Ln} & j_{26Ln} & j_{27Ln} & j_{28Ln} \end{bmatrix}$$

Appendix III presents the individual elements of the above matrix with respect to the system algorithms. The vector of the corrections to the calibration parameters is -

$$\underset{(8)}{\boldsymbol{dx}_L} = \begin{bmatrix} dU_r t_{IPL} & d\omega_L & d\varphi_L & d\kappa_L & df_L & dt_{xL} & dt_{yL} & dt_{zL} \end{bmatrix}^{\mathrm{T}}$$

The vector of the functions $\boldsymbol{F(x)}_L$ is -

$$\underset{(2n)}{\boldsymbol{F(x)}_L} = \begin{bmatrix} x_{fL1} - (x_{fL1})_0 & y_{fL1} - (y_{fL1})_0 & \cdots & x_{fLn} - (x_{fLn})_0 & y_{fLn} - (y_{fLn})_0 \end{bmatrix}^{\mathrm{T}}$$

It is important to note that, although in the above formulation the calibration unknowns are evaluated using linear matrix algebra, the underlying problem is non-linear. In addition, the high dimensionality of the process means that there can be numerous local minima (hopefully well separated from the global minimum). Consequently, if convergence to the global minimum is to be achieved, initial approximations for all the calibration unknowns which are close enough to the target solution are required. This is discussed in section 4.5.2.

The Stochastic Model:

Various factors, such as the presence of spatial quantisation noise and inaccuracies in the functional model, mean that, in practice, the functional model can never fit the control correspondence data exactly. Therefore, solving for the calibration parameters is an *optimisation* process. A *residual vector* $\boldsymbol{r}$ is used to represent the residual error in each calibration unknown, and equation 4-27 is expressed as an exact equality -

$$\underset{(4n,16)}{\boldsymbol{J}}\ \underset{(16)}{\boldsymbol{dx}} = \underset{(4n)}{\boldsymbol{r}} - \underset{(4n)}{\boldsymbol{F(x)}} \tag{4-28}$$

where $\boldsymbol{r}$ is given by -

$$\underset{(4n)}{\boldsymbol{r}} = \left[r_{xL1} \quad r_{yL1} \quad \cdots \quad r_{xLn} \quad r_{yLn} \quad | \quad r_{xR1} \quad r_{yR1} \quad \cdots \quad r_{xRn} \quad r_{yRn} \right]^{\mathrm{T}}$$

A statistically optimal solution in each calibration parameter can be obtained by implementing the well-known technique of *least squares*[102,103]. This requires that redundant equations exist. A correspondence of n control points produces an overdetermined system of $4n$ simultaneous equations and the calibration parameters are obtained by least squares minimisation of the objective function F -

$$F = \sum_{i=1}^{n} \left\{ \left[\tilde{x}_{fiL} - x_{fiL}(\boldsymbol{x}) \right]^2 + \left[\tilde{y}_{fiL} - y_{fiL}(\boldsymbol{x}) \right]^2 + \left[\tilde{x}_{fiR} - x_{fiR}(\boldsymbol{x}) \right]^2 + \left[\tilde{y}_{fiR} - y_{fiR}(\boldsymbol{x}) \right]^2 \right\}$$

where $\tilde{x}_{fi}$ and $\tilde{y}_{fi}$ are the vectors of the observed X and Y axis frame buffer coordinates of control points defined for both cameras, and $x_{fi}(x)$ and $y_{fi}(x)$ are the corresponding X and Y frame buffer coordinates computed by the calibration process.

In other words, the calibration process seeks the vector of calibration unknowns x which minimises the sum of the squares of the differences between the measured and calculated frame buffer coordinates of the control points. Since the latter are functions of x, minimising the above objective function in the presence of normally distributed, i.e. Gaussian, noise leads to the statistically optimum set of calibration parameters.

With reference to equation 4-28, i.e. the algorithmic procedure outlined so far, the calibration problem can be stated as -

$$\|r\|^2 = \min_{dx} \|Jdx + F(x)\|^2$$

that is, a solution in dx that minimises the components of the residual vector in the least squares sense. In this work, the least squares minimisation is obtained by the *singular value decomposition* method, described in Press *et al* [104]. Due to the computational involvement of the calibration process, dedicated software code has been written to make its implementation possible. Details of this are given in section 4.5.

PHASE 4: SPACE INTERSECTION

The final step in extracting three-dimensional information from the object workspace is *space intersection*. For each image shown in Figure 4-8, a unique line passing through the perspective centre O, the spatial point P and its image p is defined. The point of intersection of these lines coincides ideally with the unknown point. However, lens distortions result in the lines $O_L P$ and $O_R P$ being skewed and non-intersecting[105]. Additionally, errors in the determination of the calibration parameters and in the measurement of the frame buffer coordinates of the spatial point will generally be present.

This gives rise to corresponding errors in the derived coordinates of the original point obtained by space intersection.

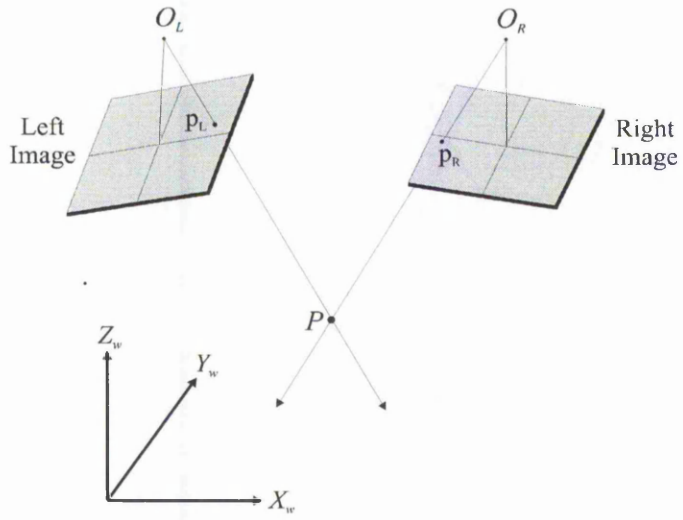

Figure 4-8 *Space intersection.*

Algebraically, space intersection is formulated in a similar way to the calibration process, utilising iterative linear matrix algebra. The aim is to compute a 4×3 matrix $\boldsymbol{A}$ that maps the computer image space to the object space, that is -

$$\boldsymbol{Ap} = \boldsymbol{f}$$

where $\boldsymbol{p} = \begin{bmatrix} X & Y & Z \end{bmatrix}^{\mathrm{T}}$ and $\boldsymbol{f} = \begin{bmatrix} x_{fL} & y_{fL} & x_{fR} & y_{fR} \end{bmatrix}^{\mathrm{T}}$ are the column vectors of the spatial and frame buffer coordinates, respectively, of a point. The objective function G to be minimised is -

$$G = \left[\tilde{x}_{fL} - x_{fL}(\boldsymbol{x})\right]^2 + \left[\tilde{y}_{fL} - y_{fL}(\boldsymbol{x})\right]^2 + \left[\tilde{x}_{fR} - x_{fR}(\boldsymbol{x})\right]^2 + \left[\tilde{y}_{fR} - y_{fR}(\boldsymbol{x})\right]^2$$

where the tilde indicates observed frame buffer coordinates. In linearising the system algorithms for the purposes of space intersection, the calibration parameters are known and therefore do not appear in the linearised form of the equations. Consequently, the system algorithms are linearised as -

$$j_{11L}dX + j_{12L}dY + j_{13L}dZ \approx x_{fL} - (x_{fL})_0 \tag{4-29}$$

$$j_{21L}dX + j_{22L}dY + j_{23L}dZ \approx y_{fL} - (y_{fL})_0 \tag{4-30}$$

$$j_{11R}dX + j_{12R}dY + j_{13R}dZ \approx x_{fR} - (x_{fR})_0 \tag{4-31}$$

$$j_{21R}dX + j_{22R}dY + j_{23R}dZ \approx y_{fR} - (y_{fR})_0 \tag{4-32}$$

Introducing the residual vector and expressing equations 4-29 to 4-32 in matrix notation -

$$\begin{bmatrix} j_{11L} & j_{12L} & j_{13L} \\ j_{21L} & j_{22L} & j_{23L} \\ j_{11R} & j_{12R} & j_{13R} \\ j_{21R} & j_{22R} & j_{23R} \end{bmatrix} \begin{bmatrix} dX \\ dY \\ dZ \end{bmatrix} = \begin{bmatrix} x_{fL} - (x_{fL})_0 \\ y_{fL} - (y_{fL})_0 \\ x_{fR} - (x_{fR})_0 \\ y_{fR} - (y_{fR})_0 \end{bmatrix} + \begin{bmatrix} r_{xL} \\ r_{yL} \\ r_{xR} \\ r_{yR} \end{bmatrix} \tag{4-33}$$

where j_{11} to j_{23} are the partial derivatives of the system algorithms with respect to the coordinates of spatial points, provided in Appendix III.

Since there are four equations and three unknowns, the problem is inherently overdetermined and the redundancy is used to obtain a better estimate for the object space coordinates of the unknown point. The least squares adjustment also solves the problem of non-intersecting rays, which could otherwise preclude the computation of a solution. It is apparent that, in the presence of stochastic noise, the greater the number of images available, the more accurate the least squares adjustment solution will be[106], up to a limit imposed by systematic errors. The minimisation of G is performed using the *singular value decomposition*, which according to Rothwell[107] is the most accurate projective reconstruction method.

4.5 Implementing the System Model

When iterative methods are used, convergence and numerical stability must be addressed. According to Atkinson[108],

"When automated processes largely replace direct human activity, automated methods of data screening must be part of the automated process; in other words, the quality of the data that are automatically generated must be automatically assessed".

The following section details the measures taken to enhance the convergence properties of the system model and the reliability of the produced data under real imaging conditions. The associated processes are implemented through the calibration and space intersection software code, which, as such, is an integral part of the line-scan system.

4.5.1 Calibration Implementation Issues

The least squares estimation (LSE) employed in the calibration and space intersection processes is a powerful technique. Mikhail states that[109] -

"This principle endeavours to ascertain that the new estimates are as close as possible to the sample values of the observations taking their stochastic properties also into consideration".

However, the LSE solution can only be as good as the underlying system model; LSE cannot compensate for an ill-conceived functional model, which will lead to divergence or, perhaps more significantly, to convergence into erroneous minima. Such an issue can be raised from the inclusion of lens distortion parameters to the system model (see Appendix II). If the effect of lens distortion is not apparent on the input data set, erroneous data will be produced for the lens distortion coefficients[89,110]. In addition, numerical instability is likely to occur[75]. Significantly, this will affect the integrity of the rest of the produced data[94]. In

this case, the erroneous fit is made possible by the existence of multiple local minima stemming from the suboptimal nature of the model.

Consider the incorporation of lens distortion parameters into the line-scan system functional model developed previously. The non-linearity of the lenses used in the experimental system has been investigated in previous work carried out by Godber[19] at The Nottingham Trent University. It was found that the 25mm lens suffers from barrel distortion which, in the worst case of fully opened aperture, amounts to 1% of the maximum image space radius. Godber's work also showed that the 50mm lens suffered approximately one-third the distortion of the 25mm lens.

In its most benign form, the radial non-linearity of the 25mm lens will be third order only. In contrast, if the amount of radial distortion is known for a given image space radius, the assumption that the non-linearity is produced *only* by a third order component is the most stringent. The maximum image space shift of 1%, recorded from the image centre to the bottom of the image, corresponds to a third order radial distortion coefficient k_1 of $0.225 \times 10^{-6}\,\mathrm{m}^{-2}$. This figure is arrived at from equation A2-2 of Appendix II by constraining the lens non-linearity to a third order component only ($k_2 = 0$). However, the cameras used in this work have a sensor length of 6.656mm, which is half of that used by Godber. Consequently, the corresponding maximum image space radius for the cameras used in this work is 3.328mm, giving a maximum image space distortion of approximately $8.1\mu m$, calculated from the value of k_1 arrived at above. This result is in accord with the lens distortion curve produced in Godber, the onset of radial distortion being identifiable only at radial distances greater than approximately 3.5mm. Moreover, the value of the lens distortion computed above represents the worst case, since the lens aperture will generally be decreased to allow for greater depth of field, thus improving lens linearity.

It can therefore be concluded that, for the cameras and lenses used in this research, lens distortion will affect the results only to a subpixel level. Consequently, the inclusion of lens distortion correction coefficients in the system model *must* be avoided. Of course, this would not be the case if cameras having smaller photosite aperture or subpixelation techniques were employed.

The issue of data quality is now addressed. The following tests are implemented during the calibration process:

i. dynamic (run-time) detection and removal of singularities;

ii. convergence detection using an adaptive criterion;

iii. divergence prevention.

After calculation of the elements of the Jacobian matrix, the product $\boldsymbol{J}^T\boldsymbol{J}$ is tested for singularities. The presence of singularities indicates that the problem is ill-conditioned[111] as one or more equations are linear combinations of another. If singularities are detected, the associated equations are removed at run-time. This is because singular data not only do not contribute to the least squares minimisation, but can also decrease the robustness of the solution.

The iterative process of residual minimisation is continuously monitored for divergence. This can occur if the control correspondence is incorrect and, if not monitored, it will lead to numerical overflow. Poor estimates for the initial approximations are also likely to result in divergence (or convergence towards the local minimum at infinity). In order to ascertain convergence, the following criterion is implemented -

$$\sqrt{\sum_{n=1}^{i}\left(\frac{dx_n}{x_n}\right)^2} < c \qquad (4\text{-}34)$$

where i is the number of calibration unknowns, x_n and dx_n denote the corresponding unknown and the correction to be applied to it after completion of the current iterative step. The constant c is initially assigned the value of 10^{-12}. If equation 4-34 is not satisfied after fifteen iterations, the convergence criterion is relaxed by increasing c to account for less robust correspondences. This is known as an *adaptive* convergence criterion. After convergence has been attained, the iterative process is terminated and the reliability of the produced solution is ascertained by:

i. verifying the orthonormality of the rotation matrices;

ii. displaying the variance of the calibration parameters;

iii. examining the norm of the residual vector.

The first test involves calculating the norm of the three rows of the rotation matrices to ensure that the three-dimensional camera coordinate systems are strictly rectangular. The precision of this test is limited only by truncation errors. All arithmetic computations and tests are performed in double floating-point precision, i.e. 64 bit.

Following the orthonormality test, the variance-covariance matrix of the solution is calculated and displayed. The variance of each calibration unknown is the squared uncertainty associated with its computed value. A large variance in a calibration parameter indicates a poor fit.

The norm of the residual vector is subsequently examined and if the model fit is poor, the user is informed. This can occur if the control correspondence is inaccurate due to, for instance, the presence of blunders. A more detailed review of the statistical testing of the least squares adjustment can be found in King[112].

4.5.2 The Requirement for Initial Approximations

As previously mentioned, the iterative process of linearising the system algorithms for both calibration and space intersection requires reasonable initial approximations for the unknowns. The task of producing these initial estimates is usually carried out manually and is generally a disadvantage of all iterative optimisation techniques. To this extent, work undertaken by numerous researchers in the field of machine vision has been aimed towards eliminating the need for user-supplied initial approximations. Weng[113] devised a method whereby a closed-form solution is obtained and is subsequently used as an initial approximation to the iterative optimisation. However, such methods have been developed for systems employing area array sensors where a closed-form solution is readily obtained if the camera optics are assumed to be linear. For the system developed in this work, the

degree of non-linearity inherent in the system algorithms impedes the implementation of such an approach.

To obtain approximate values for the calibration parameters of the line-scan system, nominal values for the lens focal length, the rotational speed and the integration period are used. Approximate values for the camera pose and orientation are obtained through observation or simple measurements performed on the particular system set-up. To this extent, providing successful initial approximations requires familiarisation with the operation of the line-scan system and the requirements of the calibration procedure. Tests carried out by the author indicate that, if a sufficient number of well-separated control points are used, the global minimum is well separated from local minima. In particular, initial estimates as far off the global minimum as 200% still lead to proper convergence. However, if the temporal relationship of the camera and object space coordinate systems at the start of scan cannot be estimated to this accuracy, it may be necessary to adopt a trial-and-error approach for the parameters concerned.

4.5.3 Space Intersection Implementation Issues

Initial approximations for space intersection are calculated in an entirely automated process by utilising geometric constraints. In particular, the estimation utilises the *midpoint* method where the mid-point of the vector of the minimum distance between the left and right camera rays is calculated. This is done in order to minimise user intervention and increase the overall reliability of the system. Hartley[105] states that -

"...an algorithm that attempts to minimise the cost function (...) by an iterative search beginning from an arbitrary initial point is in danger of finding a local minimum, even in the case of perfect point matches."

Consequently, the existence of a good starting estimate is imperative to the space intersection iterative minimisation (section 4.4.2, *phase four*). An approximate solution for the unknown point's radius r_i is obtained from Figure 4-9 as follows.

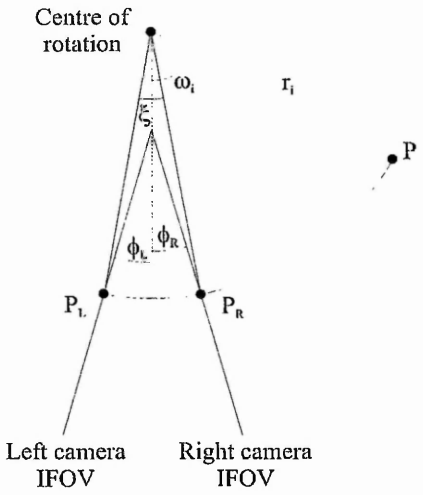

Figure 4-9 *Determining the unknown point's radius.*

From equation 3-8 of section 3.6.2 and Figure 4-9, angle ξ is -

$$\xi = s_{xL}^{-1} x_{fL} - s_{xR}^{-1} x_{fR} \tag{4-35}$$

Let -

$$\begin{aligned} \lambda &= |t_{xL}|, \quad \text{if} \quad |t_{xL}| > |t_{xR}| \\ \lambda &= |t_{xR}|, \quad \text{otherwise} \end{aligned} \tag{4-36}$$

The radius r_i is obtained by numerically minimising the expression -

$$\left| \xi - |\varphi_L| - \phi_R \right| - sin^{-1} \left| \frac{t_{xL}}{r_i} \right| - sin^{-1} \left| \frac{t_{xR}}{r_i} \right| \tag{4-37}$$

in the range -

$$\lambda < r < \mu \tag{4-38}$$

where λ is given by equation 4-36 and μ is any constant greater than the maximum expected radius.

An initial approximation for the angle ω_i of point P to the vertical axis, illustrated in Figure 4-10, is also required. This is calculated as follows.

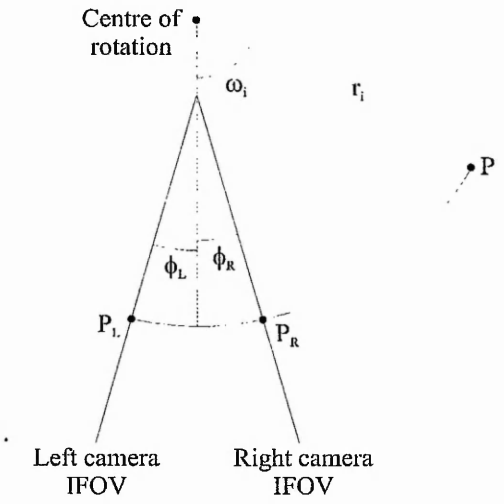

Figure 4-10 *Determining the unknown point's angle.*

From the left camera geometry, angle ω_i is -

$$\omega_{iL} = \frac{x_{fL}}{s_{xL}} + sin^{-1}\frac{t_{xL}}{r_i} + \frac{1}{2}\left(\varphi_L + \varphi_R - \left|\varphi_L - \varphi_R\right|\right) \tag{4-39}$$

Similarly, the right camera produces -

$$\omega_{iR} = \frac{x_{fR}}{s_{xR}} + sin^{-1}\frac{t_{xR}}{r_i} + \frac{1}{2}\left(\varphi_L + \varphi_R + \left|\varphi_L - \varphi_R\right|\right) \tag{4-40}$$

The initial approximation for the angle ω_i is obtained as the mean of these two angles. A cylindrical to Cartesian coordinate transformation is subsequently implemented as -

$$\begin{aligned} X_i &= r_i \sin\omega_i \\ Z_i &= r_i \cos\omega_i \end{aligned} \tag{4-41}$$

where X_i and Z_i are the X and Z axis Cartesian components of the initial approximation for the unknown point's spatial location.

An initial approximation for the Y axis component can be obtained by combining the Y axis algorithm (equation 4-12) and the solution of equation 4-37. The left camera produces -

$$Y_{Li} = \frac{1}{r_{22L}}\left[\frac{s_y}{f_L}\left(t_{zL} - \sqrt{r_i^2 - t_{xL}^2}\right)\left(y_{0L} - y_{fL}\right) + t_{yL} - r_{21L}X_i - r_{23L}Z_i\right] \qquad (4\text{-}42)$$

and the right camera -

$$Y_{Ri} = \frac{1}{r_{22R}}\left[\frac{s_y}{f_R}\left(t_{zR} - \sqrt{r_i^2 - t_{xR}^2}\right)\left(y_{0R} - y_{fR}\right) + t_{yR} - r_{21R}X_i - r_{23R}Z_i\right] \qquad (4\text{-}43)$$

where all symbols have their previously defined meaning. The Y axis initial approximation Y_i is obtained by taking the mean of Y_{Li} and Y_{Ri}.

4.5.4 Calculating the Spatial Quantisation Error

The procedure adopted to calculate the spatial quantisation error is as follows -

- given a point of interest in space, its left and right camera frame buffer coordinates are calculated by the system algorithms and the calibration data;
- successive small increments di_+ are applied to one spatial coordinate until any one of the four frame buffer coordinates changes by one pixel;
- successive small decrements di_- are applied to the same spatial coordinate until any one of the four frame buffer coordinates changes by one pixel;
- the spatial quantisation error is calculated as $di_+ + \left|di_-\right|$;
- the process is repeated for the remaining two spatial coordinates.

A flow chart of the algorithmic process is shown in Figure 4-11.

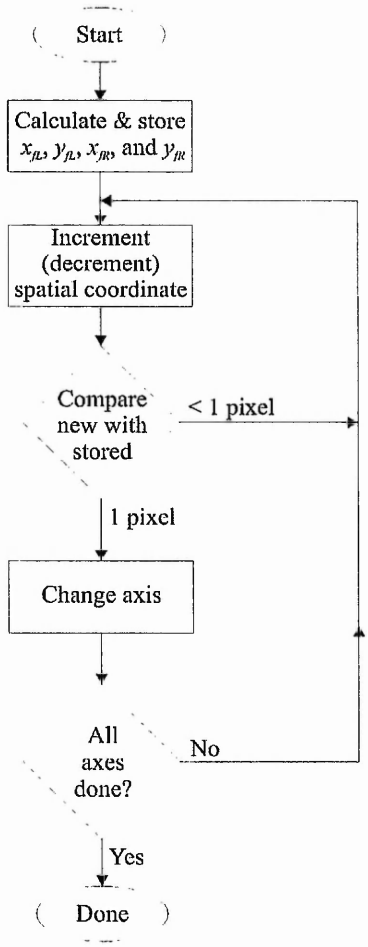

Figure 4-11 *Flow chart of spatial quantisation error calculation.*

Analogous procedures are adopted for the calculation of the *radial* and *angular uncertainties*, to be introduced in Chapter 5. The associated numerical processes are implemented via the calibration software code.

4.5.5 Software Code

The software code was implemented in Visual C++. In addition to the class hierarchy produced by Application Wizard, the following classes implement the calibration and the space intersection processes:

TABLE 4-2 *The calibration code hierarchy.*

Class:	Purpose:
CCalibration	Encapsulates the calibration unknowns, calculates the rotation matrices, loads control correspondence data files from permanent storage media, and performs data quality tests. This class is derived from MFC's (Microsoft Foundation Classes) *CCmdTarget* and forms the base class for the calibration code hierarchy.
CObjectSpace	Encapsulates the spatial coordinates of a point, calculates the system algorithms' partial derivatives and verifies the orthonormality of the rotation matrices. This class is derived from *CCalibration.*
CFrameBuffer	Encapsulates the frame buffer coordinates of a point, uses the system algorithms to calculate frame buffer coordinates from a set of calibration data and a control point, and constructs the vector of the functions to be minimised $\boldsymbol{F}$. This class is also derived from *CCalibration.*
CIntersection	Encapsulates the data structures used in the space intersection process and calculates the uncertainty components (see section 5.6). This class is derived from MFC's *CObject.*
COperations	LSE matrix algebra and other numerical routines (including the Singular Value Decomposition algorithm). *COperations* is a base class.

The source code for the above classes and their implementation is provided in Appendix IV. The *CDocument*-derived class which handles the main Newton-Raphson iterative loop is also provided in Appendix IV. However, the rest of the supporting classes of the MFC hierarchy used in the calibration code are not included.

4.6 Summary

The work presented in this chapter is summarised as follows:

- the stereoscopic region of the line-scan system has been defined, its governing parameters identified;
- the spatial sampling properties of the stereoscopic system have been analysed;
- a geometrical model with eight degrees of freedom has been established for each camera;
- a functional model stemming from the above algorithms has been derived;
- a full-scale calibration employing least squares optimisation has been presented;
- issues pertaining to the implementation of the above process have been discussed; constraints imposed by the experimental system were identified.

The following chapter presents the results of experiments undertaken with the stereoscopic line-scan system.

5. EXPERIMENTS WITH THE STEREOSCOPIC SYSTEM

5.1 Introduction

This chapter presents the experimental methodology and the results of a series of experiments undertaken to ascertain the metrological ability of the stereoscopic line-scan system. The experiments are divided in the following two categories:

I. *Imaging characteristics experiments*, devised to investigate the attributes of the stereoscopic line-scan system that produce stereoscopic parallax, thus allowing the extraction of three-dimensional coordinate information.

II. *Metrological experiments*, aiming to substantiate the mathematical model derived in the previous chapter and characterise the coordinate measurement accuracy of the stereoscopic system.

A number of considerations pertaining to all the experiments undertaken here and the resultant decisions are initially presented. In order to carry out the experiments, a special control field, consisting of a number of distributed targets, is utilised. The design and implementation of this structure is discussed.

5.2 Preliminary Considerations

5.2.1 Feature Matching

The extraction of spatial coordinate information from a set of perspective images requires that conjugate image points are identified. This is known as the correspondence problem[114]. Solving the correspondence problem automatically can be a complicated process[115] as conjugate image points not only appear spatially shifted between the two images, but generally also differ in brightness level due to the appearance of shadows or light

reflections. Feature occlusion, whereby parts of the object are hidden in one or more images due to the different imaging perspectives and shape of the object, may also be present.

A review of the methods developed to solve the correspondence problem is provided by Jones[116]. Techniques that have been developed include feature[117,118]- and area[119,120]-based matching. However, it is not the intention of this work to provide an automatic solution to the correspondence problem. Thus, points appearing in the images produced by the line-scan system are manually identified. As this task requires the interaction of the user and the images produced by the line-scan system, it was also decided not to employ an automated subpixel target location technique. The following section addresses this issue.

5.2.2 Subpixelation

In section 2.5.4 it is stated that if a subpixelation technique is not used, establishing the image location of a target has an uncertainty of ± 1 pixel. However, the least squares adjustment utilised in the calibration and the space intersection processes reduces the effect of this uncertainty. This is achieved by the stochastic modelling of these processes in conjunction with the redundancy present in the input data (see section 4.4.2).

An additional reason that would complicate the implementation of a subpixelation technique in this work stems from an idiosyncrasy of the experimental system: the odd and even video streams produced by each camera have unequal amplitudes. Although the video combiner circuit has separate gain adjustments for each video stream, temperature drift effects reduce the effectiveness of this adjustment and, hence, the radiometric quality of the images. Consequently, the accuracy of a subpixelation technique would be compromised by a systematic error, its magnitude being a function of the amount of gain difference between the odd and even video streams.

Finally, the metrological experiments conducted with the stereoscopic line-scan system show that the validity of the system model developed in the previous chapter can be established without resorting to a subpixelation technique.

5.2.3 Target Location

The size of the targets appearing in the two perspective images depends on the spatial dimensions of the target and the operating parameters of the line-scan system that control the dimensions of the spatial sample (see section 4.3). Hence, the location of a target in the images produced by the line-scan system and the accuracy of an automated or a manual target location measurement technique, are affected by -

Geometric factors:

i. the relative size and position of the target to the spatial sample;

ii. the shape of the target;

iii. the obliquity of view, giving rise to perspective distortion[121];

iv. the effects of spatial oversampling and undersampling;

Radiometric factors:

i. the intensity profile of the imaged targets, determined by the light reflection properties of the target's surface, the target illumination and the sensitivity of the line-scan system;

ii. any difference in the sensitivity of the odd and even video lines;

iii. radiometric artefacts, such as blooming.

Thus the location, size and shape of a target appearing in an image is affected by numerous parameters and its accurate identification can be an involved process. In related work carried out by Singh on this topic, it is stated that[122] -

"Although the target points produced by the laser projector look similar to each other, different values of thresholding give different shapes for each target...So the centroid

calculation of the area detected by the thresholding technique may not give the correct co-ordinate value".

A substantial amount of research work has been carried out in the area of feature detection and location, and various techniques are now available, e.g.[123,124,125]. However, since in this work the targets are located manually, the pixel closest to the author's interpretation of the centroid of the target is used. This is further discussed in section 6.6.

5.2.4 Nominal System Alignment

The nominal alignment of the stereoscopic system is depicted in Figure 5-1. To obtain an experimental set-up closely conforming to these conditions is not a trivial task. This is because, not only must both camera convergence angles φ be precisely equal, but the rotation axis must intersect the line normal to half the camera basewidth B, as shown in Figure 5-1.

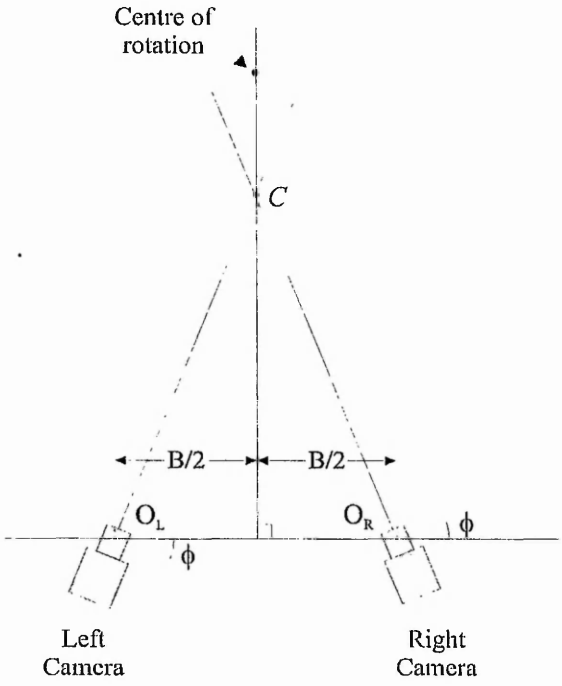

Figure 5-1 *Nominal stereoscopic system alignment.*

In addition, the imaging characteristics of the line-scan system (see section 5.6) are substantially affected by the calibration parameters t_{xL} and t_{xR}, that is, the distance of the IFOVs from the rotation axis. As for the experimental set-ups considered in this work these distances are of the order of a centimetre, the location of the test field in the axis parallel to the camera basewidth has to be controlled, ideally, to millimetric accuracy.

An area-array camera produces images that are interpreted by a human as real-time. Conversely, considering that the experimental line-scan system stores 3048 scan-lines in each frame, when a long integration period is used the production of a line-scan image can take up to one minute. Moreover, in their existing configuration the frame grabbers used do not permit the displaying of the line-scan image concurrently to its production. Instead, a complete frame has to be grabbed and then copied to VGA memory. As a result, the setting up the line-scan system is a time-consuming process.

Although an experimental set-up closely conforming to the nominal alignment is not required by the system model, it is of critical importance to the identification of the imaging characteristics of the system. This is because such an alignment, in conjunction with a suitable object such as the cylindrical control field described in the next section, will allow the intuitive interpretation of the images and, hence, the imaging characteristics of the line-scan system.

5.3 Control Fields

A three-dimensional *control* or *test field* consists of a number of distributed targets that have known spatial locations in a local coordinate system. In general, a control field can be used for:

i. camera calibration;

ii. validation of the mathematical model by metrological experiments;

The targets on the control field can be divided into two categories, according to the purpose they serve:

- *control* points;
- *test* points.

Control points are used to calibrate the system, whereas test points are used to evaluate the metrological accuracy of the system.

In addition to the above general requirements, in this work the control field must also cater for:

i. the setting up of the stereoscopic line-scan system;

ii. the identification of the stereoscopic system's imaging characteristics.

Furthermore, for space intersection to be applicable, a global object space reference coordinate system, mutual to both cameras, must be established. This is achieved by calibrating both cameras simultaneously. It can be appreciated that this process is similar to photogrammetric relative orientation of a stereo-pair of photographs.

The difficulties associated with setting up the line-scan system were identified previously. Accordingly, it is important that the images produced by the control field are intuitive and easily interpretable.

5.3.1 The Cartesian Control Field

Initial work involved the utilisation of an existing control field, depicted in Figure 5-2. This consists of a distribution of targets comprising black-anodised aluminium rods having conically machined tips painted white.

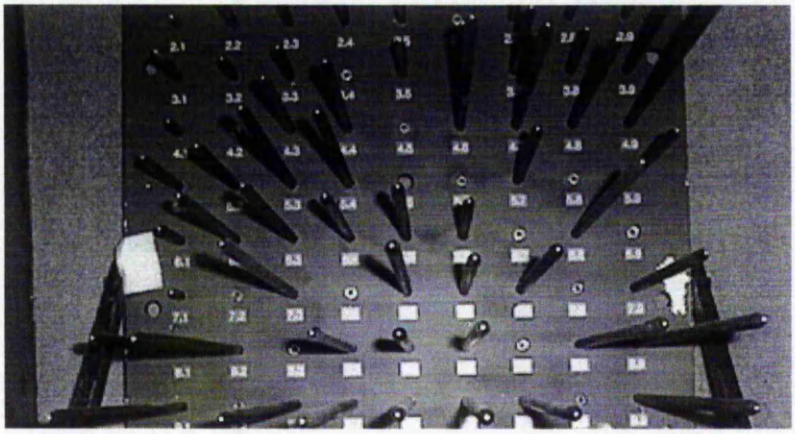

Figure 5-2 *The Cartesian control field.*

However, it became apparent that this structure was far from ideal when used with the line-scan system. The reasons for this are:

i. the shape of the control field is governed by a Cartesian format;

ii. the complete structure weighs 21 kilograms, thus restricting the maximum achievable rotational speed to 1.5 rpm;

iii. the protruding rods are imaged off-axis (in oblique views), producing images that are difficult to interpret;

iv. no indication of the position of the rotation axis is available in the images as no cylindrical symmetry exists;

v. the targets are concentrated in an area covering a maximum angular field of view of less than 180^0..

Thus, this test field was deemed unsuitable and a new structure was designed, specifically with the needs of the line-scan system in mind.

5.3.2 The Cylindrical Control Field

In order to be suitable for the line-scan system, a test field should fulfil certain criteria. These can be divided into three categories:

I. *Targets:* the targets should -

i. provide sufficient contrast and, preferably, also be highly structured so that their identification by the system operator is unambiguous;

ii. have the smallest size possible, consistent with the above requirement;

iii. protrude as little as possible from the main body of the control field in order to avoid the sides of the targets appearing in the images.

II. *Image production:* the control field should be designed such that -

i. the images are intuitive and easily interpretable;

ii. it facilitates the setting up of the line-scan system in accordance with the arguments detailed in section 5.2.4.

III. *Calibration:* the control field should have -

i. the coordinates of the targets specified to an accuracy of at least one order of magnitude higher than the expected accuracy of the line-scan system;

ii. the coordinates of the targets specified in a local coordinate system, referenced to the rotation axis;

iii. a number of non-collinear and non-coplanar targets.

In addition to these requirements, the control field should be removable to allow for imaging of other objects.

From the above set of requirements, it can be appreciated that not all criteria can be satisfied with one structure. Most notably, the requirement of non-collinear targets conflicts with the setting up of the line-scan system, the intuitive interpretation of the images and, as discussed in section 5.6, the imaging characteristics experiments. Furthermore, manufacturing constraints and limitations in the coordinate measurement procedure of the targets necessitated the adoption of a 'segmented' cylindrical structure of progressively decreasing diameter. The resulting control field is depicted in Figures 5-3 and 5-4.

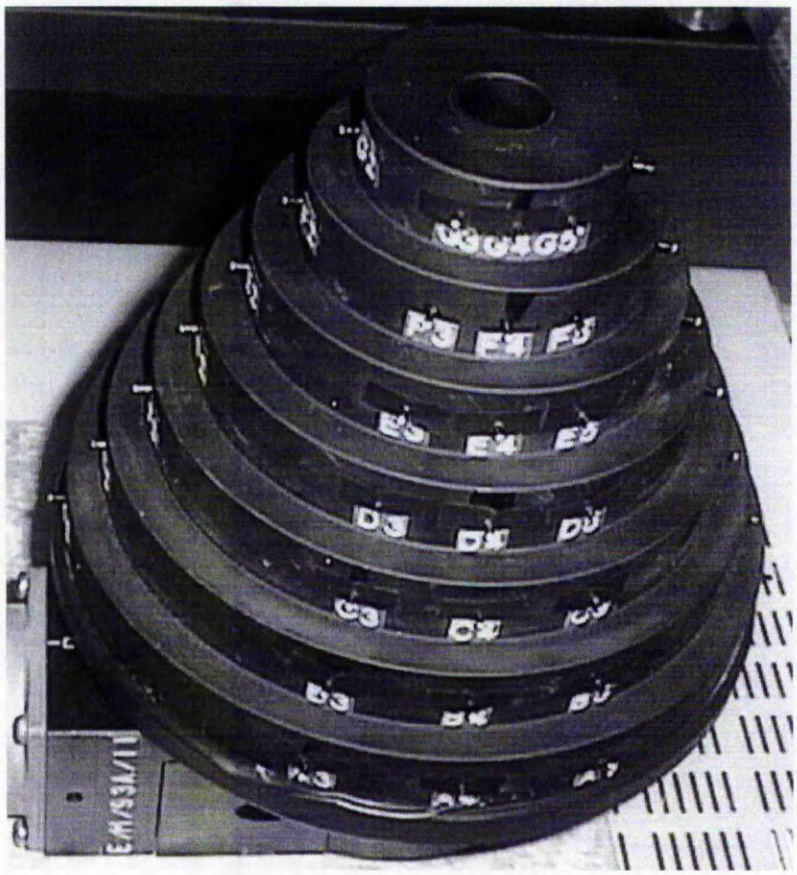

Figure 5-3 *Side view of the cylindrical control field.*

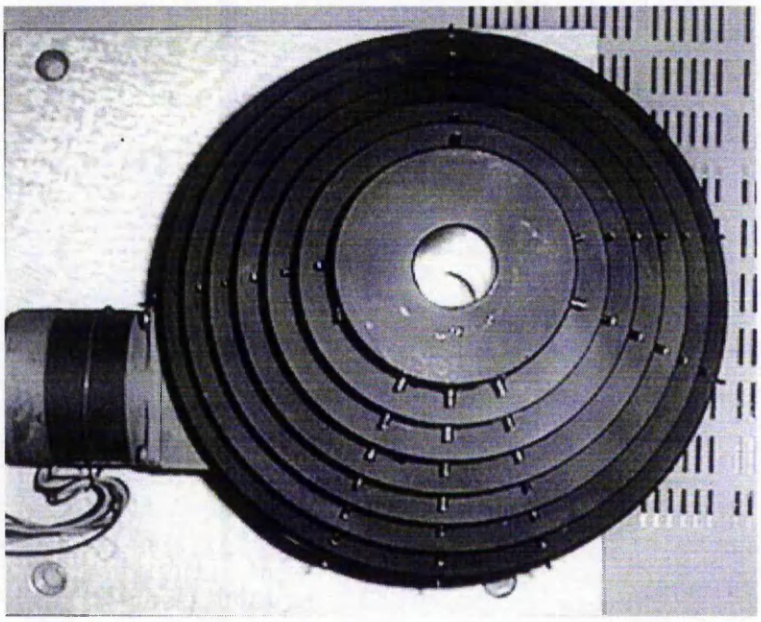

Figure 5-4 *Plan view of the cylindrical control field.*

This structure was machined at the Mechanical Engineering Department of The Nottingham Trent University to the specification described below. The height of the test field is approximately 140mm and its minimum and maximum diameters are 70 and 188mm, respectively. Seven 'slices' are used and the diameter of the central 'slice' is calculated to produce an approximately 1:1 pixel aspect ratio at a rotational speed of 2rpm and an integration period of 8mS (see section 3.6.4).

The whole structure is lightweight enough to be rotated at up to six revolutions per minute using the existing hardware. In addition, the control field has a recessed base that fits precisely to a specially constructed metal plate mounted rigidly on the rotary stage. This way, the test field can be removed, for instance after completion of a system calibration, to facilitate imaging of other objects.

Forty-nine targets, arranged collinearly in groups of seven, are distributed around the periphery of the control field, as shown in Figure 5-5. Each set of collinear targets is assigned a number from 1 to 7 and each set of concentric targets a letter from A to G. In order to provide a sufficient number of targets for angular fields of view smaller than 360^0, the distribution of the collinear sets of targets is angularly asymmetrical. The targets consist of black anodised aluminium rods having matt white tips to avoid specular light reflections.

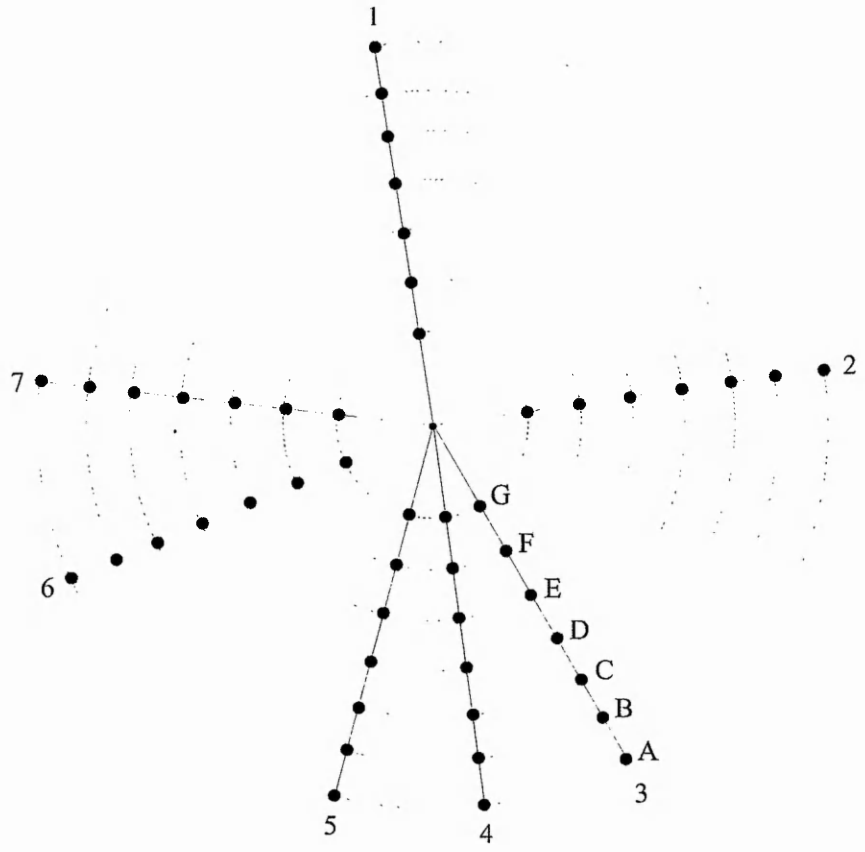

Figure 5-5 *Plan view of target distribution in the cylindrical control field.*

The manufacturing of the control field imposed a number of limitations. These are summarised as follows:

i. the minimum diameter of the targets required by the coordinate measurement machine was 1.5mm. This is larger than the spatial sample in most operating conditions, thus the image of a target occupies more than one pixel. This can make the manual identification of their location less intuitive and subject to larger errors;

ii. a finite target length extending from the main body of the control field was required by the coordinate measurement machine. This can make the identification of the centroid of the targets more difficult as part of the targets' sides appear next to the targets' tip in the image;

iii. the location of the targets could only be specified relative to the axis of cylindrical symmetry of the control field instead of the rotation axis. Therefore, any

misalignment between the two axes produces systematic errors in the coordinates of the targets;

iv. the location of the targets could only be specified to an accuracy of $\pm 75\mu$m.

Despite these limitations, the cylindrical control field is effective both for the purposes of setting up the line-scan system and for calibration, as the metrological experiments indicate.

5.4 Test Area Illumination

The test area, consisting of the components of the line-scan system described in section 3.2, is illustrated in Figure 5-6. In addition to the two line-scan cameras, the host controller and the camera basewidth/convergence stage, a linear translation table carrying the test field or the object to be imaged and two Halogen flood lights are used.

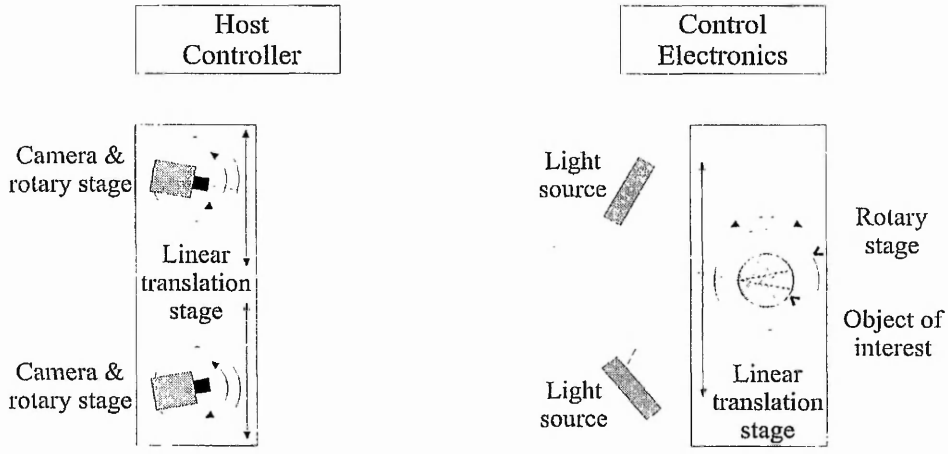

Figure 5-6 *The test area.*

The linear translation table supporting the control field can be precisely adjusted along the direction parallel to the camera basewidth, thus simplifying setting up the system. The two lamps are mounted on stands of adjustable height and they are controlled by individual dimmer units. This allows the setting of the illumination level according to the operating parameters of the line-scan system in order to produce images having sufficient brightness and contrast.

As previously mentioned, the radiometric properties of the line-scan system are not studied in this work. As long as the features of the object to be imaged have high contrast and the location of the targets can be the identified to an accuracy of ± 1 pixel, the radiometric properties of the line-scan system and details of the illumination arrangement are not of critical importance. However, this may not be the case in a real application where both lighting and the system's radiometric properties may have to be optimised.

5.5 Coordinate Measurement: An Example

The procedure of extracting coordinate information from an object workspace begins with the initialisation of the line-scan system, as described in section 3.3. The system parameters set by the operator are:

i. the camera basewidth;

ii. the camera convergence angles;

iii. the camera-to-object range;

iv. the workspace illumination.

After these preliminary settings are carried out and prior to image capture, the following parameters are also set:

i. the integration period;

ii. the object rotation arc;

iii. the object rotation speed.

The start of scan, i.e. the instant image capture is initiated, is controlled automatically so that synchronisation with object rotation is maintained. However, the relevant automatic process can be manually overridden by the operator, if so required.

After adjustment of the lenses' focus, the acquired left and right images are saved on to disk. The frame buffer coordinates of control points are measured via an image-processing package and the control correspondence is established. Coordinate data for this correspondence is edited into a text file. Subsequently, the calibration code is invoked. The control correspondence data are loaded at run-time, and a dialog box allows editing of the initial approximations for the sixteen calibration unknowns.

On completion of the calibration process, the calibration results and the variance of the fitted parameters are displayed on dialog boxes similar to those of Figures 5-7 and 5-8, respectively. Additional information on the calibration results is reported as per Figure 5-9.

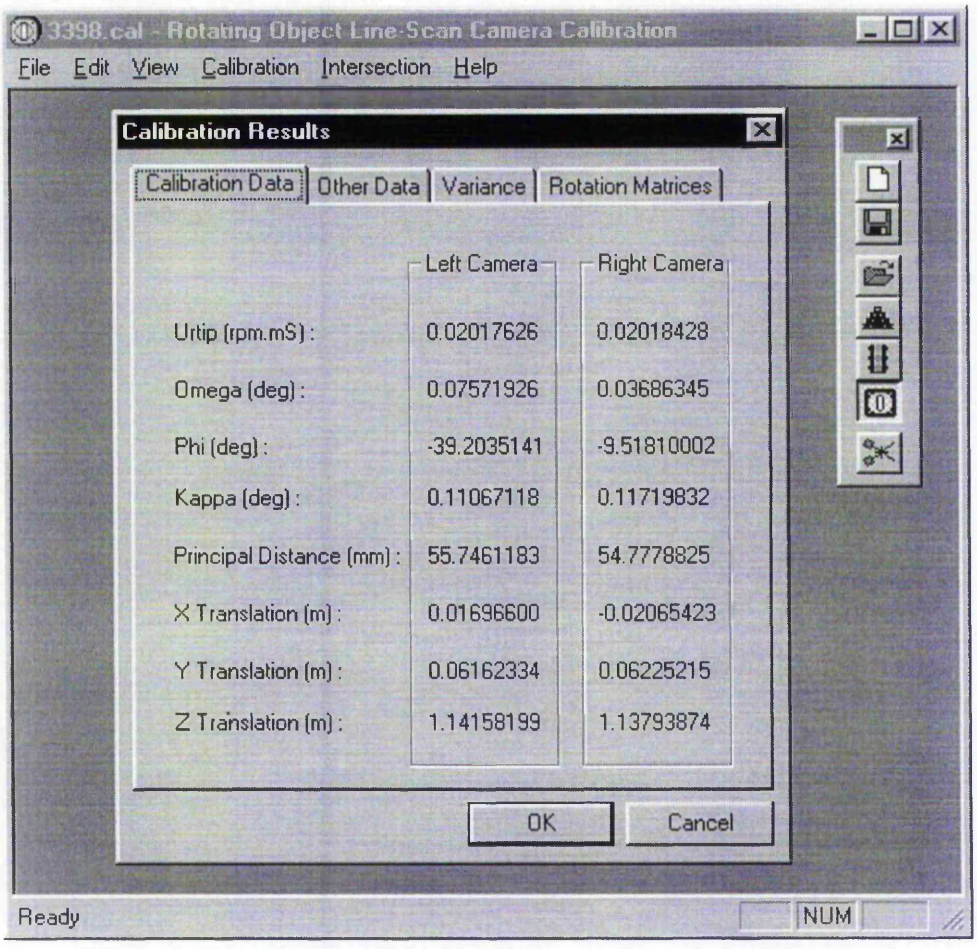

Figure 5-7 *The calibration code showing sample calibration results.*

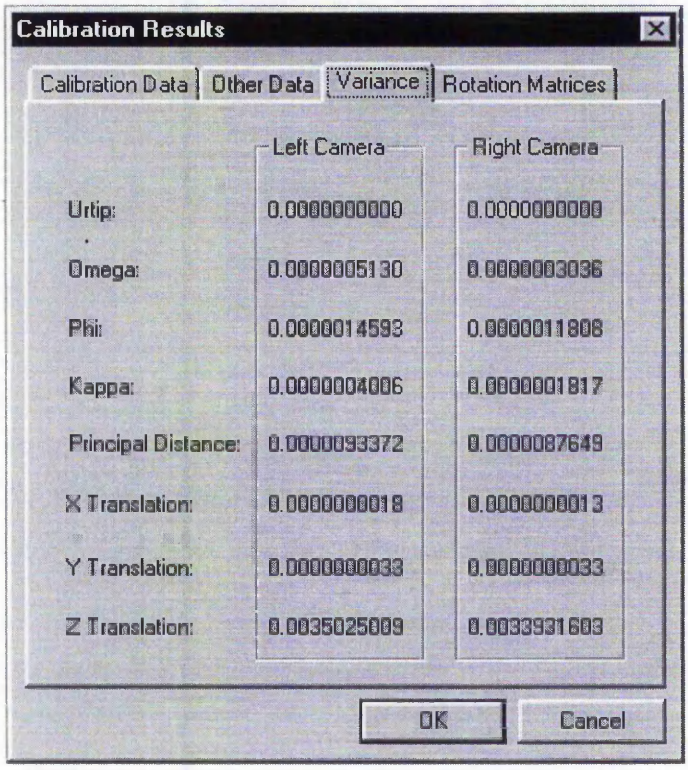

Figure 5-8 *Variance of the calibration parameters.*

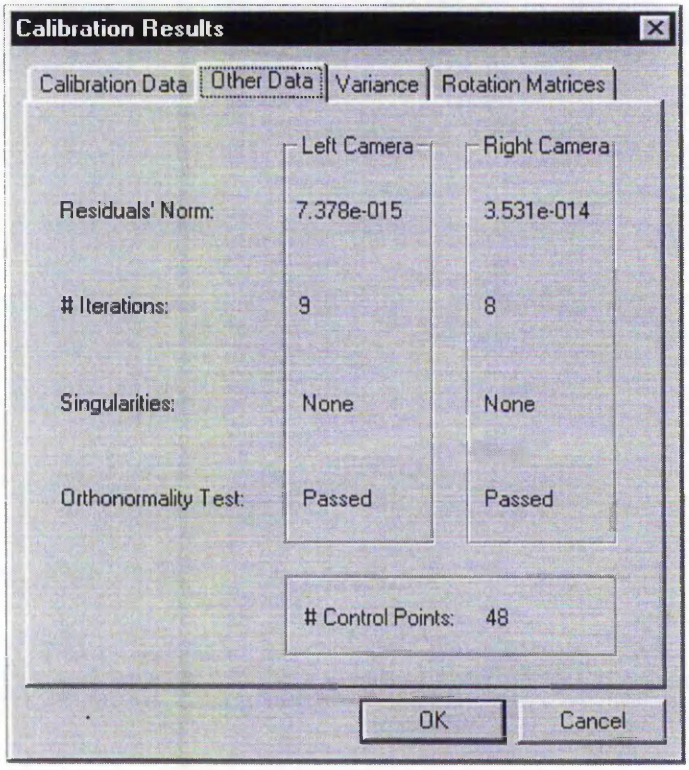

Figure 5-9 *Other calibration data.*

The calculated calibration data can be subsequently used to perform space intersection. The image correspondence, consisting of the left and right frame buffer coordinates of the unknown spatial point, is also required. The results of the iterative least squares space intersection are reported as shown in Figure 5-10.

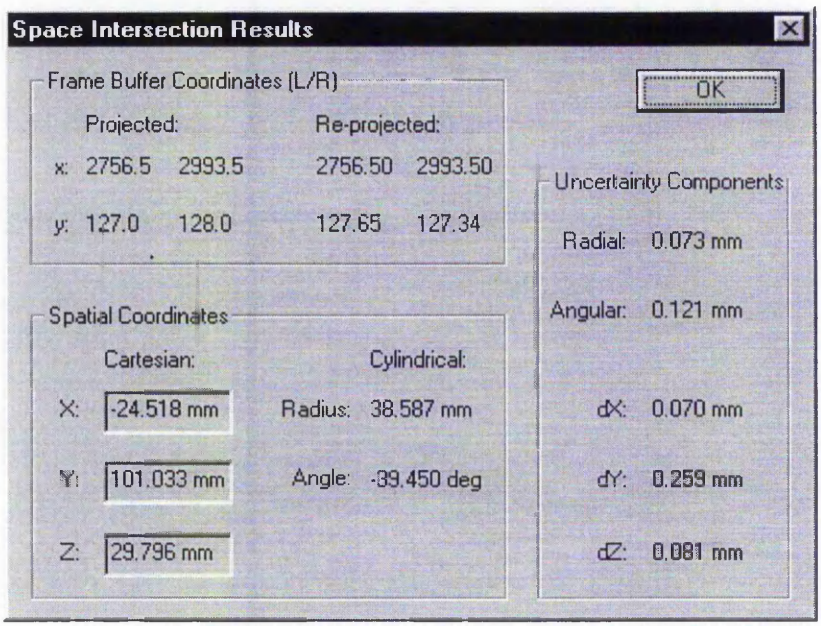

Figure 5-10 *Space intersection results.*

Additional information consists of the spatial quantisation noise calculated at the current solution and the cylindrical coordinates of the spatial point. The calculated spatial solution is also re-projected and the corresponding frame buffer coordinates are reported. These can be compared with the original projected coordinates to provide an indication of how accurately the data fitted the model. Section 5.7.3 addresses such issues in more detail.

5.6 Imaging Characteristics

A series of experiments were conducted to characterise the imaging characteristics of the stereoscopic line-scan system. These experiments are based on the observation that the information produced by the system is conveyed exclusively from the two perspective images. Therefore, it should be possible to draw conclusions pertaining to the imaging characteristics of the line-scan system by an intuitive analysis of the 'raw' digital image data.

Figure 5-11 depicts left and right images of the cylindrical control field. The system set-up used to produce these images is such that the IFOVs of both cameras converge on the rotation axis.

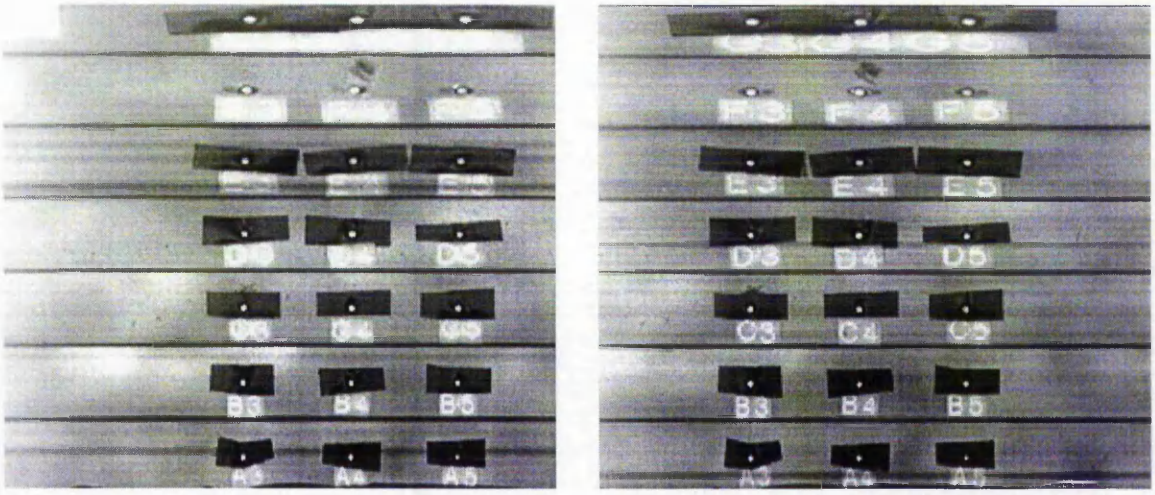

Figure 5-11 *Left and right camera images having no stereoscopic parallax.*

It can be seen that, although there is a constant lateral shift of corresponding targets between the two images, no stereoscopic parallax exists. In other words, under these operating conditions, the left and right views are essentially the same. The two images are differentiated only by the time-delay associated with 'capturing' a target by each camera, which is realised as a lateral shift. Evidently, this time-delay is independent of the radial distance of the targets to the rotation axis.

Figure 5-12 *Left and right camera images depicting stereoscopic parallax.*

If the line-scan system is arranged such that the IFOVs converge at a finite distance from the rotation axis, as illustrated in Figure 5-1, stereoscopic parallax is produced. This can be seen from the left and right images depicted in Figure 5-12. Recalling the X axis algorithm of the functional model, stereoscopic parallax is created in this case as the image location of the targets is a function of their radial displacement. Therefore, the condition for stereoscopic parallax to exist is that the IFOVs of the line-scan cameras converge at a finite distance from the rotation axis.

As shown in the following series of experiments, further conclusions pertaining to the imaging characteristics of the line-scan system can be drawn from a consideration of the raw digital image data produced by imaging the cylindrical control field. A typical system set-up is obtained by setting the line-scan system's operating parameters to the following values:

Integration period: 10mS

Rotational speed: 2rpm

Convergence angles: 10^0

Nominal lenses' principal distance: 50mm

Absolute distance of the IFOVs from the rotation axis: 15mm

Range: 1.5m

The integration period and the rotational speed are set such that an angular field of view of 360 degrees is obtained. The perspective centre to rotation axis distance setting produces a Y axis field of view slightly larger that the height of the cylindrical control field. Finally, the camera convergence angles are set such that the radius of the volume of occlusion is slightly smaller that the minimum radius of the control field.

The following experiments utilise the calibration data for the cylindrical control field, which are provided in Table A, Appendix V. The first experiment utilises the cylindrical control field in order to plot the X image axis coordinates as a function of the radial displacement from the rotation axis.

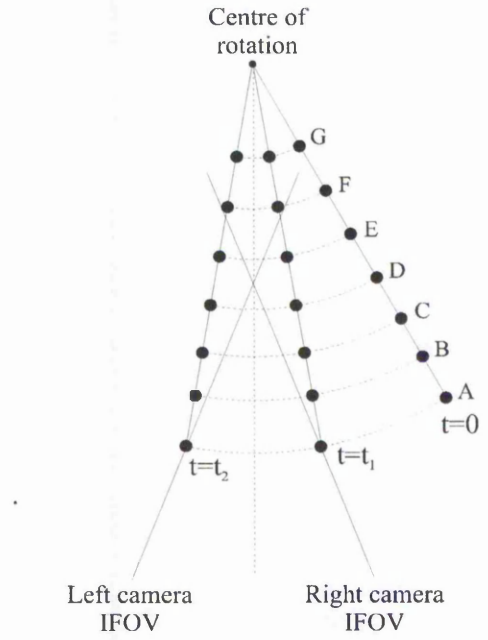

Figure 5-13 *A set of collinear targets imaged by the line-scan system.*

Seven collinear targets A, B, C, D, E, F and G are coplanar with the rotation axis at time t=0, as shown in Figure 5-13. This set of targets is subjected to clockwise rotation and is imaged first by the right and then by the left camera. From Figure 5-13 it can be seen that the right camera 'sees' target A first at time $t=t_1$, whereas the same target is seen last by the left camera at time $t=t_2$. This effect is apparent also in Figure 5-14, which shows part of the left and right perspective images superimposed.

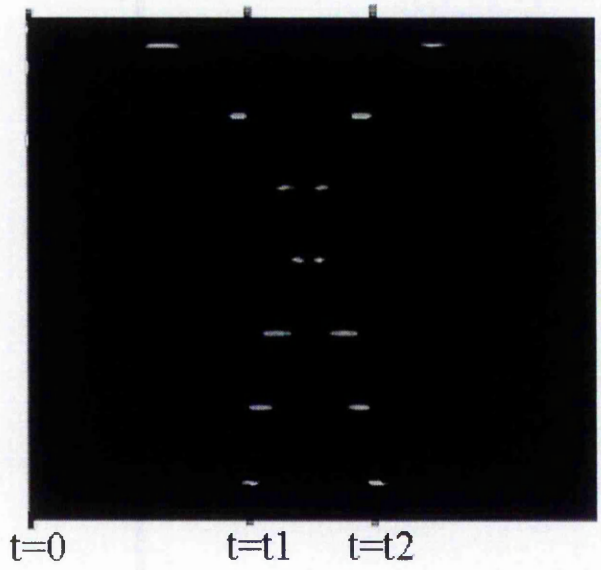

Figure 5-14 *Superimposed left and right perspective images.*

As the same target appears in reverse order in the left and right images, lines produced by the collinear set of targets have opposite gradients. This can be appreciated from the graph of Figure 5-15, which plots the X axis frame buffer coordinates of the collinear set of targets as a function of their radial displacement.

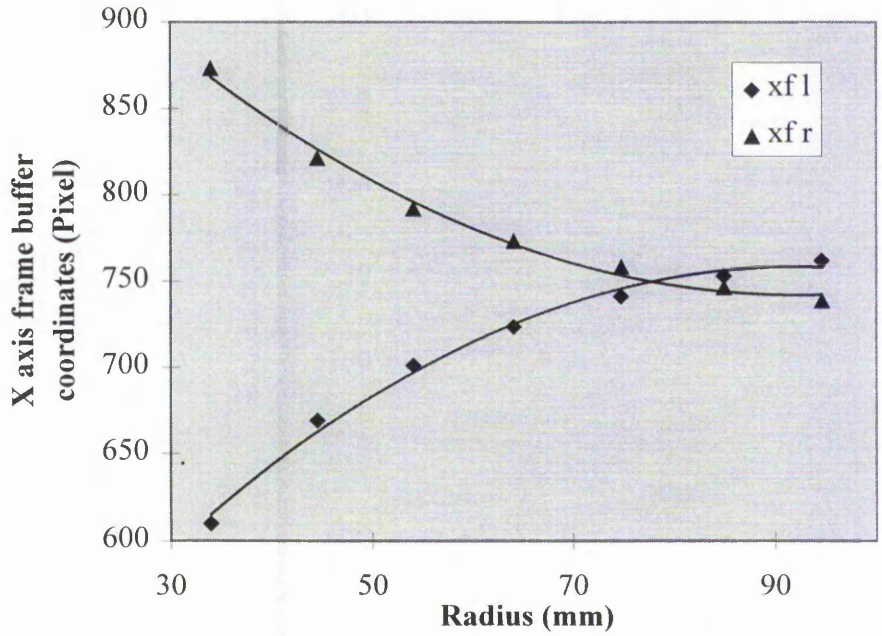

Figure 5-15 *Graph of left and right image X axis frame buffer coordinates against radius for a single set of collinear points.*

The graph of the X axis frame buffer coordinates of the seven sets of collinear targets present on the cylindrical control field as a function of radius is shown in Figure 5-16. A polynomial line fit is added for each collinear set of targets.

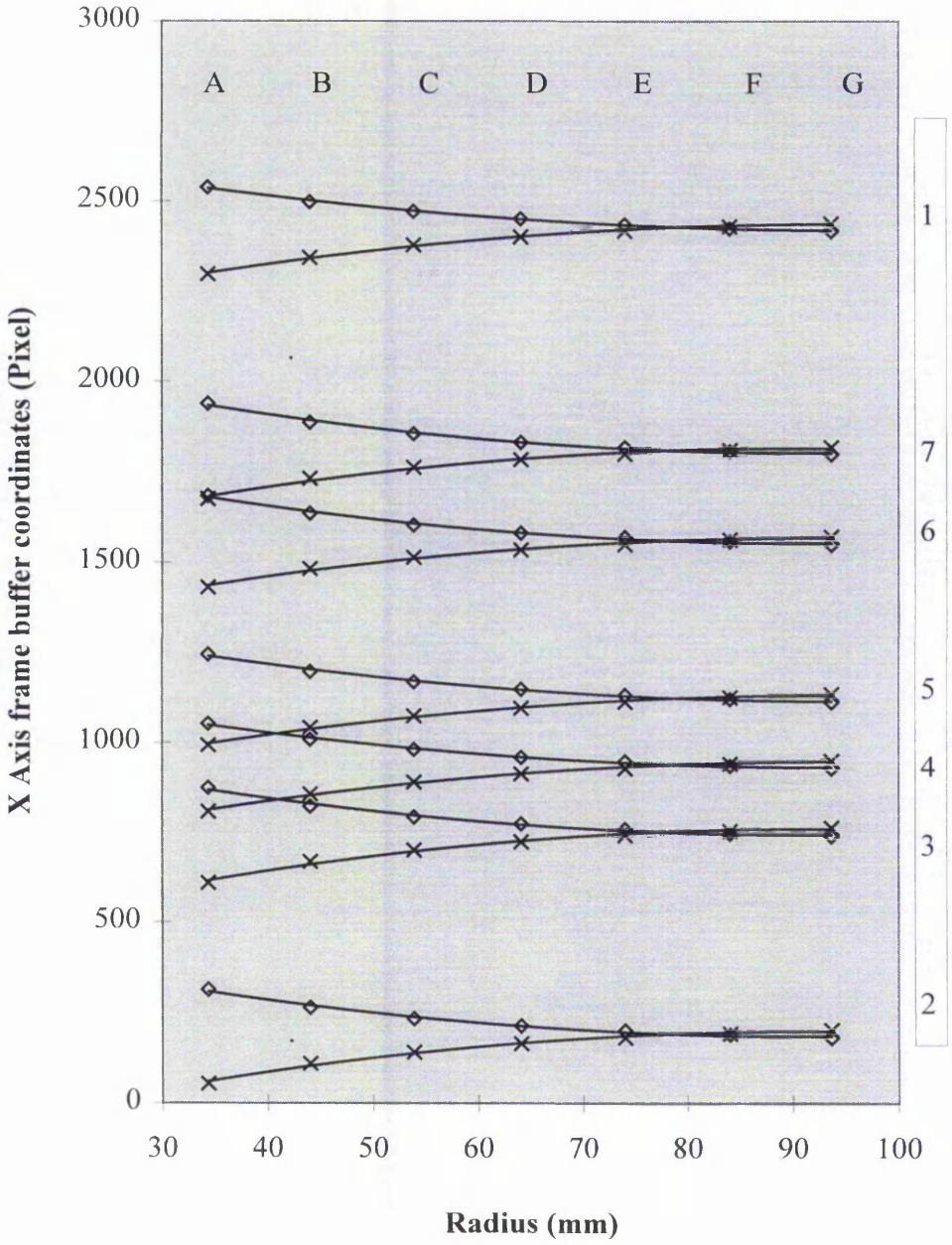

Figure 5-16 *X axis frame buffer coordinates against radius.*

As indicated by the legends, each symmetrical set of adjacent lines corresponds to the X axis frame buffer coordinates of a collinear set of targets (which are also coplanar with the rotation axis), produced by the left and right cameras. Only a lateral shift in the X image axis differentiates the pairs of lines corresponding to each set of targets, as the X axis coordinates are a linear function of angle. However, each line is non-linear, as the X axis coordinates are a non-linear function of the radial displacement.

The radial distance from the rotation axis at which the IFOVs converge can also be estimated from the left and right perspective images and the corresponding graphs depicted in Figures 5-15 and 5-16. At this distance the parallax is zero. This range corresponds approximately to the radius of "slice *E*", i.e. 44mm from the rotation axis.

The graph of the X axis frame buffer coordinates of control points as a function of angle is shown in Figure 5-17. This graph is produced from the concentric set of targets *G*. A polynomial line is fitted.

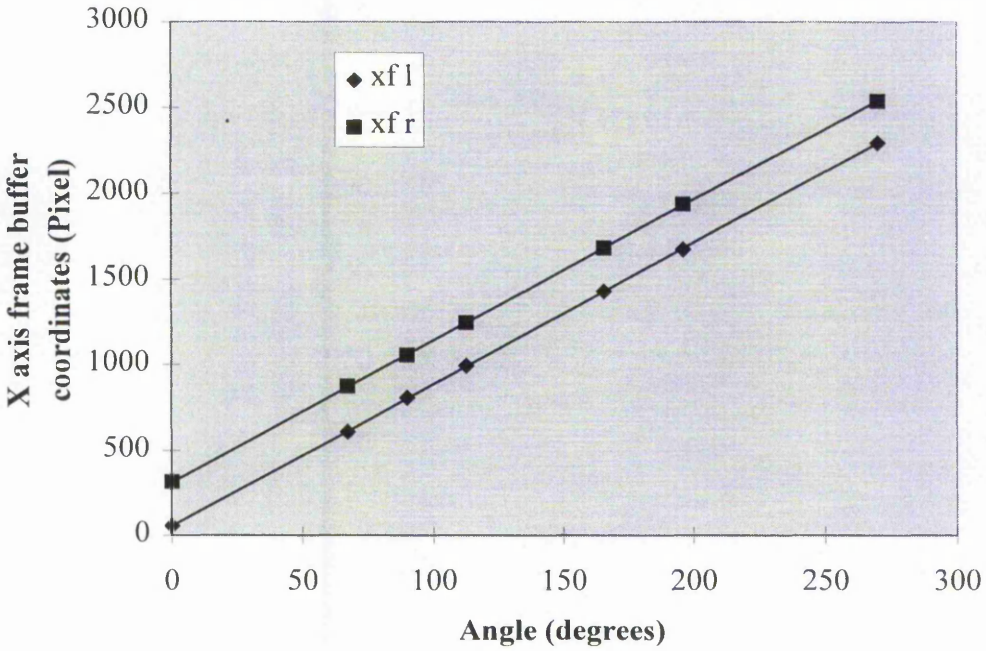

Figure 5-17 *X axis frame buffer coordinates against angle.*

As expected from both the two-dimensional analysis presented in Chapter 3 and the stereoscopic system's model, X axis frame buffer coordinates are linear in angle. The parallax Δx_f for the seven concentric sets of targets as a function of angle is depicted in Figure 5-18.

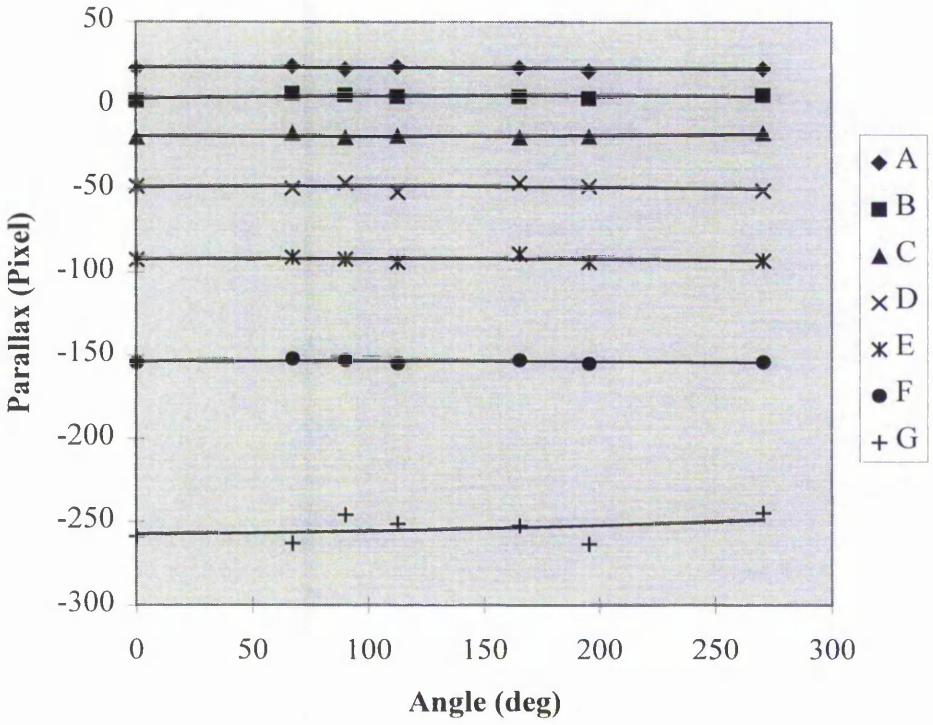

Figure 5-18 *Graph of parallax* Δx_f *against angle.*

From this figure it is apparent that the concentric set of targets G1 to G7, which has the smallest radius, produces the largest disparity. This effect can be appreciated from the diagram of Figure 5-13. It is also noted that the parallax is independent of angle. This is attributed to the X axis system algorithm: the left and right camera frame buffer coordinates are linear in angle and are both scaled by the horizontal scale factor s_x.

The graph of the Y axis frame buffer coordinates of control points as a function of Y axis spatial distance is shown in Figure 5-19. A polynomial line is fitted to the left and right camera frame buffer coordinates.

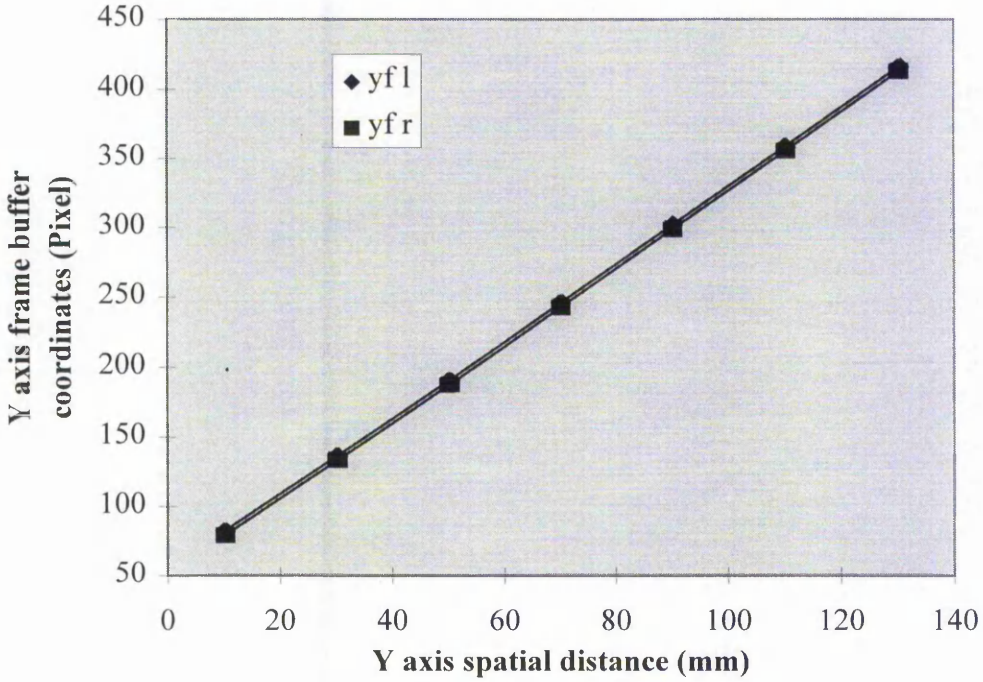

Figure 5-19 *Y axis frame buffer coordinates against Y axis spatial distance.*

Since the projection in the line-scan sensor's main axis is perspective and each set of concentric targets lies at a different range from the perspective centre, the graphs are slightly non-linear. However, the maximum radius of the targets is approximately an order of magnitude smaller that the perspective centre to rotation axis range. Thus, the variation in range is minimal and, in the above graphs, the Y axis frame buffer coordinates appear approximately linear in Y axis spatial distances.

THE UNCERTAINTY COMPONENTS

The spatial sampling pattern of the line-scan system is presented in section 4-3. It is noted that the spatial resolution of the line-scan system is limited by the dimensions of the voxels. From Figures 5-20 and 5-21 it can be seen that the dimensions of the voxels are characterised by a radial, an angular and a Y axis component. The latter is defined in the direction parallel to the sensor's main axis.

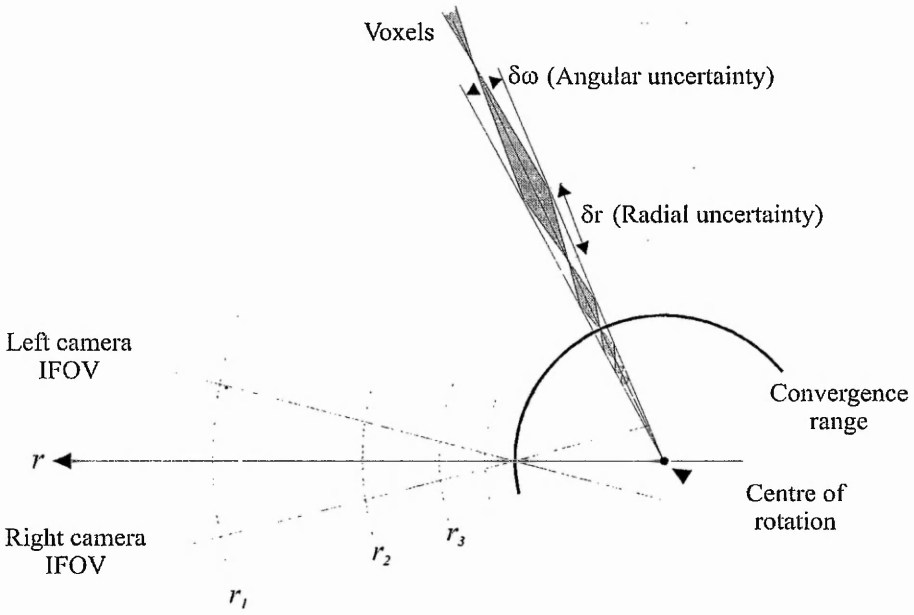

Figure 5-20 *Radial and angular uncertainty.*

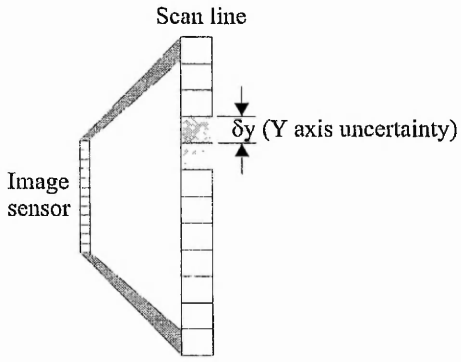

Figure 5-21 *Y axis uncertainty.*

The location of a target appearing anywhere within the volume of a voxel cannot be determined to an accuracy better than the spatial dimensions of that voxel. Thus, the terms *radial uncertainty*, *angular uncertainty* and *Y axis uncertainty* are adopted to represent the corresponding spatial quantisation noise component.

Estimating the Radial Uncertainty:

Figure 5-22 plots the parallax Δx_f of a set of collinear points as a function of the radius.

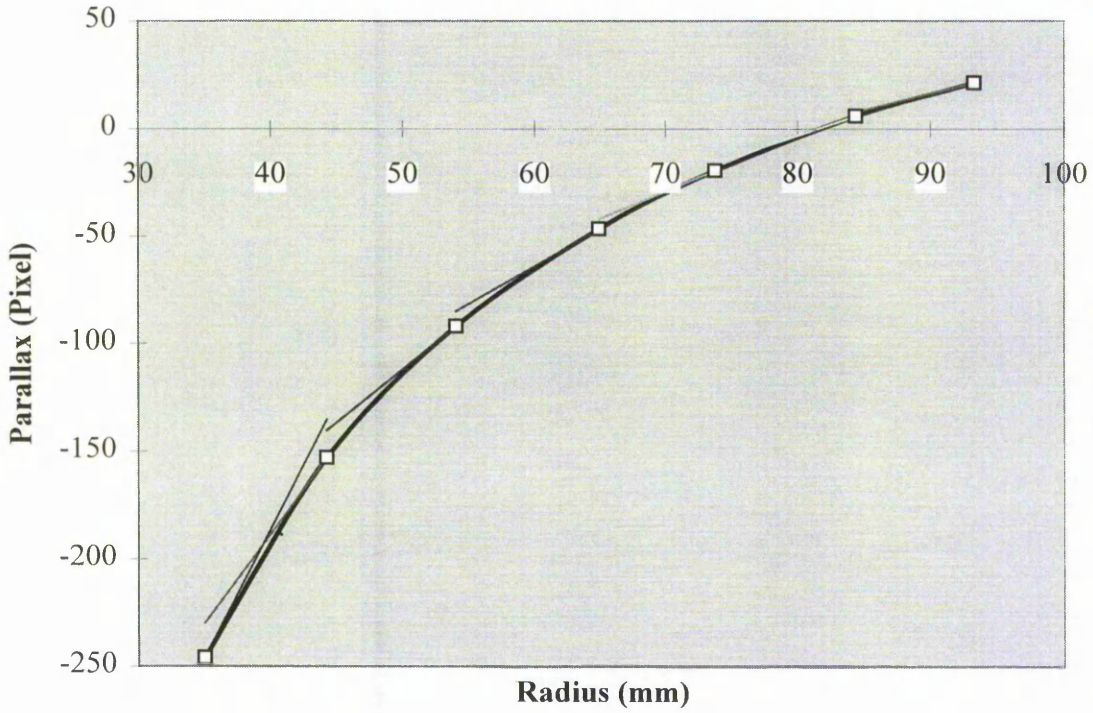

Figure 5-22 *Graph of parallax Δx_f against radius with fitted tangents.*

The radial uncertainty δr at a given radius is given by the gradient of the parallax graph. To estimate this from Figure 5-22, seven straight lines, approximately tangential to the parallax curve at different radii, are fitted. The radial uncertainty estimated from these tangents is shown in the fourth column of Table 5-3.

TABLE 5-3 *Estimated and calculated radial uncertainties.*

Target	r (mm)	Δx_f	Estimated radial uncertainty δr_e (mm/pixel)	Calculated radial uncertainty δr_c (mm/pixel)
A4	93.286	21.5	0.661	0.601
B4	84.369	6	0.482	0.496
C4	73.769	-19.5	0.394	0.374
D4	64.913	-46.5	0.282	0.288
E4	54.058	-92	0.204	0.197
F4	44.243	-153	0.120	0.130
G4	34.998	-246	0.090	0.078

In order to compare the results of this approach with those obtained analytically, the image information utilised in the above experiment is used to calibrate the system. Following this, space intersection is performed and the radial uncertainty calculated by the method described in section 4.5.4 is recorded for each target. The results from this method are shown in the last column of Table 5-3. The graph of the estimated and calculated radial uncertainties is shown in Figure 5-23.

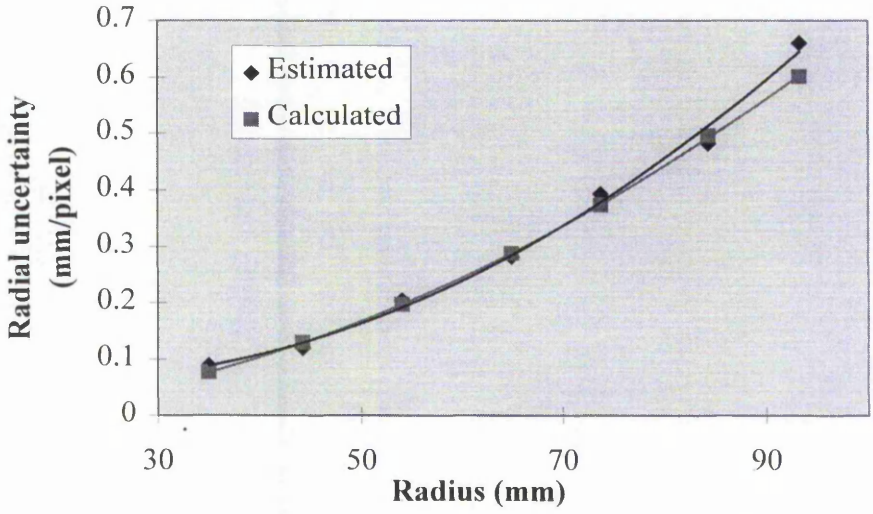

Figure 5-23 *Graph of estimated and calculated radial uncertainties against radius.*

From Figure 5-23 it can be seen that the estimated and the calculated radial uncertainties are in close agreement.

Estimating the Angular Uncertainty:

The angular uncertainty can be estimated graphically from the graphs of Figure 5-17: it is equal to the gradient of each line. For each concentric set of targets, the gradient is calculated at points (1) and (2), which have the largest angular separation (270^0). Both the parallax and the estimated angular uncertainty are shown in Table 5-4. In addition, the calculated angular uncertainty, computed after calibration and space intersection, is shown in the last column of this table.

TABLE 5-4 *Estimated and calculated angular uncertainties.*

Concentric set of targets	Angle(1-2) (^{0})	Δx_{fL} (1-2) (pixel)	Δx_{fR} (1-2) (pixel)	Estimated angular uncertainty (^{0}/pixel)	Calculated angular uncertainty (^{0}/pixel)
A	270.442	2233.5	2233.5	0.12108	0.12158
B	270.418	2235.5	2232	0.12096	0.12158
C	270.339	2234	2231.5	0.12115	0.12158
D	270.339	2233	2235.5	0.12106	0.12158
E	270.331	2235	2235.5	0.12095	0.12158
F	270.150	2232	2231.5	0.12106	0.12158
G	270.063	2242	2224	0.12143	0.12158

From this table, it is noted that the graphically estimated and the calculated uncertainties are in close agreement.

Estimating the Y axis Uncertainty:

The Y axis uncertainty may be estimated from the graphs shown in Figure 5-19. Assuming that these graphs are linear between two consecutive collinear points, the Y axis uncertainty, obtained graphically, is shown in the third column of Table 5-5:

TABLE 5-5 *Estimated and calculated Y axis uncertainties.*

Target	Y axis spatial coordinate (mm)	Estimated Y axis uncertainty (mm/pixel)	Calculated Y axis uncertainty (mm/pixel)
A4	129.926	0.3502	0.3530
B4	109.965	0.3521	0.3550
C4	89.894	0.3617	0.3580
D4	69.822	0.3610	0.3600
E4	49.786	0.3703	0.3630
F4	29.791	0.3712	0.3650
G4	9.929	-	0.3680

The last column in Table 5-5 shows the Y axis uncertainty calculated after calibration and space intersection. As the range variation in the perspective centre to the targets is minimal, the uncertainty in the main axis is approximately constant for all seven targets. The results of both the graphical and the analytical method are similar.

5.7 Experiments with Synthetic Data

In evaluating the robustness of the functional model, it is important to devise a procedure whereby the calibration process is evaluated independently from space intersection. This is because the calibration process is the computationally most complex process in extracting coordinate information from an object workspace and hence, potentially more prone to data reliability problems. Thus, the convergence properties of the calibration must be investigated to ensure that meaningful data are produced in real imaging conditions.

Stochastic effects, such as the spatial quantisation noise present on the images and high-order non-linear effects, such as lens distortions, mean that, in practice, no functional model can be strictly satisfied. Moreover, since no *a priori* knowledge of the precise values of the calibration parameters is available, the assessment of the reliability of the calibration process with data produced from image measurements is problematic. The procedure shown in Figure 5-24 addresses these problems by testing the functional model in controlled conditions.

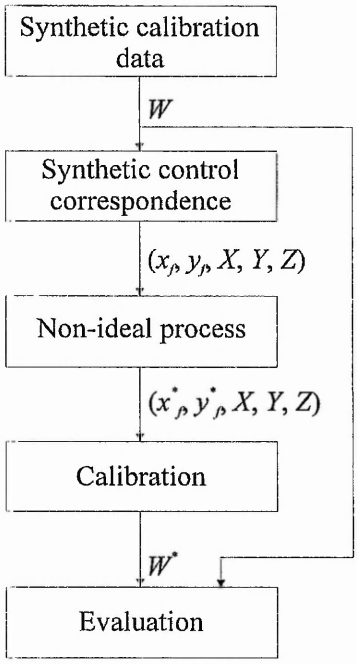

Figure 5-24 *Flow chart of calibration tests.*

To achieve this, a set of synthetic calibration data, denoted by W, is produced. The functional model is then used to produce frame buffer coordinates corresponding to a number of simulated control points. The latter can be either randomly generated or conform to a given geometrical structure, such as, the cylindrical control field. The synthetic control correspondence, denoted by the set (x_f, y_f, X, Y, Z), is subjected to a non-ideal process, such as lens distortion or contamination with noise. These processes are used to simulate the real imaging conditions where lens distortion is present and the spatial and frame buffer coordinates of control points are not known exactly.

Subsequently, the non-ideal control correspondence is used to calibrate the system. The resultant calibration results are denoted by W^*. By comparing the sets W and W^*, the robustness of the calibration process under non-ideal imaging conditions can be established.

5.7.1 Varying the Amount of Control in the Presence of Noise

Spatial quantisation noise can be modelled by adding zero mean Gaussian noise[126] to noise-free synthetic frame buffer coordinates (x_f, y_f). In the following experiments the effect of varying the number of control points in the presence of noise is ascertained. Table 5-6 shows the calibration data used to produce the control correspondence.

TABLE 5-6 *Calibration data used in simulation.*

$U_r t_{IP}$ (rpm.s)	f (mm)	ω, φ, κ (°)	t_x, t_y, t_z (mm)
0.02	50	10, 10, 10	-15, 65, 1100

A simulated control field is used to produce the control data. Both the radius and the Y axis coordinates of the control points are generated randomly with a uniform distribution. The simulated control points occupy a volume approximately equal to that of the real cylindrical control field. The number of control points n is varied from 4 to 50, and uncorrelated Gaussian noise of zero mean and ± 1 pixel maximum is added to the frame buffer coordinates of the control points.

The calibration parameters have different scale. Thus, in order to allow a direct comparison between the calibration unknowns, the error in each parameter is normalised to unity. In addition, a *mean normalised calibration error* (MNCE) is calculated as the mean of the normalised errors in each calibration parameter. Since the MNCE is calculated over all the calibration unknowns, it is an indicator of the robustness of the calibration.

The results of this experiment are presented in Table 5-7. Although four control points are theoretically sufficient for calibration, in the presence of noise a significant number of simulations failed to converge when this minimal amount of control was used. Consequently, in real imaging conditions at least five control points must be used.

TABLE 5-7 *Calibration error for varying amounts of control.*

Parameters	n=5	n=6	n=8	n=15	n=50
$\lvert U_r t_{IP} - U_r t_{IP}^* \rvert / \lvert U_r t_{IP} \rvert$	0.020029	0.000266	0.002005	0.001934	0.000586
$\lvert \omega - \omega^* \rvert / \lvert \omega \rvert$	0.066321	0.001052	0.008779	0.004501	0.003750
$\lvert \varphi - \varphi^* \rvert / \lvert \varphi \rvert$	0.013533	0.011673	0.021709	0.013742	0.000952
$\lvert \kappa - \kappa^* \rvert / \lvert \kappa \rvert$	0.053403	0.006138	0.011572	0.012848	0.003358
$\lvert f - f^* \rvert / \lvert f \rvert$	0.420857	0.294711	0.123085	0.054450	0.007062
$\lvert t_x - t_x^* \rvert / \lvert t_x \rvert$	0.025199	0.005506	0.005473	0.002087	0.001820
$\lvert t_y - t_y^* \rvert / \lvert t_y \rvert$	0.009035	0.000006	0.003492	0.001758	0.000720
$\lvert t_z - t_z^* \rvert / \lvert t_z \rvert$	0.394466	0.281566	0.117619	0.052574	0.006968
MNCE	**0.125355**	**0.075115**	**0.036717**	**0.017987**	**0.003152**

The following conclusions can be drawn from the results of Table 5-7:

i. when few control points are used, the MNCE is dominated by the error in the lens principal distance and the translation component t_z. These two parameters require well-separated non-coplanar control points for reliable estimation. If the control points are coplanar, the two parameters become linearly depended and it is not possible to determine their values. In practice, control points separated by at least an order of magnitude from the maximum value of the radial uncertainty component must be available. Unless large control redundancy is utilised, both the simulated and the real control fields do not provide adequately separated control points for the reliable estimation of the lens principal distance and the translation component t_z in the presence of noise.

ii. with a maximum noise amplitude of ± 1 pixel, there is a tangible improvement in the MNCE when up to fifty control points are used. Thus, all forty-nine targets available on the cylindrical control field should be used in calibrating the line-scan system.

iii. as illustrated in the graph of Figure 5-25, the MNCE falls off initially very rapidly with increasing number of control points.

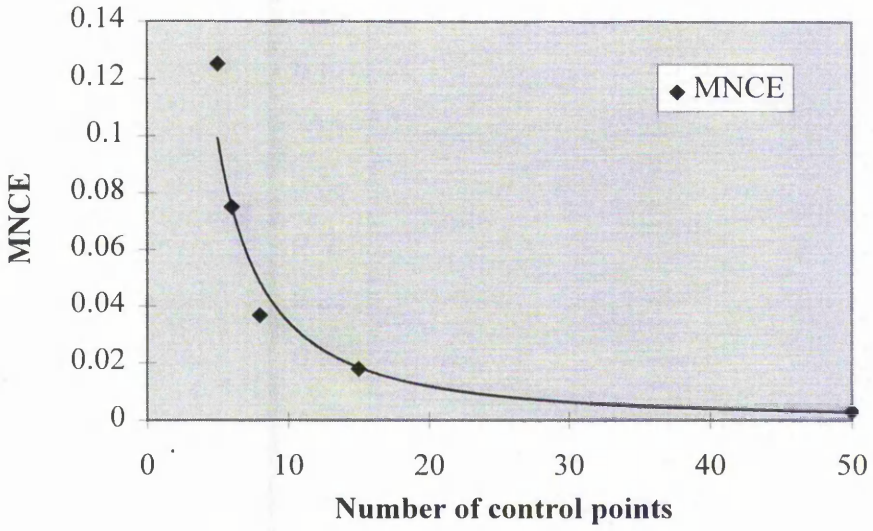

Figure 5-25 *The MNCE as a function of the number of control points.*

When the number of control points is close to the required minimum, the available redundancy is correspondingly minimal and the calibration results are affected to a large extent by the stochastic noise. As the number of control points is increased, the least squares adjustment reduces the effect of this noise.

5.7.2 Effect of Radial Lens Distortion

In this experiment a synthetic control correspondence consisting of fifty control points is produced and the frame buffer coordinates of the targets are subjected to simulated radial lens distortion. The distorted correspondence is used to compute a set of calibration data, which is then compared to the ideal calibration data set. The calibration data used in this experiment are the same as those of the previous experiment (see Table 5-6).

The radial distortion coefficient k computed in section 4.5.1 is increased by a factor of five to yield a maximum distortion of approximately ± 2.5 pixel. The normalised errors in each calibration unknown and the MNCE are shown in Table 5-8.

TABLE 5-8 *Calibration error due to radial lens distortion.*

Parameters	$k = 11.125 \times 10^{-7}$
$\lvert U_r t_{IP} - U_r t_{IP}^* \rvert / \lvert U_r t_{IP} \rvert$	0.000221
$\lvert \omega - \omega^* \rvert / \lvert \omega \rvert$	0.000093
$\lvert \varphi - \varphi^* \rvert / \lvert \varphi \rvert$	0.000339
$\lvert \kappa - \kappa^* \rvert / \lvert \kappa \rvert$	0.000228
$\lvert f - f^* \rvert / \lvert f \rvert$	0.003643
$\lvert t_x - t_x^* \rvert / \lvert t_x \rvert$	0.000045
$\lvert t_y - t_y^* \rvert / \lvert t_y \rvert$	0.000022
$\lvert t_z - t_z^* \rvert / \lvert t_z \rvert$	0.010227
MNCE	**0.001852**

The following conclusions can be drawn from the results presented in Table 5-8:

i. the error in the computed calibration parameters is smaller that that produced by a target location measurement uncertainty of ± 1 pixel. This is because the target measurement uncertainty affects all the control points, whereas radial lens distortion only affects the points close to the upper and lower parts of the image. Therefore, for uniformly distributed control points, relatively few points are affected by this distortion.

ii. since the radial distortion coefficient used in this test was five times larger than its actual value computed in section 4.5.1, radial lens distortion will have a negligible impact on the calibration accuracy of the experimental system.

5.7.3 Projection and Back-Projection Accuracy

The following series of experiments are devised in order to evaluate the overall effect of a non-ideal process on the accuracy of the complete system model. This includes both the

processes of calibration and space intersection. Figure 5-26 shows the flow chart of the synthetic data tests. To evaluate the *projection* accuracy of the system model, the frame buffer coordinates of targets are calculated from their spatial coordinates and the calibration parameters. The *back-projection* accuracy of the system model is the 3-D coordinate measurement error in object space obtained by space intersection.

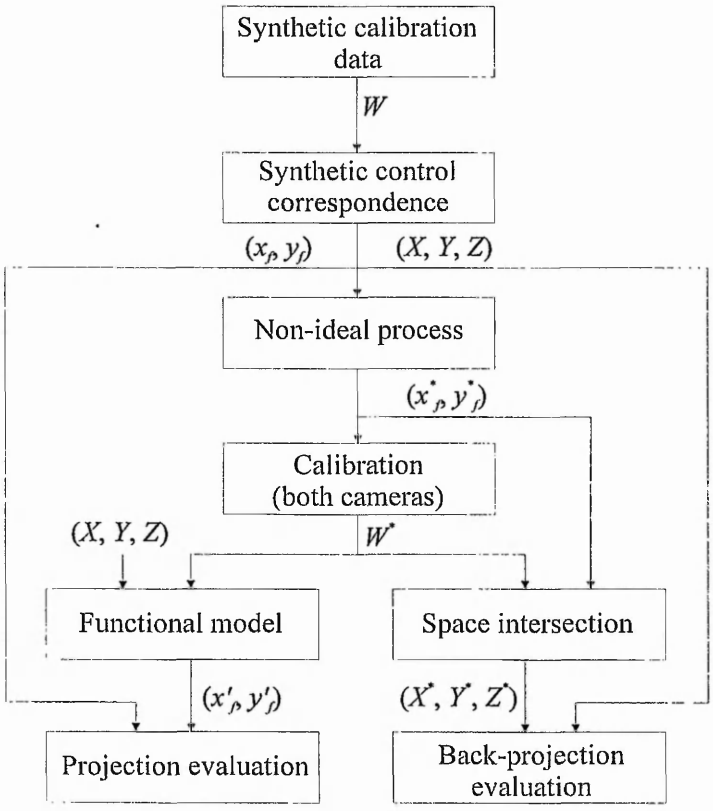

Figure 5-26 *Projection and back-projection tests.*

The ideal (error-free) control correspondence is denoted by (x_f, y_f, X, Y, Z), and the control correspondence produced by the functional model and space intersection is denoted by $(x^*_f, y^*_f, X^*, Y^*, Z^*)$. To determine the projection accuracy of the system model, the following expression is used -

$$\mu = \sqrt{\frac{1}{N}\left(\sum_{n=1}^{N}\left(x_{fn} - x'_{fn}\right)^2 + \sum_{n=1}^{N}\left(y_{fn} - y'_{fn}\right)^2\right)}$$

where -

N is the number of observations;

x_f, y_f are the nominal (error-free) X and Y frame buffer coordinates, respectively;

$x_f^{'}$, $y_f^{'}$ are the X and Y frame buffer coordinates calculated from the calibration data set W^* and the object space coordinates of the control points.

The parameter μ is referred to as *the image error* and is the mean square error of the X and Y frame buffer coordinates calculated over N targets. An image error for each image axis can also be evaluated as -

$$\mu_x = \sqrt{\frac{1}{N}\left(\sum_{n=1}^{N}\left(x_{fn} - x_{fn}^{'}\right)^2\right)} \qquad \mu_y = \sqrt{\frac{1}{N}\left(\sum_{n=1}^{N}\left(y_{fn} - y_{fn}^{'}\right)^2\right)}$$

The following experiment is conducted to evaluate the effect of radial lens distortion on the projection and back-projection accuracy of the system model. Table 5-9 shows the calibration parameters used.

TABLE 5-9 *Calibration data used in simulation.*

$U_r t_{ll}$ (rpm.s)	f (mm)	ω, φ_L, φ_R κ (°)	t_{xL}, t_{xR} t_y, t_z (mm)
0.015	55	1, -10, 10, 1	-15, 15, 65, 1000

The radial lens distortion coefficient is set to $k = 11.125 \times 10^{-7}$, and fifty synthetic control points are generated with uniform distribution. Half of these are used as control points for the purposes of calibration and the remaining points are used as test points.

Table 5-10 shows the mean square errors μ and the maximum projection errors $\delta_{\max}$ in each image axis.

TABLE 5-10 *Projection accuracy.*

Axis	x_{fL}	y_{fL}	x_{fR}	y_{fR}
μ (pixels)	3.86×10^{-5}	0.98101	4.75×10^{-5}	0.98215
$\delta_{\max}$ (pixels)	0.00011	-2.46399	-0.00012	-2.46377

From Table 5-10 it can be seen that the X axis image errors are negligible. This result is expected as radial lens distortion affects only the Y image axis. The maximum Y axis image errors are relatively large, as the system model does not correct for radial lens distortion.

The back-projection errors in each axis due to radial lens distortion are shown in Table 5-11. The second row, labelled $\overline{L}_e$, shows the mean object space error, the third row indicates the standard deviation m_e, and the last row $\Delta_{\max}$ shows the maximum errors.

TABLE 5-11 *Back-projection accuracy.*

Axis	X	Y	Z
$\overline{L}_e$ (mm)	0.00146	0.01325	0.00014
m_e (mm)	0.00264	0.08348	0.00267
$\Delta_{\max}$ (mm)	0.00742	0.19481	-0.00353

From the above table it can be seen that the X and Z axis errors are very small. However, the mean Y axis error is approximately one order of magnitude larger. Overall, the errors presented in the above two tables follow their expected trend, and no erratic behaviour is present.

5.8 Experiments with Real Data

When real data are used, no *a priori* knowledge of the calibration parameters or the projection data is available. Thus, the metrological accuracy of the line-scan system is evaluated using 'ground truth', that is, objects containing a distribution of targets with

known coordinates in a local coordinate system. To achieve this, the following tests are performed:

i. measurement of test points on the cylindrical control field;

ii. measurement of 3-D vectors defined on a generic object.

Recalling from section 5.3.2 that the coordinates of the targets on the cylindrical control field are specified to an accuracy of $\pm 75\mu m$, the results presented in this section are subject to errors not directly related to the mathematical model. Furthermore, a compromise between the number of control and test points has to be made since, from section 5.7.1, all forty-nine targets should be used in calibrating the line-scan system.

5.8.1 Accuracy in the Measurement of Test Points

This experiment involves determining the coordinates of test points on the cylindrical control field. The nominal or approximate parameters of the experimental set-up are shown in Table 5-12 -

TABLE 5-12 *Line-scan system set-up parameters.*

U_r (rpm)	t_{IP} (mS)	f (mm)	ω, $\lvert\varphi_L - \varphi_R\rvert$, κ (°)	t_{xL}, t_{xR} t_y, t_z (mm)
2	10	50	0, 20, 0	-15, 15, 65, 1500

Of the forty-nine targets imaged, the following twenty-five are used for calibration:

A1, A2, A4, A7, B2, B3, B5, B7, C2, C5, C7, D1, D3, D4, D5, E2, E4, E5, F2, F3, F5, F7, G1, G3, G5.

The remaining targets are used as test points. Table 5-13 shows the calibration results of this experiment.

TABLE 5-13 *Calculated calibration parameters.*

Parameters	Left camera	Right camera
$U_r t_{IP}$ (rpm.s)	0.020141	0.020161
ω (°)	-1.511896	0.243810
φ (°)	-94.01425	-72.95121
κ (°)	-1.586053	0.131903
f (mm)	57.74528	60.01034
t_x (mm)	15.64549	-14.45087
t_y (mm)	63.07157	64.23931
t_z (mm)	1659.198	1721.118
Residuals' norm	1.02×10^{-12}	2.96×10^{-13}

The norm of the residual vector, shown in the last row of Table 5-13, indicates that, although half the test field's control points have been used in the calibration, the model fit is good. Having calibrated the line-scan system, space intersection is performed for each of the twenty-four test points. The mean error $\overline{L}_e$ and the standard deviation m_e in each axis are presented in Table 5-14.

TABLE 5-14 *Accuracy in the measurement of test points.*

Axis	X	Y	Z
$\overline{L}_e$ (mm)	0.084417	-0.09713	-0.06188
m_e (mm)	0.147261	0.349605	0.17681

From the above table, it can be seen that the Y axis standard deviation is approximately twice that of the X and Z axis. This result is expected as, for the experimental conditions, the mean value of the circumferential uncertainty, i.e. the product of the radius and the angular uncertainty, is lower than the mean Y axis uncertainty. Indeed, from Table 5-4 of section 5.6, the mean circumferential uncertainty is approximately 0.136mm/pixel, whereas the mean Y axis uncertainty is approximately 0.361mm/pixel.

The spatial error in each test point is presented in Table 5-15.

TABLE 5-15 *Spatial error in the test points.*

Test Point	ΔX (mm)	ΔY (mm)	ΔZ (mm)
B1	0.330	-0.248	-0.058
C1	0.201	-0.347	-0.267
E1	0.044	-0.267	-0.059
F1	0.042	-0.594	-0.468
D2	-0.273	-0.005	0.214
G2	0.095	-0.416	-0.246
A3	-0.184	0.752	-0.273
C3	-0.010	0.744	-0.102
E3	0.009	0.619	0.270
B4	0.123	-0.260	0.288
C4	-0.048	-0.140	0.007
F4	0.058	-0.426	0.091
G4	0.022	-0.437	0.009
A5	0.019	-0.264	-0.002
A6	-0.134	-0.282	0.033
B6	0.208	-0.206	-0.077
C6	0.130	-0.013	-0.039
D6	0.207	0.010	-0.003
E6	0.270	-0.137	-0.067
F6	0.236	-0.198	-0.144
G6	0.236	-0.192	-0.145
D7	0.212	0.065	-0.240
E7	0.106	-0.016	-0.214
G7	0.127	-0.073	0.007

From this table it can be seen that, although half the control points have been used in the calibration, the mathematical model maps the object workspace to a high degree of accuracy. As previously mentioned, part of the error recorded is due to the coordinates of the control points being specified to $\pm 75\mu m$. Furthermore, due to the off-axis views of the target rods and the resulting perspective distortion, the unequal gains of the odd and even video streams and the imperfect finish of the rods, the frame buffer coordinates of some test points may have been estimated to an accuracy lower than ± 1 pixel.

The results obtained from a different system set-up are presented below. In order to produce a radically different system set-up, the integration period, the rotational speed, the nominal lens focal length, the convergence angles, the range and the start of scan are all changed. Table 5-16 contains the approximate system set-up parameters.

TABLE 5-16 *Line-scan system set-up parameters.*

U_r (rpm)	t_{IP} (mS)	f (mm)	ω, $\lvert\varphi_L - \varphi_R\rvert$, κ (°)	t_{xL}, t_{xR} t_y, t_z (mm)
3	7	25	0, 40, 0	-15, 15, 60, 800

The control points used in the previous experiment are employed to calibrate the system. The results of the calibration are shown in Table 5-17.

TABLE 5-17 *Calculated calibration parameters.*

Parameters	Left camera	Right camera
$U_r t_{IP}$ (rpm.s)	0.021087	0.021119
ω (°)	-0.158614	0.053569
φ (°)	-118.3418	-75.14895
κ (°)	-0.246367	-0.093288
f (mm)	27.03242	26.82368
t_x (mm)	16.59464	-12.47871
t_y (mm)	57.61908	58.33130
t_z (mm)	778.6365	785.8020
Residuals' norm	8.28×10^{-15}	1.47×10^{-15}

With the calibration parameters determined, space intersection is performed for the twenty-four test points. The mean errors and the standard deviation in each axis are presented in Table 5-18. Table 5-19 shows the spatial error in each test point.

TABLE 5-18 *Accuracy in the measurement of test points.*

Axis	X	Y	Z
$\overline{L}_e$ (mm)	0.00311	-0.06513	-0.07171
m_e (mm)	0.14825	0.31249	0.16760

TABLE 5-19 *Spatial error in the test points.*

Test Point	ΔX (mm)	ΔY (mm)	ΔZ (mm)
B1	0.372	-0.409	0.317
C1	0.307	-0.427	0.134
E1	-0.066	-0.307	-0.117
F1	-0.002	-0.421	-0.228
D2	-0.215	-0.239	-0.208
G2	0.014	-0.021	-0.068
A3	-0.077	0.641	-0.515
C3	-0.047	0.698	-0.046
E3	-0.001	0.606	-0.047
B4	-0.209	-0.226	-0.448
C4	-0.001	-0.106	-0.013
F4	0.003	-0.240	0.014
G4	0.072	-0.310	-0.050
A5	-0.002	-0.165	0.022
A6	0.141	-0.157	0.011
B6	0.208	-0.206	-0.103
C6	-0.043	-0.065	-0.003
D6	-0.130	-0.045	0.067
E6	0.268	-0.056	-0.062
F6	-0.049	-0.174	-0.055
G6	0.141	-0.125	-0.133
D7	0.081	0.096	-0.043
E7	0.044	0.158	-0.149
G7	-0.062	-0.063	0.002

Although the experimental conditions in the above two experiments are different, the results presented in Tables 5-18 and 5-19 indicate that the mathematical model maps the object workspace with comparable confidence. Indeed, comparison of the mean errors and the

standard deviation shows that, in both cases, the coordinate measurement accuracy is limited mainly by the uncertainty in establishing the frame buffer coordinates of the test points.

5.8.2 Accuracy in the Measurement of a Generic Object

This experiment involves measuring three-dimensional vectors defined on an object other than the cylindrical control field. This is done in order to assess the coordinate measurement accuracy of the line-scan system in a practical application. The object used in this experiment is a metal cylinder having a diameter of 91mm and a height of 144mm. Since no *a priori* knowledge of the exact dimensions of the object is available, nine targets consisting of high-contrast crosshairs are placed randomly on the surface of the object. The targets are labelled '1' to '9'. A number of three-dimensional vectors between them are measured manually with a pair of callipers. The precision of this measurement is estimated to be approximately 0.5mm.

The process of measuring this object starts with the calibration of the line-scan system. After setting-up the system with the cylindrical control field, the control correspondence of the forty-nine targets is established and the system is calibrated. Table 5-20 shows the calibration parameters.

TABLE 5-20 *The calibration parameters.*

Parameters	Left camera	Right camera
$U_r t_{lp}$ (rpm.s)	0.020175	0.020182
ω (°)	0.076350	0.038681
φ (°)	-39.199697	-9.51205
κ (°)	0.114650	0.124578
f (mm)	0.055763	0.054896
t_x (mm)	0.016966	-0.020651
t_y (mm)	0.061621	0.062247
t_z (mm)	1.141911	1.140252
Residuals' norm	3.13×10^{-14}	2.74×10^{-14}

The control field is subsequently removed and replaced by the cylindrical object. Left and right perspective images are acquired and the frame buffer coordinates of the nine targets are measured. Space intersection is performed for each target, yielding the spatial coordinates of the targets in the three-dimensional coordinate system established by the calibration process. The magnitude of the three-dimensional vectors is then calculated. For two targets $p(x_p, y_p, z_p)$ and $q(x_q, y_q, z_q)$, this is given by the norm of the vector $\boldsymbol{p} - \boldsymbol{q}$:

$$\|\boldsymbol{p} - \boldsymbol{q}\| = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2 + (z_p - z_q)^2}$$

Table 5-21 presents the magnitude of the three-dimensional vectors obtained both by the analytical process (labelled 'calculated' values) and the manual measurement (labelled 'measured' values).

Table 5-21 *Magnitude of the three-dimensional vectors.*

Vectors	Calculated (mm)	Measured (mm)	Residual error (mm)
1->8	83.600	84.5	-0.900
1->9	104.428	105.0	-0.572
2->9	71.789	73.0	-1.211
9->3	78.237	78.5	-0.263
3->5	84.174	85.0	-0.826
4->3	90.152	91.5	-1.348
6->2	96.334	95.0	1.334
5->4	35.919	35.5	0.419
6->7	76.492	77.0	-0.508
7->1	79.506	79.5	0.006
8->3	98.487	99.0	-0.513
Mean (mm)	81.738	82.136	-0.398

The accuracy of the manual measurement is considerably lower than that of the line-scan system, thus the distances shown in the third column of Table 5-21 cannot be considered to be definitive. Consequently, this experiment does not aim to quantify the coordinate

measurement accuracy of the line-scan system. Rather, it provides an intuitive method to verify the coordinate measurement capability of the system under real imaging conditions.

The results obtained from the experiments conducted with the stereoscopic system will be discussed further in the next chapter.

6. SUMMARY, CONCLUSIONS AND FUTURE WORK

6.1 Introduction

This chapter concludes this research by:

i. summarising the work carried out in this research;

ii. providing concluding remarks;

iii. presenting possible future directions following on from this research.

6.2 The Imaging Concept

6.2.1 Motivation

An investigation into the imaging of cylindrical objects was carried out. Initially, this involved assessing the standard area array sensor in the context of cylindrical object imaging. The results of this investigation indicated that imaging of objects having a high degree of cylindrical symmetry by means of such sensors suffers from the following limitations:

i. the circumferential resolution decreases rapidly at picture columns of increasing separation from the sensor's centre;

ii. it is difficult to obtain an 'all-round view' of the object;

iii. potentially uncontrollable specular light reflections can occur if the surface of the object under inspection is highly reflective;

iv. any mismatch between the aspect ratio of the object and the area array sensor leads to poor utilisation of the resources of the imaging system, as part of the image area cannot be utilised.

In light of the above limitations, it was concluded that an alternative imaging technique must be developed to allow efficient imaging of cylindrical objects.

6.2.2. The Rotating Object Line-Scan System

A review of existing techniques indicated that photographic cameras employing a slit aperture have been used in the past to image cylindrical objects. Such systems overcome the limitations of conventional photographic or electronic image cameras that have a two-dimensional image plane. This is made possible by the different projective properties of slit cameras and the application of rotational object motion. Specifically, a slit aperture photographic camera realises the perspective projection in the slit axis, and the orthographic projection in the motion axis. The equivalent of a slit aperture camera in electronic form is the line-scan camera. Thus, this device was selected to be the imaging sensor for the cylindrical object vision system.

The rotating object line-scan system was subsequently analysed. The following points can be made about this system:

i. the spatial resolution is constant for a given radius;

ii. an angular field of view of 360^0 is readily obtained, and the Y axis field of view can be set independently, in relation to the object's height;

iii. specular reflections affecting image quality can be reduced or completely eliminated.

Sample images demonstrating these attributes were presented in the third chapter. A ballistics application, involving imaging of highly specular metallic surfaces, indicated the advantages of the rotating object line-scan system.

6.2.3 The Image Sensors

The operation of the line-scan sensor is summarised below. This device employs a single column of photosensitive elements and, therefore, has different imaging characteristics from the area array sensor. In particular, the field of view in the main axis depends on:

i. the lens principal distance;

ii. the sensor length;

iii. the sensor to object range.

For a two-dimensional line-scan image produced either by lateral motion parallel to the camera face and normal to the main axis or by rotational motion as in the rotating object line-scan system, the field of view in the X image axis is controlled by:

i. the integration period;

ii. the relative speed of the motion between the camera and the object;

iii. the number of scan lines stored in the image buffer memory.

In section 2.5.3 it was seen that because the field of view in each image axis is determined by different parameters, a line-scan camera can produce affine images, i.e. of different scale in the X and Y image axes. The parameters that determine the resolution of a line-scan system were also discussed in section 2.5.4.

6.3 The Experimental Two-Dimensional System

Details of the experimental line-scan system were presented in the third chapter. This included a description of the hardware and the operation of the system. A geometrical model describing the two-dimensional system was established and object space coordinate

extraction algorithms were developed. A series of experiments were carried out to quantify the consistency and reliability of the experimental system operating in two-dimensional mode.

The experiment conducted to establish the consistency of the rotational speed indicated that, for the purposes of this work, the rotational speed can be controlled accurately enough. Specifically, it was shown that, even in the worst case, the error caused by speed variation is at least an order of magnitude less than the resolution of the system (see section 3.7.2).

Additional experiments were carried out to establish the validity of the mathematical model. The results of these experiments indicated that the two-dimensional system had the necessary attributes to be used for coordinate measurement and could thus form the basis of a stereoscopic system.

6.4 The Stereoscopic System

6.4.1 The System Model

The ability to perform dimensional measurement with the stereoscopic line-scan system was a fundamental requirement of this work. Essentially, this task involves the ability to reconstruct an object workspace from the pair of perspective images produced by the system. To achieve this, the processes of *calibration* and *space intersection* were employed.

The aim of the calibration process is to determine the interior, exterior and temporal parameters of each camera for a given system set-up. These parameters are:

i. the lens principal distance;

ii. three Euler angles and a three-dimensional vector, mapping the orientation and the location of the camera, respectively, in a reference object space coordinate system. The incorporation of the reference coordinate system is required in order to allow coordinate measurement in a local, independent coordinate system, rather than the camera-centred frame;

iii. the product of the camera's integration period with the rotational speed.

The derivation of the calibration model involved three independent coordinate transformations: a *rigid* transformation from the object space coordinate system to the camera frame, an orthographic and a perspective *projective* transformation from the camera frame to the X and Y image space axes respectively, and a *linear mapping* of the image space coordinates to the frame buffer memory. The projective and the linear mapping transformations were adapted from the two-dimensional system algorithms by incorporating the necessary changes in the geometry of the two-dimensional system. This entailed shifting the instantaneous field of view of each line-scan camera off the rotation axis (see section 4.4.1).

In addition to the resulting eight calibration parameters, the incorporation of lens distortion correction into the system model was considered. The analysis presented in section 4.5.1 indicated that lens distortion correction parameters should not be included to the calibration model. This is because, for the lenses used in this work, the maximum radial distortion is comparable to the uncertainty component in the sensor's main axis. In general, if the mathematical model does not describe the physical system accurately, numerical instability will occur, thus producing unreliable results.

The process of space intersection was employed in order to extract coordinate information from an object of interest. This requires the eight calibration parameters and the projection of unknown spatial points in the left and right images to be known. Space intersection entails determining the point of intersection of the two collinear rays defined by the unknown point in space, the perspective centres of the two lenses and the corresponding images of the point. Ideally, these rays are straight lines intersecting at the unknown point. In practice, however, the two rays can be skewed due to lens non-linearities. Furthermore, the presence of spatial quantisation noise will generally produce non-intersecting rays. To solve this problem, the vector of the minimum distance between the two rays can be determined, and the spatial point can be assumed to lie at the mid-distance of the vector. In this work, determination of the spatial location of the unknown point is performed using the singular value decomposition method, which, although more abstract and computationally

more intensive than the mid-point method, produces more accurate results (see section 4.4.2, 'phase 4').

Both the calibration and the space intersection processes are optimisation problems, that is, their solution involves determining the minimum value of an objective function. This is because the functional model of the line-scan system, utilised in both processes, is an approximation of the physical system. Furthermore, the discrete photosensitive elements of the sensor give rise to spatial quantisation noise, which, in turn, introduces errors in the calibration and the space intersection processes. To this extent, a least squares adjustment, in conjunction with redundant input data, is employed to produce a statistically optimal solution. In the calibration process, redundant data are obtained by utilising more than the minimum required number of control points, whereas in space intersection redundancy is inherent when two or more views, acquired from different perspectives, are used.

6.4.2 The Experimental Results

Imaging characteristics -

Experiments were conducted to characterise the imaging properties of the line-scan system. Initially, these experiments aimed to identify the conditions required to produce stereoscopic parallax. It was shown that if the instantaneous fields of view (IFOV) of the two cameras converge on the rotation axis, no stereoscopic parallax is produced. Thus, extraction of three-dimensional coordinate data requires that the IFOVs converge at a finite distance from the rotation axis.

As indicated by the spatial sampling pattern of the stereoscopic line-scan system (see section 4.3), the distribution of voxels is symmetrical about the rotation axis. The dimensions of the voxels in the stereoscopic region at planes normal to the rotation axis can thus be quantified by an angular and a radial component. A Y axis component, defined along the rotation axis, is also required. Both the angular and the radial uncertainties, defined for a point at a given radius, are independent of the absolute angle the point makes with the reference object space coordinate system and its Y axis coordinate. They are, however, non-linear functions of the radius. Conversely, the Y axis uncertainty depends

linearly on the range of the point of interest from the sensor, the lens' principal distance and the photosite aperture, but it is independent of the angle of the point.

The voxels can be thought of as a graphical interpretation of the spatial sampling properties of an imaging system that utilises discrete image sensors. In order to analytically determine the spatial dimensions of the uncertainty components, the mathematical model developed for coordinate measurement is required. In addition to this method, a less accurate, albeit more intuitive, method was also utilised in the imaging characteristics experiments. This graphical method involved measuring the gradient of the X and Y image data as functions of the radial parallax, the angle and the Y axis component of a target. The results of this graphical method were compared with those obtained analytically. It should be noted that the graphical method is subject to larger errors as perfect system alignment is assumed and the gradient of the tangents cannot be determined accurately if the graphs are non-linear. Nevertheless, the results of both methods were found to be in close agreement.

Experiments with synthetic data -

This series of experiments aimed to analyse the robustness of the calibration process by simulating the real imaging conditions. Computer simulations are useful when tests utilising real data cannot be performed. This is the case with the calibration process, as no *a priori* knowledge of the calibration data exists. Although the metrological accuracy of the line-scan system can be established without resorting to an independent evaluation of the calibration process, it is imperative to analyse the robustness of the latter for two reasons:

- the calibration process is computationally more intensive than space intersection, and hence subject to larger errors;
- the spatial quantisation noise can, to some extent, 'mask' calibration inaccuracies, thus rendering the identification of the source of coordinate measurement errors inconclusive.

The following tests utilising synthetic data were performed:

i. evaluation of the effect of spatial quantisation noise on the robustness of the calibration;

ii. evaluation of the effect of radial lens distortion on the robustness of the calibration;

iii. evaluation of the effect of radial lens distortion on the coordinate measurement accuracy of the stereoscopic line-scan system.

In order to evaluate the effect of spatial quantisation noise on the calibration process, the number of control points was varied in the presence of Gaussian noise. To achieve this, the following procedure was devised. Synthetic sets of calibration parameters and control points were established. The system model was used to produce the corresponding 'ideal' synthetic control correspondence. The frame buffer coordinates of the control points were then subjected to Gaussian noise of zero mean and ± 1 pixel magnitude. The resultant control correspondence was used to calibrate the system and the calibration results were compared with the 'ideal' set. This process was repeated for 4, 5, 6, 8, 15 and 50 control points. The results indicated that the calibration error drops rapidly as the number of control points is increased from the minimum. When fifty control points are used, the calibration error is insignificant. Consequently, all forty-nine targets present in the cylindrical control field should be used to calibrate the experimental line-scan system.

The analysis presented in the main text (see section 4.5.1.) indicated that lens distortion correction parameters should not be incorporated into the system model. However, in order to evaluate the effects of radial lens distortion on the system model, a suitable experiment was devised. This involved accentuating the radial non-linearity of the lenses used in this work by a factor of five to produce a 'worst case' scenario. Fifty control targets were generated with uniform probability distribution and their frame buffer coordinates were subjected to radial lens distortion. The results of this experiment (see section 5.7.2) indicate that radial lens distortion has a minimal impact on the robustness of the calibration.

Experiments with real data -

Experiments were conducted to determine the coordinate measurement accuracy of the stereoscopic line-scan system when real objects are imaged. The first two tests involved using half the targets of the cylindrical control field to calibrate the system, whilst the remaining targets were used as test points. After calibration of the line-scan system, the coordinates of the test points were determined through space intersection, and the results compared with the calibration data supplied for the control field.

Ideally, the coordinates of the test points must be known to an accuracy of at least an order of magnitude higher than the expected accuracy of the line-scan system. However, technical limitations in the coordinate measurement machine utilised to calibrate the cylindrical control field resulted in a measurement accuracy of $\pm 75\mu m$. The results obtained by the two experiments (see section 5.8.1) indicated a mean error of less than $100\mu m$ and a standard deviation of approximately $200\mu m$ in the X and Z axes, and $350\mu m$ in the Y axis. Consequently, these figures can be expected to be compromised by a factor not directly related to the metrological accuracy of the line-scan system.

It can be seen that the standard deviation in the X and Z axes are approximately half of that in the Y axis. However, this does not have to be the case. For example, if the cameras' sensors consisted of twice the number of photosensitive elements and the Y axis field of view was kept constant, *ceteris paribus*, the Y axis standard deviation would nominally be halved. Furthermore, in contrast to the Y axis uncertainty which, for the experimental conditions, remains almost constant with radius, the radial uncertainty that determines the X and Z resolution is a non-linear function of the radius. Thus, the Y axis mean square error is approximately constant, but the X and Z axis mean square errors are strongly affected by the radius of the targets (see Tables 5-3 and 5-5, section 5.6). Furthermore, the images produced by the stereoscopic line-scan system consist of 3048 lines in the X axis, but only 508 pixels in the Y axis.

The experiments discussed so far utilised the cylindrical control field both for the purposes of calibration and as a test object. However, in a practical situation, the control field would be used to calibrate the system and then be substituted by the object to be measured. Hence, a test was devised in which a cylindrical object with a number of targets distributed over its

surface was imaged. A number of three-dimensional vectors defined between targets were measured manually with a pair of callipers, and the same distances were determined using the stereoscopic line-scan system. The results indicate that the coordinate measurement errors are higher than those obtained in the previously mentioned experiments. This is because the manual measurement has lower precision than that offered by the line-scan system.

6.5 Conclusions

The principal objective of this research was to develop and analyse a machine vision system capable of efficient and accurate inspection and dimensional measurement of cylindrical objects. The investigation into different imaging sensors and operating modes initially concentrated on the standard area array sensor. However, it was found that this sensor is far from ideal when imaging of cylindrical objects is required. Thus, the rotating object line-scan system was developed.

The conclusions that can be drawn from this research are presented below.

In the imaging concept -

i. By applying rotational motion to the object under inspection and utilising a line-scan sensor, a 360^0 field of view can be obtained in a single image. Thus, the complete surface of the object can be efficiently inspected;

ii. The spatial resolution is constant over the whole surface of the object, and it is not affected by the object's curvature; consequently, the whole of the cylindrical surface appears "unfolded" into a flat sheet;

iii. Lighting has to be optimised only along the instantaneous field of view, i.e. a thin strip, rather than a large area on the surface of the object. Hence, if a highly reflective surface is imaged, specular reflections can be controlled more effectively in the line-scan system;

iv. The aspect ratio of the line-scan images can be set according to the dimensions of the object; that is, the instantaneous field of view can be optimised for a given object height without affecting the angular field of view.

In the two-dimensional line-scan system -

i. An experimental system conforming to the previously discussed imaging concepts was designed and built;

ii. Sample images of cylindrical objects were acquired, demonstrating the imaging characteristics of the rotating object line-scan system;

iii. Details of a ballistics application utilising the line-scan system to image firearm cartridge cases were given;

iv. A mathematical model was developed in order to facilitate the modelling of the system and allow coordinate measurement;

v. Experiments were conducted to verify the integrity of the experimental system; in particular, the consistency of the rotational speed was independently evaluated, and was found to be precise enough for the purposes of this work;

vi. Further experimentation was carried out to substantiate the mathematical model. The accuracy of both the X and the Y axis algorithms was established, thus allowing the development of a stereoscopic variant of the two-dimensional system.

In the analysis of the stereoscopic line-scan system -

i. The stereoscopic region was defined;

ii. The spatial sampling pattern of the system was presented, and the factors determining the voxel dimensions were identified (also see point x, below);

iii. A functional model describing the geometry of each camera was established. This included three independent coordinate transformations: a rigid transformation from

the object space to the camera coordinate system, a perspective and orthographic transformation from the camera to the image plane, and a linear mapping from the image to the frame buffer coordinate systems;

iv. A rigorous calibration technique was developed, which computed the eight calibration unknowns in one step. This utilised a full-scale iterative process;

v. Space intersection algorithms were developed to allow extraction of three-dimensional coordinate data from an object of interest;

vi. Both calibration and space intersection were implemented using a least squares adjustment to allow stochastic modelling of the system. This enabled the computation of a statistically optimal solution;

vii. The issue of lens distortion was considered. It was found that, for the hardware used in this work, lens distortion correction coefficients should not be incorporated into the system model;

viii. The implementation of the system model addressed the issues of data quality. Tests performed on the input data consist of checking the control correspondence for singularities. If singularities are detected, the corresponding data are removed. The iterative optimisation is continuously monitored to prevent divergence. The computed solution is checked to verify the orthonormality of the rotation matrices, and the variance-covariance matrix of the calibration unknowns is displayed. In addition, the norm of the residual vectors is computed;

ix. A method to compute an approximate solution for the spatial coordinates of unknown points was devised. The results of this step are used as an initial approximation for space intersection, so that reliable convergence is attained;

x. A numerical method utilising the system model and the calibration parameters was developed to calculate the spatial quantisation error at a given point in the stereoscopic region;

xi. Software code was written to allow the implementation of the calibration and space intersection processes.

In the experiments conducted with the stereoscopic system -

i. The design of a calibration structure, specifically developed for this work, was presented;

ii. Stereoscopic parallax is produced only when the instantaneous fields of view of the cameras converge at a finite distance from the rotation axis;

iii. Graphs depicting the X axis frame buffer coordinates as a function of radius and angle, and Y axis frame buffer coordinates against Y axis spatial distance were presented;

iv. Experiments were carried out to determine graphically the uncertainty components for a typical system set-up;

v. The results of the above step were compared with those obtained by the analytical process utilising the system model;

vi. An experimental strategy based on computer simulations was devised to characterise the robustness of the calibration process in the presence of noise and lens non-linearities;

vii. The coordinate measurement accuracy of the stereoscopic line-scan system was established using both the cylindrical control field and a generic object. For the set-ups considered, the results indicate a precision of approximately $200\mu m$ in the X and Z axes, and $350\mu m$ in the Y axis.

The experiments both with the two-dimensional and the stereoscopic system indicate that there is no significant discrepancy between the results obtained and those which are predicted theoretically.

6.6 Future Work

This section presents the authors' suggestions on possible future work that can be considered with the stereoscopic line-scan system.

6.6.1 Automating the Calibration Process

Although the system model developed during the course of this research produces accurate results, the calibration process is labour intensive and time-consuming. This is because the frame buffer coordinates of forty-nine targets have to be manually measured. Similarly, the correspondence of each point is manually established, as is the control correspondence data file. Since it was not the intention of this work to automatically solve the correspondence problem, this limitation was considered to be acceptable. Nevertheless, it is accepted that the successful integration of the line-scan system into a practical application would benefit from an automated solution to the above problems. To this extent, image processing algorithms could be developed to identify, label and measure the coordinates of the control points automatically[127,128]. This would then allow the automated production of the control correspondence, so that the calibration process would be considerably simplified.

6.6.2 Improving Accuracy

Subpixelation -

Image processing algorithms to extract the frame buffer coordinates of targets to subpixel accuracy can be developed for use with the line-scan system. Such algorithms would increase the spatial accuracy of the system (see section 5.2.1). The implementation of such a technique would have to take into account various factors, such as target distortion due to oblique views, the affine nature of the line-scan images, i.e. the different scale in each image axis, and the radiometric properties of the cameras.

The process that would benefit most from subpixelation is space intersection. This is because the calibration can utilise a large number of redundant control points, which, in

conjunction with the least squares adjustment, decreases the effects of the spatial quantisation noise. The redundancy available in space intersection, however, is determined by the number of cameras used. To this extent, it may be possible to utilise more than two cameras, so that space intersection would produce more accurate results.

Modelling of lens non-linearity -

If a subpixel technique is implemented, lens non-linearities would, in general, have to be taken into account. For this reason, the system model developed in the fourth chapter has been extended to encompass radial lens distortion correction parameters. The relevant formulation is presented in Appendix II. Although this model takes into account only radial distortion, numerous researchers have shown that this type of distortion is approximately an order of magnitude larger than tangential distortion. Nevertheless, in highly critical applications or if wide-angle lens are used, tangential distortion correction parameters should also be incorporated into the system model. It should be stated that, despite the essentially one-dimensional geometry of the line-scan sensor, tangential distortion still affects the images as the distortion is produced by the lens.

6.6.3 Further Investigation of the Line-Scan Sensor

Additional work can be carried out in the following areas:

i. the radiometric performance of the line-scan sensors;

ii. modulation transfer function experiments.

An investigation into the radiometric properties of the line-scan system was beyond the scope of the work undertaken here. Such an analysis would concentrate on the evaluation of the noise, the linearity and the dynamic range of the line-scan system. As feature detection is strongly affected by these factors, the radiometric characteristics of the line-scan system can be of critical importance in a practical application.

Modulation transfer function (MTF) experiments can be carried out to evaluate the spatial resolution of the line-scan system under different operating conditions. Such an analysis would assess the resolution of the complete imaging system, from the camera optics to the video display unit[129].

REFERENCES

References

1. D. Hutber, "Active Stereo Vision and its Application to Industrial Inspection", *IEEE Colloquium on Active and Passive Techniques for 3-D Vision*, pp. 1-4, 1991.

2. M. Becker, J. Weber, E. Schubert, "Sensor System for Subpixel Geometry and Surface Inspection of Cuboid and Ring-Shaped Objects", *Proceedings of SPIE Videometrics*, Vol. 3174, pp. 258-267, 1997.

3. C. S. Fraser. M. R. Shortis, "Vision Metrology in Industrial Inspection: A Practical Evaluation", *Proceedings of ISPRS Close Range Techniques and Machine Vision*, Vol. 30, Part 5, pp. 87-91, 1994.

4. A. D. Marshall, D. R. Roberts, "Automatically Planning the Inspection of Three-Dimensional Objects Using Stereo Computer Vision", *Proceedings of SPIE Three-Dimensional and Unconventional Imaging for Industrial Inspection and Metrology*, Vol. 2599, pp. 94-105, 1995.

5. P. Seitz, M. Gale, H. Meier, J. M. Raynor, "Locometer: On-line Inspection of Locomotive Wheel to Rail Movements Using High Precision CCD Metrology", *SPIE In-Process Optical Measurements and Industrial Methods*, Vol. 1266, pp. 39-45, 1990.

6. R. L. Horst, M. Negin, "Machine Vision System for Precision Dimensional Measurements and On-Line SPC", *IEEE Conference Record, Industry Applications Society Annual Meeting*, Vol. 2, pp. 1712-1716, 1989.

7. F. Angrilli, G. Bianchini, G. Fanti, M. Mozzi, "On Line Measurements to Control the Forming Process of Glass Vials", *Proceedings of the SPIE Videometrics*, Vol. 1820, pp. 145-157, 1992.

8. N. Harle, M. Xi, "Robot Vision System Using Differential Image Information and Geometrical Relations of Moving Edges", *Proceedings of the International Symposium on Signal Processing and its Applications*, pp. 573-574, 1996.

9. Y. F. Li, M. H. Lee, M. A. Rodrigues, J. J. Rowland, "Supervisory Robotic Control Under Vision Guidance", *Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems*, Vol. 3, pp. 2030-2035, 1994.

10. A. Ohya, E. Shoji, S. Yuta, "3-D Range Sensor Using Fiber Grating for Recognition of Autonomous Mobile Robot's Passage Space", *Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems*, Vol. 3, pp. 1759-1763, 1994.

11. S. B. Marapane, M. M. Trivedi, "Extracting Depth by Binocular Stereo in a Robot Vision System", *Proceedings of the SPIE Applications of Digital Image Processing XI*, Vol. 974, pp. 97-102, 1988.

12. H. T. Roman, B. A. Pellegrino, W. R. Sigrist, "Pipe Crawling Inspection Robots: An Overview", *IEEE Transactions on Energy Conversion*, Vol. 8, No. 3, pp. 577-583, 1993.

13. R. Cipolla, N. J. Hollinghurst, "Human-Robot Interface by Pointing with Uncalibrated Stereo Vision", *Image and Vision Computing*, Vol. 14, pp. 171-178, 1996.

14. R. A. Bachnak, M. Celenk, "A Stereo System for 3-D Measurements in Robot Workspaces", *Proceedings of the IEEE International Symposium on Intelligent Control*, pp. 293-298, 1989.

15. S. X. Godber, M. Robinson, J. P. O. Evans, "The Line-Scan Sensor: An Alternative Sensor Modality for the Extraction of 3-D Co-ordinate Information", *Optical Engineering - The Journal of SPIE*, 1995.

16. B. Neldam, "Vision Based Inspection and Quality Control for Use in Industrial Laundries", *Proceedings of the SPIE Industrial Inspection*, Vol. 1010, pp. 118-121, 1988.

17. A. N. Zographos, J. P. O. Evans, S. X. Godber, M. Robinson, "A Line-Scan System for All-Round Inspection of Objects", *Proceedings of the SPIE Videometrics*, Vol. 3174, pp. 274-282, 1997.

18. R. Benosman, T. Maniere, J. Devars, "Multidirectional Stereovision Sensor, Calibration and Scenes Reconstruction", *Proceedings of the IEEE 13th International Conference on Pattern Recognition*, pp. 161-165, 1996.

19. *"Optics for Photography"*, R. Kingslake, SPIE Optical Engineering Press, p. 25, 1992.

20. *"The Development of Novel Stereoscopic Imaging Sensors"*, S. X. Godber, Nottingham Polytechnic, PhD Thesis, 1991.

21. *"Stereoscopic Line-Scan Imaging Using Rotational Motion"*, R. S. Petty, The Nottingham Trent University, PhD Thesis, 1997.

22. D. H. Chung, I. D. Yun, S. U. Lee, "Registration of Multiple-Range Views Using the Reverse-Calibration Technique", *Pattern Recognition*, Vol. 31, No. 4, pp. 457-464, 1998.

23. K. Miwada, H. Ito, H. Yamamoto, "A 100MHz Data-Rate, 5000-Element CCD Linear Image Sensor with Reset Pulse Level Adjustment Circuit", *IEEE International Solid-State Circuits Conference*, pp. 168-169, 1992.

24. Y. Li, T. Young, C. Huang, "Noncontact Measurement Using Line-Scan Cameras: Analysis of Positioning Error", *IEEE Transactions on Industrial Electronics*, Vol. 36, No. 4, pp. 545-551, 1989.

25. D. F. Garcia, M. A. del Rio, J. L. Diaz, F. J. Suarez, "Flatness Defect Measurement System for Steel Industry Based on a Real-Time Linear-Image Processor", *Proceedings of IEEE Systems, Man and Cybernetics*, Vol. 3, pp. 331-336, 1993.

26. M. A. Colbert, T. H. Nayfeh, A. H. Nayfeh, J. C. McKeeman, "An Optical System for Three-Dimensional Motion Measurement", *IEEE Proceedings*, pp. 873-876, 1990.

27. S. Murai, F. Otomo, H. Ohtani, "Automated Three Dimensional Measurement Using Stereo CCD Camera in the Application to Close Range Photogrammetry", *International Archives of Photogrammetry & Remote Sensing*, Vol. 26, Part 5, pp. 409-413, 1986.

28. W. Lutz, G. Holzmuller, G. Steinwender, E. S. Hochmair, "Fast Measuring of Solid Angles by Means of CCD Line Scan Cameras with Analytically Rectified Optics",

Proceedings of 16th Annual International Conference of the IEEE, Engineering in Medicine and Biology Society, pp. 1007-1008, 1994.

29. Y. Yamashita, N. Saeki, K. Hayashi, N. Suzuki, "Automated Three-Dimensional Measurement Using Multiple One-Dimensional Solid-State Image Sensors and Laser Spot Scanners", *16th International Congress of ISPRS*, Vol. 27, Part B5, 1988.

30. J. Shabushnig, "Inspection of Pharmaceutical Packaging with Linear-Array Video Sensors", *Proceedings of the Conference on Vision '89, Society of Manufacturing Engineers*, pp. 1-13 - 1-23, 1989.

31. L. Khoudour, L. Duvieubourg, B. Meunier, Y. F. Wan, J. P. Deparis, "A New Fast Algorithm Using an Adaptive Structuring Element Applied to a Counting Device", *Proceedings of the IEEE International Conference on Image Processing*, pp. 57-60, 1996.

32. J. P. Deparis, L. Khoudour, B. Meunier, L. Duvieubourg, "A Device for Counting Passengers Making Use of Two Active Linear Cameras: Comparison of Algorithms", *IEEE International Conference on Systems, Man and Cybernetics*, pp. 1629-1634, 1996.

33. D. J. Svetkoff, D. B. Kilgus, R. C. Boman, "3-D Line Scan Imaging with Orthogonal Symmetry", *Proceedings of SPIE 3D and Unconventional Imaging for Industrial Inspection & Metrology*, Vol. 2599, pp. 177-188, 1996.

34. J. L. Bruyelle, J. G. Postaire, "Disparity Analysis for Real Time Obstacle Detection by Linear Stereovision", *Proceedings of the IEEE, Intelligent Vehicles Symposium*, pp. 51-56, 1992.

35. J. C. Burie, J. G. Postaire, "Enhancement of the Road Safety with a Stereovision System Based on Linear Cameras", *Proceedings of the IEEE, Intelligent Vehicles Symposium*, pp. 147-152, 1996.

36. S. Okada, H. Miyauchi, M. Imade, T. Sumimoto, H. Yamamoto, "3-D Shape Measurement Based on an Adaptive Lighting-Pattern Projection Method", *IEEE*

Proceedings of International Conference on Industrial Electronics, Control and Instrumentation, pp. 2362-2367, 1991.

37. D. J. Gugan, I. J. Dowman, "Topographic Mapping from SPOT Imagery", *Photogrammetric Engineering and Remote Sensing*, Vol. 54, No. 10, pp. 1409-1414, 1988.

38. *"Image Sensing and Solid State Camera Products 1995/1996"*, EG&G Reticon, 1995.

39. D. Lake, *"Solid State Cameras"*, Fairchild CCD Imaging, 1993.

40. *"LD2040 Digital Line Scan Camera (Preliminary)"*, EG&G Reticon, Ca., 1997.

41. M. Miethig, J. C. Renn, B. C. Doody, P. T. Jenkins, R. G. Ambrose, DALSA Inc., "A Miniature Variable Scan Camera for Machine Vision", *Proceedings of SPIE Cameras, Scanners and Image Acquisition Systems*, Vol. 1901, pp. 202-210, 1993.

42. *"Image Processing & Optic Technology"*, Exhibition, Birmingham, England, 1997.

43. *"Lord Ingenierie Digital Linescan Cameras"*, Lambda Photometrics World-Wide-Web Page, http://www.lambdaphoto.co.uk/vision/Linescan.htm, 1998.

44. *"Optical Science, Engineering, and Instrumentation"*, Exhibition, San Diego, California USA, 1997.

45. W. S. Boyle, G. E. Smith, "Charge Coupled Semiconductor Devices", *Bell System Technical Journal*, Vol. 49, pp. 587-593, 1970.

46. *"Solid-State Imaging with Charge-Coupled Devices"*, A. J. P. Theuwissen, Kluwer Academic, 1995.

47. *"Modular Series on Solid-State Devices"*, D. K. Schroder, Addison-Wesley, pp. 77-125, 1987.

48. V. Kaftandjian, Y. M. Zhu, G. Peix, D. Babot, "Contrast Transfer Function Measurement of X-Ray Solid State Linear Detectors Using Bar/Space Pattern Methods", *Non-Destructive Testing & Evaluation International*, Vol. 29, No. 1, pp. 3-11, 1996.

49. *"Manual of Photogrammetry"*, American Society of Photogrammetry, Fourth Edition, Falls Church, Va., p. 1034, 1980.

50. Y. Shan Li, T. Y. Young, J. A. Magerl, "Subpixel Edge Detection and Estimation with a Microprocessor-Controlled Line Scan Camera", *IEEE Transactions on Industrial Electronics*, Vol. 35, No. 1, pp. 105-112, 1988.

51. P. Brand, R. Mohr, "Accuracy in Image Measure", *Proceedings of SPIE Videometrics III*, Vol. 2350, pp. 218-228, 1994.

52. K. Jensen, D. Anastassiou, "Subpixel Edge Localization and the Interpolation of Still Images", *IEEE Transactions on Image Processing,* Vol. 4, No. 3, pp. 285-295, 1995.

53. *"LC1902 Modular Line-Scan Camera - User's Manual"*, EG&G Reticon, USA, 1988.

54. *"LC1902 Line-Scan Camera - User's Manual"*, EG&G Reticon, USA, 1987.

55. *"Imaging Products - P360F Power Grabber/Introduction & Installation"*, Dipix Technologies, Canada, 1995.

56. *"Imaging Products - P360F Power Grabber/Hardware Reference"*, Dipix Technologies, Canada, 1995.

57. *"Imaging Products - P360F Power Grabber / Function Library"*, Dipix Technologies, Canada, 1995.

58. *"RPS Operating Manual"*, North East Electronics Ltd, Tyne and Wear, UK, 1990.

59. *"SMCU / MC3E Interface Specification"*, North East Electronics Ltd, Tyne and Wear, UK, 1993.

60. A. Zographos, M. Robinson, C. Smith, "Ballistics Identification Using Line-Scan Imaging Techniques", *IEEE Proceedings of International Carnahan Conference on Security Technology*, 31st Annual, Canberra, Australia, pp. 82-87, 1997.

61. *"Analytical Photogrammetry"*, Ghosh, S. K., Pergamon Press, Second Edition, pp. 42-44, 1988.

62. *"Observations and Least Squares"*, E. M. Mikhail, Harper & Row, pp.40-42, 1976.

63. *"Quick Assembly Two and Three Channel Optical Encoders - Technical Data, HEDS-5500/5540"*, Hewlett Packard, 1996.

64. G. Wei, S. De Ma, "Implicit and Explicit Camera Calibration - Theory and Experiments", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, No. 5, pp. 469-480, 1994.

65. M. A. Sid-Ahmed, M. T. Boraie, "Dual Camera Calibration for 3-D Machine Vision Metrology", *IEEE Transactions on Instrumentation and Measurement"*, Vol. 39, No. 3, pp. 512-516, 1990.

66. R. K. Miller, D. G. Stewart, W. H. Brockman, S. B. Skaar, "A Camera Space Control System for an Automated Forklift", *IEEE Transactions on Robotics and Automation*, Vol. 10, No. 5, pp. 710-716, 1994.

67. Y. El-Manadili, K. Novak, "Precision Rectification of SPOT Imagery Using the Direct Linear Transformation Model", *Photogrammetric Engineering and Remote Sensing*, Vol. 62, No. 1, pp. 67-72, 1996.

68. B. H. Mitchell, R. G. Chadwick, J. F. McCabe, "Stereomicroscope Photogrammetry for the Measurement of Small Objects", *Photogrammetric Record*, Vol. 13, No. 74, pp. 289-299, 1989.

69. L. P. Adams, B. A. Geems, G. G. Jaros, J. Peter, S. Wynchank, "A Stereophotogrammetric controlled Pointing Device for Surgical Use", *Proceedings of ISPRS Close Range Techniques and Machine Vision*, Vol. 30, Part 5, pp. 1-7, 1994.

70. *"Bigshot Digital Cameras"*, Dicomed Inc., Dicomed World-Wide-Web Home Page, http://www.dicomed.com/bigshot.html, 1998.

71. S. G. Chamberlain, S. R. Kammasz, C. R. Smith, W. D. Washkurak, M. G. Farrier, "Mega Pixel CCD Image Sensor Technology", Dalsa Inc., *International Electron Devices Meeting - Technical Digest*, pp. 29.1.1-29.1.4, 1994.

72. J. Peipe, "Photogrammetric Performance Evaluation of a 4096 × 4096 Pixel Digital Camera Back", *Photogrammetric Record*, Vol. 15, No. 89, pp. 803-805, 1997.

73. M. Halloran, "7000 × 9000 Imaging on an Integrated CCD Wafer", *Advanced Imaging*, Vol. 11, No. 1, pp. 46-48, 1996.

74. C. Chatterjee, V. P. Roychowdhury, E. K. P. Chong, "A Non-Linear Gauss-Seidel Algorithm for Noncoplanar and Coplanar Camera Calibration with Convergence Analysis", *Computer Vision and Image Understanding*, Vol. 67, No. 1, pp. 58-80, 1997.

75. R. Y. Tsai, "A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses", *IEEE Journal of Robotics and Automation*, Vol. RA-3, No. 4, pp. 323-344, 1987.

76. X. Wan, G. Xu, "Camera Parameters Estimation and Evaluation in Active Vision System", *Pattern Recognition*, Vol. 29, No. 3, pp. 439-447, 1996.

77. G. Toscani, R. Vaillant, R. Deriche, O. D. Faugeras, "Stereo Camera Calibration Using the Environment", *Proceedings of the 6th Scandinavian Conference on Image Analysis*, Finland, pp. 953-960, 1989.

78. L. Robert, "Camera Calibration without Feature Extraction", *Computer Vision and Image Understanding*, Vol. 63, No. 2, pp. 314-325, 1996.

79. W. I. Grosky, L. A. Tamburino, "A Unified Approach to the Linear Camera Calibration Problem", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 7, pp.663-671, 1990.

80. S. Shah, J. K. Aggarwal, "Intrinsic Parameter Calibration Procedure for a (High Distortion) Fish-Eye Lens Camera with Distortion Model and Accuracy Estimation", *Pattern Recognition*, Vol. 29, No. 11, pp. 1775-1788, 1996.

81. M. Roberts, A. J. Naftel, "A Genetic Algorithm Approach to Camera Calibration in 3D Machine Vision", *IEE Colloquium on Genetic Algorithms in Image Processing and Vision*, pp. 12/1 - 12/5, 1994.

82. F. Dornaika, C. Garcia, "Robust Camera Calibration Using 2D to 3D Feature Correspondences", *Proceedings of SPIE Videometrics V*, Vol. 3174, pp. 123-129, 1997.

83. Y. Fu, J. Zhang, B. Xia, "Pyramid Calibration and Correspondence Analysis in Binocular Vision", *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, pp. 597-600, 1991.

84. G. P. Stein, "Accurate Internal Camera Calibration Using Rotation, with Analysis of Sources of Error", *Proceedings of IEEE International Conference on Computer Vision*, pp. 230-236, 1995.

85. K. D. Gremban, C. E. Thorpe, T. Kanade, "Geometric Camera Calibration Using Systems of Linear Equations", *Proceedings of IEEE on Robotics and Automation*, pp. 562-567, 1988.

86. M. A. Sid-Ahmed, M. Boraie, W. Loy, "Photogrammetric Aerotriangulation Using Matrix CCD Cameras for Close-Range Position Sensing", *Computers in Industry*, Vol. 12, pp. 307-311, 1989.

87. A. H. A. El-Beik, E. M. Masaad, "Development of Multi-Application Collinearity Equations Model", *Photogrammetric Record*, Vol. 14, No. 82, pp. 635-649, 1993.

88. A. Okamoto, "The Model Construction Problem Using the Collinearity Condition", *Photogrammetric Engineering and Remote Sensing*, Vol. 50, No. 6, pp. 705-711, 1984.

89. M. Qiu, S. Ma, "The Nonparametric Approach for Camera Calibration", *IEEE Proceedings of the Fifth International Conference on Computer Vision*, pp. 224-229, 1995.

90. R. Chung, S. K. Wong, "Stereo Calibration from Correspondences of OTV Projections", *IEE Proceedings in Visual Image Signal Processing*, Vol. 142, No. 2, 1995.

91. I. W. Faig, "Calibration of Close-Range Photogrammetric Systems - Mathematical Formulation", *Photogrammetric Engineering and Remote Sensing*, Vol. 41, No. 12, pp. 1479-1486, 1975.

92. Y. I. Abdel-Aziz, H. M. Karara, "Direct Linear Transformation from Comparator Coordinates into Object Space Coordinates", *Proceedings of the ASP/UI Symposium on Close-Range Photogrammetry*, pp. 1-18, 1971.

93. A. J. Naftel, J. C. Boot, "An Iterative Linear Transformation Algorithm for Solution of the Collinearity Equations", *Photogrammetric Engineering and Remote Sensing*, Vol. 57, No. 7, pp. 913-919, 1991.

94. H. A. Beyer, "Accurate Calibration of CCD-Cameras", *IEEE Proceedings of Computer Vision and Pattern Recognition Conference*, pp. 96-101, 1992.

95. R. Horaud, R. Mohr, B. Lorecki, "On Single-Scanline Camera Calibration", *IEEE Transactions on Robotics and Automation*, Vol. 9, No. 1, pp.71-75, 1993.

96. R. M. Haralick, L. G. Shapiro, "Glossary of Computer Vision Terms", *Pattern Recognition*, Vol. 24, No. 1, pp. 69-93, 1991.

97. P. Gros, R. Hartley, R. Mohr, L. Quan, "Reply - How Useful is Projective Geometry?", *Computer Vision and Image Understanding*, Vol. 65, No. 3, pp.442-446, 1997.

98. J. G. Fryer, D. C. Brown, "Lens Distortion for Close-Range Photogrammetry", *Photogrammetric Engineering and Remote Sensing*, Vol. 52, No. 1, pp. 51-58, 1986.

99. D. C. Brown, "Decentering Distortion of Lenses", *Photogrammetric Engineering*, Vol. 32, No. 3, pp. 444-462, 1966.

100. E. H. Thompson, "A Note on Distortion", *Photogrammetric Record*, Vol. 9, No. 49, pp. 93-99, 1977.

101. S. F. El-Hakim, "System Calibration and Self-Calibration, Part I - Rotationally Symmetrical Lens Distortion and Image Deformation", *Photogrammetric Engineering and Remote Sensing*, Vol. 52, No. 10, pp. 1617-1625, 1986.

102. *"Adjustment by Least Squares in Geodesy and Photogrammetry"*, R. A. Hirvonen, Frederick Ungar Publishing, 1971.

103. *"Method of Least Squares and Principles of the Theory of Observations"*, Y. V. Linnik, Pergamon Press, 1961.

104. *"Numerical Recipes in C - The Art of Scientific Computing"*, W. H. Press (Ed.), Cambridge University Press, Second Edition, 1992.

105. R. I. Hartley, P. Sturm, "Triangulation", *Computer Vision and Image Understanding*, Vol. 68, No. 2, pp. 146-157, 1997.

106. T. A. Clarke, T. J. Ellis, S. Robson, "High Accuracy 3-D Measurement Using Multiple Camera Views", *IEEE Colloquium on 3D Imaging and Analysis of Depth/Range Images*, pp. 1-4, 1994.

107. C. Rothwell, O. Faugeras, G. Csurka, "A Comparison of Projective Reconstruction Methods for Pairs of Views", *Computer Vision and Image Understanding*, Vol. 68, No. 1, pp. 37-58, 1997.

108. *"Close Range Photogrammetry and Machine Vision"*, K. B. Atkinson (Ed.), Whittles Publishing, p. 44, 1996.

109. *"Observations and Least Squares"*, E. M. Mikhail, Harper & Row, p. 104, 1976.

110. S. Shih, Y. Hung, W. Lin, "When Should We Consider Lens Distortion in Camera Calibration", *Pattern Recognition*, Vol. 28, No. 3, pp. 447-461, 1995.

111. "*Handbook of Applicable Mathematics, Volume III - Numerical Methods"*, R. F. Churchhouse, John Wiley & Sons, pp. 174, 1981.

112. B. A. King, "Some Considerations for the Statistical Testing of Least Squares Adjustments of Photogrammetric Bundles", *Photogrammetric Record*, Vol. XV, No. 90, pp. 929-935, 1997.

113. J. Weng, "Camera Calibration with Distortion Models and Accuracy Evaluation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 10, pp. 965-980, 1992.

114. *"Photogrammetry"*, F. H. Moffitt, E. M. Mikhail, Harper&Row, Third Edition, pp. 516-521, 1980.

115. C. Heipke, "A Global Approach for Least-Squares Image Matching and Surface Reconstruction in Object Space", *Photogrammetric Engineering and Remote Sensing*, Vol. 58, No. 3, pp. 317-323, 1992.

116. G. A. Jones, "Constraint, Optimisation, and Hierarchy: Reviewing Stereoscopic Correspondence of Complete Features", *Computer Vision and Image Understanding*, Vol. 65, No. 1, pp. 57-78, 1997.

117. O. Hellwich, W. Faig, "Graph-Based Feature Matching Using Descriptive and Relational Parameters", *Photogrammetric Engineering and Remote Sensing*, Vol. 60, No. 4, pp. 443-450, 1994.

118. R. Horaud, T. Skordas, "Stereo Correspondence Through Feature Grouping and Maximal Cliques", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 11, pp. 1168-1180, 1989.

119. Z. Lan, R. Mohr, "Robust Matching by Partial Correlation", *INRIA Rapport de recherche, Programme 4: Robotique, image et vision*, 22 pages, 1995.

120. T. Kanade, M. Okutomi, "A Stereo Matching Algorithm with an Adaptive Window: Theory and Experiment", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, No. 9, pp. 920-932, 1994.

121. R. J. Valkenburg, A. M. McIvor, "Accurate 3D Measurement Using a Structured Light System", *Image and Vision Computing*, Vol. 16, pp. 99-110, 1998.

122. R. Singh, D. P. Chapman, K. B. Atkinson, "Digital Photogrammetry for Automatic Close Range Measurement of Textureless and Featureless Objects", *Photogrammetric Record*, Vol. 15, No. 89, pp. 691-702, 1997.

123. K. S. Roh, I. S. Kweon, "2-D Object Recognition Using Invariant Contour Descriptor and Projective Refinement", *Pattern Recognition*, Vol. 31, No. 4, pp. 441-455, 1998.

124. M. Kisworo, S. Venkatesh, G. West, "Modeling Edges at Subpixel Accuracy Using the Local Energy Approach", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, No. 4, pp. 405-410, 1994.

125. X. Liu, R. W. Ehrich, "Subpixel Edge Location in Binary Images Using Dithering", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 17, No. 6, pp. 629-634, 1995.

126. O. Silven, M. Hihnala, "Comparison of Two Stereo Vision-Based Motion Determination Methods", SPIE Vol. 1613 Mobile Robots, pp. 218-227, 1991.

127. S. F. El-Hakim, "Real-Time Metrology with CCD Cameras", *Photogrammetric Engineering and Remote Sensing*, Vol. 52, No. 11, pp. 1757-1766, 1986.

128. C. S. Fraser, "Innovations in Automation for Vision Metrology Systems, *Photogrammetric Record*, Vol. 15, No. 90, pp. 901-911, 1997.

129. *"Solid-State Imaging with Charge-Coupled Devices"*, A. J. P. Theuwissen, Kluwer Academic Publishers, Netherlands, pp. 142-156, 1995.

Appendix I

The Rigid Transformation and the Rotation Matrix

A transformation between two three-dimensional coordinate systems, whereby both true shape and scale are retained, is referred to as a *rigid* transformation. Two three-dimensional rectangular coordinate systems are depicted in Figure A1-1. A rigid transformation from the $X'Y'Z'$ to the XYZ system requires the application of a rotation about each of the X', Y' and Z' axes, denoted by Euler angles ω, φ and κ (omega, phi and kappa) respectively. These angles are defined as positive if they are clockwise when viewed from the origin of their respective axis. Also required is a translation along the X, Y and Z axes through vector $\boldsymbol{t} = \begin{bmatrix} t_x & t_y & t_z \end{bmatrix}^{\mathrm{T}}$. The positive direction for each axis is indicated by the relevant arrowheads on Figure A1-1.

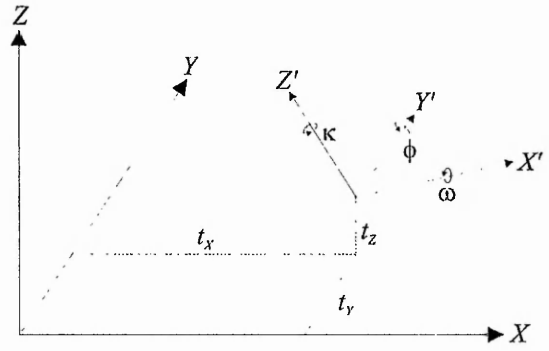

Figure A1-1 *Coordinate transformation.*

If vectors $\bar{\boldsymbol{x}} = \begin{bmatrix} X & Y & Z \end{bmatrix}^{\mathrm{T}}$ and $\boldsymbol{x} = \begin{bmatrix} X' & Y' & Z' \end{bmatrix}^{\mathrm{T}}$ define the location of a point in the XYZ (transformed) and $X'Y'Z'$ (original) systems respectively, provided that the order of rotation is ω, φ, κ and all rotations are positive, the rigid transformation can be expressed as -

$$\bar{\boldsymbol{x}} = \boldsymbol{R}^{\mathrm{T}}\boldsymbol{x} + \boldsymbol{t}$$

where $\boldsymbol{R}$ is the *rotation matrix*, presented below.

Figure A1-2 shows a point P in a rectangular (xyz) system with coordinates (x, y, z). It is required to calculate the coordinates of this point in a rectangular coordinate system (X, Y,

Z) obtained by three independent rotations ω, φ and κ about the x, y and z axes respectively.

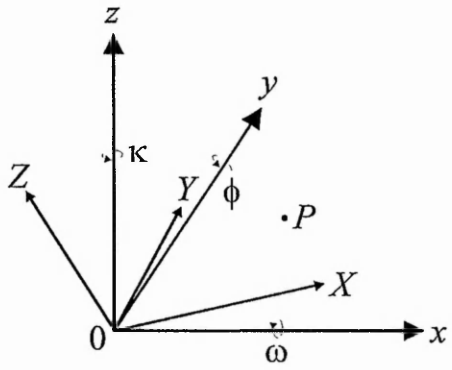

Figure A1-2 *Original and rotated coordinate systems.*

Vector $\boldsymbol{P}$ defined from the origin of the coordinate systems to point P has coordinate components $\begin{bmatrix} x & y & z \end{bmatrix}^{\mathsf{T}}$. If clockwise rotation about the X axis is applied the coordinates of $\boldsymbol{P}$ become -

$$\boldsymbol{P}_{\omega} = \boldsymbol{R}_{\omega} \begin{bmatrix} x & y & z \end{bmatrix}^{\mathsf{T}}$$

where -

$$\boldsymbol{R}_{\omega} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\omega & \sin\omega \\ 0 & -\sin\omega & \cos\omega \end{bmatrix}$$

The clockwise rotation about the Y axis produces -

$$\boldsymbol{P}_{\omega\varphi} = \boldsymbol{R}_{\varphi} \boldsymbol{R}_{\omega} \begin{bmatrix} x & y & z \end{bmatrix}^{\mathsf{T}}$$

where -

$$\boldsymbol{R}_{\varphi} = \begin{bmatrix} \cos\varphi & 0 & -\sin\varphi \\ 0 & 1 & 0 \\ \sin\varphi & 0 & \cos\varphi \end{bmatrix}$$

Finally, the clockwise rotation about the Z axis gives -

$$\boldsymbol{P}_{\omega\varphi\kappa} = \boldsymbol{R}_{\kappa}\boldsymbol{R}_{\varphi}\boldsymbol{R}_{\omega}\begin{bmatrix} x & y & z \end{bmatrix}^{\mathsf{T}}$$

where -

$$\boldsymbol{R}_{\kappa} = \begin{bmatrix} \cos\kappa & \sin\kappa & 0 \\ -\sin\kappa & \cos\kappa & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The rotation matrix $\boldsymbol{R}$ is obtained by multiplying $\boldsymbol{R}_{\omega}$, $\boldsymbol{R}_{\varphi}$ and $\boldsymbol{R}_{\kappa}$ as -

$$\boldsymbol{R} = \boldsymbol{R}_{\omega}\boldsymbol{R}_{\varphi}\boldsymbol{R}_{\kappa} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

with -

$$\begin{aligned}
r_{11} &= \cos\varphi\cos\kappa \\
r_{12} &= \sin\omega\sin\varphi\cos\kappa + \cos\omega\sin\kappa \\
r_{13} &= -\cos\omega\sin\varphi\cos\kappa + \sin\omega\sin\kappa \\
r_{21} &= -\cos\varphi\sin\kappa \\
r_{22} &= -\sin\omega\sin\varphi\sin\kappa + \cos\omega\cos\kappa \\
r_{23} &= \cos\omega\sin\varphi\sin\kappa + \sin\omega\cos\kappa \\
r_{31} &= \sin\varphi \\
r_{32} &= -\sin\omega\cos\varphi \\
r_{33} &= \cos\omega\cos\varphi
\end{aligned}$$

Appendix II

Extension of the System Algorithms to Account for Radial Lens Distortion

Radial lens distortion is caused by imperfect curvature of the lens elements and is symmetrical about the optical axis. Its effect is to reduce or increase the lens focal length, and hence the lateral magnification, at the outer parts of the image. This gives rise to *barrel* or *pincushion* distortion respectively. In Figure A2 straight lines parallel to the edge of the field of view are reproduced as curved lines. When barrel distortion exists, these lines are concave towards the optical axis. Similarly, pincushion distortion produces convex lines.

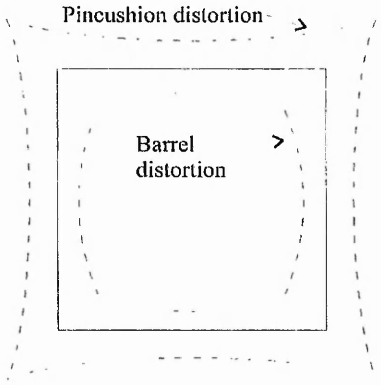

Figure A2 *Radial lens distortion.*

Let y_u and y_d be the undistorted and distorted image space coordinates of a spatial point respectively. A lens radial distortion factor D_y may be used to transform from undistorted to distorted image space coordinates as -

$$y_u = y_d + D_y \qquad \text{A2-1}$$

where the image space coordinates y_u and y_d are specified in milimeters. The radial distortion factor is commonly expressed as a polynomial of the form -

$$D_y = y_d\left(k_1 y_d^2 + k_2 y_d^4\right) \qquad \text{A2-2}$$

where the distortion coefficients are denoted by k_1 and k_2 and up to fifth order distortion components have been included. Combining equations A2-1 and A2-2 produces -

$$y_u = y_d\left(1 + k_1 y_d^2 + k_2 y_d^4\right) \qquad \text{A2-3}$$

Transforming from (distorted) frame buffer coordinates y_f to distorted image space coordinates y_d -

$$y_d = s_y\left(y_f - C_y\right) \qquad \text{A2-4}$$

where C_y is the Y axis frame buffer image centre and s_y is the Y axis scale factor.

The Y axis algorithm, given by equation 4-12 of section 4.4.2, can thus be expressed as -

$$s_y\left(y_f - C_y\right)\left(1 + k_1 y_d^2 + k_2 y_d^4\right) = f\left(\frac{r_{21}.X_w + r_{22}.Y_w + r_{23}.Z_w - T_y}{T_z - \sqrt{\left(r_{11}.X_w + r_{12}.Y_w + r_{13}.Z_w\right)^2 + \left(r_{31}.X_w + r_{32}.Y_w + r_{33}.Z_w\right)^2 - T_x^2}}\right) \qquad \text{A2-5}$$

Appendix III

The Partial Derivatives of the System Algorithms

The partial derivatives of the system algorithms with respect to the calibration unknowns are -

$$j_{11} = -\frac{x_f}{U_r t_{IP}}$$

$$j_{12} = \alpha_x \left\{ -\beta_x \left(-r_{33}Y + r_{32}Z \right) + \delta \left[X_c \left(r_{13}Y + r_{12}Z \right) + Z_c \left(r_{33}Y + r_{32}Z \right) \right] \right\}$$

$$j_{13} = \alpha_x \left[\delta Z_c \left(\varepsilon - X_c \cos\kappa \right) - \beta_x \varepsilon \right]$$

$$j_{14} = \alpha_x \delta X_c \left(Y_c + t_y \right)$$

$$j_{16} = -\frac{\alpha_x \gamma_x}{t_x}$$

$$j_{22} = \beta_y \left\{ -r_{23}Y + r_{22}Z + \gamma_y \left[X_c \left(-r_{13}Y + r_{12}Z \right) + Z_c \left(-r_{33}Y + r_{32}Z \right) \right] \right\}$$

$$j_{23} = \beta_y Z_c \left[\sin\kappa + \gamma_y \left(\varepsilon - X_c \cos\kappa \right) \right]$$

$$j_{24} = \beta_y X_c \left[-1 + \gamma_y \left(Y_c + t_y \right) \right]$$

$$j_{25} = \frac{\beta_y Y_c}{f}$$

$$j_{26} = -\beta_y \gamma_y t_x$$

$$j_{27} = -\beta_y$$

$$j_{28} = -\frac{\beta_y Y_c}{t_z - \alpha_y}$$

where -

$$X_c = r_{11}X + r_{12}Y + r_{13}Z$$
$$Y_c = r_{21}X + r_{22}Y + r_{23}Z - t_y$$
$$Z_c = r_{31}X + r_{32}Y + r_{33}Z$$

$$r_c = \sqrt{X_c^2 + Z_c^2}$$

$$\alpha_x = \frac{s_x}{r_c}$$

$$\beta_x = \left[1 - \left(\frac{Z_c}{r_c}\right)^2\right]^{-1/2}, \quad \text{if} \quad -\frac{\pi}{2} < \omega_1 \le \frac{\pi}{2}$$

$$\beta_x = -\left[1 - \left(\frac{Z_c}{r_c}\right)^2\right]^{-1/2}, \quad \text{if} \quad \frac{\pi}{2} < \omega_1 \le \frac{3\pi}{2}$$

$$\gamma_x = t_x\left[1 - \left(\frac{t_x}{r_c}\right)^2\right]^{-1/2}$$

$$\delta = \frac{\gamma_x + \beta_x Z_c}{r_c^2}$$

$$\varepsilon = X\cos\varphi + Y\sin\omega\sin\varphi - Z\cos\omega\sin\varphi$$

$$\alpha_y = \sqrt{r_c^2 - t_x^2}$$

$$\beta_y = -\frac{f}{s_y}\cdot\frac{1}{t_z - a_y}.$$

$$\gamma_y = \frac{Y_c}{a_y\left(t_z - \alpha_y\right)}$$

The partial derivatives of the system algorithms with respect to the coordinates of spatial points are -

$$j_{11} = \alpha_x \left\{ \beta_x r_{31} - \gamma \left[\left(r_{11}^2 + r_{31}^2 \right) X + \left(r_{11} r_{12} + r_{31} r_{32} \right) Y + \left(r_{11} r_{13} + r_{31} r_{33} \right) Z \right] \right\}$$

$$j_{12} = \alpha_x \left\{ \beta_x r_{32} - \gamma \left[\left(r_{11} r_{12} + r_{31} r_{32} \right) X + \left(r_{12}^2 + r_{32}^2 \right) Y + \left(r_{12} r_{13} + r_{32} r_{33} \right) Z \right] \right\}$$

$$j_{13} = \alpha_x \left\{ \beta_x r_{33} - \gamma \left[\left(r_{11} r_{31} + r_{31} r_{33} \right) X + \left(r_{12} r_{13} + r_{32} r_{33} \right) Y + \left(r_{13}^2 + r_{33}^2 \right) Z \right] \right\}$$

$$j_{21} = \alpha_y \left[r_{21} + \beta_y \left(r_{11} X_c + r_{31} Z_c \right) \right]$$

$$j_{22} = \alpha_y \left[r_{22} + \beta_y \left(r_{12} X_c + r_{32} Z_c \right) \right]$$

$$j_{23} = \alpha_y \left[r_{23} + \beta_y \left(r_{13} X_c + r_{33} Z_c \right) \right]$$

where -

$$\alpha_x = \frac{s_x}{r_c}$$

$$\beta_x = -\left[1 - \left(\frac{Z_c}{r_c} \right)^2 \right]^{-1/2}, \quad \text{if} \quad -\frac{\pi}{2} < \omega_1 \leq \frac{\pi}{2}$$

$$\beta_x = \left[1 - \left(\frac{Z_c}{r_c} \right)^2 \right]^{-1/2}, \quad \text{if} \quad \frac{\pi}{2} < \omega_1 \leq \frac{3\pi}{2}$$

$$\gamma = \frac{1}{r_c^2} \left\{ \beta_x Z_c - t_x \left[1 - \left(\frac{t_x}{r_c} \right)^2 \right]^{-1/2} \right\}$$

$$\alpha_y = -\frac{f}{s_y} \cdot \frac{1}{t_z - \sqrt{r_c^2 - t_x^2}}$$

$$\beta_y = \frac{1}{\sqrt{r_c^2 - t_x^2}} \cdot \frac{Y_c}{t_z - \sqrt{r_c^2 - t_x^2}}$$

Appendix IV

Software Code

File: Calibrate1Doc.h
Purpose: Defines the CCalibrate1Doc class (Visual C++ v5)

```
//
////////////////////////////////////////////////////////////////////////

#if !defined(AFX_CALIBRATE1DOC_H__8627DBEB_DE89_11D1_9B78_C802892FDE4F__INCLUDED_)
#define AFX_CALIBRATE1DQC_H__8627DBEB_DE89_11D1_9B78_C802892FDE4F__INCLUDED_

#if _MSC_VER >= 1000
#pragma once
#endif // _MSC_VER >= 1000

#include "stdafx.h"
#include "calibrate1.h"
#include "Calibration.h"

#include "Operations.h"
#include "ObjectSpace.h"
#include "FrameBuffer.h"

#include "Intersection.h"
#include "InputDataDialog.h"
#include "IntersectCorr.h"
#include "InterDiagnostics.h"

class CCalibrate1Doc : public CDocument
{
protected:
    CCalibrate1Doc();
    DECLARE_DYNCREATE(CCalibrate1Doc)

    BOOL ClearMatrices();

    CCalibrationData*   m_pInitCalibL;    // Initial estimates
    CCalibrationData*   m_pInitCalibR;
    CCalibrationData*   m_pCalibDataL;    // The calibration objects
    CCalibrationData*   m_pCalibDataR;
    CCalibrationData*   m_pCalibResL;     // Calibration results
    CCalibrationData*   m_pCalibResR;

    COperations*        m_pOperations;
    CObjectSpace*       m_pObjectSpace;
    CFrameBuffer*       m_pFrameBuffer;
    CFrameBuffer*       m_pFrameBufferS;

    double**    m_ppRL;         //  The Rotation matrices
    double**    m_ppRR;
    double**    m_ppJ;          //  The Jacobian matrix
    double**    m_ppCvmL;       //  The Variance-Covariance matrices
    double**    m_ppCvmR;
    double**    m_ppV;          //  Vector of Singular values

    UINT*       m_pnvZoneL;     //  Zone conditions
    UINT*       m_pnvZoneR;

    double*     m_pdvFrmCalc;   //  Vector of calculated coordinates
    double*     m_pdvX;         //  Vector of current corrections
    double*     m_pdvF;         //  Vector of functions to be minimised
```

```
    double*     m_pdvW;          //  SVD Vector

    UINT    m_nTotPts, m_nCP,        // Control points
            m_nItL,    m_nItR,       // Iterations
            m_nSngL,   m_nSngR;      // Singularities

    BOOL    m_bFlagL, m_bFlagR,      // Orthonormality flags
            m_bAllocFlag;            // Heap allocation flag

    double  m_dNormL, m_dNormR,      // Norm of residual vectors
            m_dSthL,  m_dSthR;       // Singularity thresholds

    CString  m_sNormL, m_sNormR, m_sSngL, m_sSngR;

    CIntersection*  m_pInter;
    UCR             m_ucr;           // Uncertainty structure
    OBJSPACE        m_finOS;         // Spatial solution
    double          m_dAppRadius;    // Approximate radius

//////////////////////////////////////////////////////////////////
// Operations
public:

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CCalibrate1Doc)
    public:
    virtual BOOL OnNewDocument();
    virtual void Serialize(CArchive& ar);
    virtual void OnCloseDocument();
    //}}AFX_VIRTUAL

// Implementation
public:
    virtual ~CCalibrate1Doc();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif

protected:

// Generated message map functions
protected:
    //{{AFX_MSG(CCalibrate1Doc)
    afx_msg void OnCalibrationStart();
    afx_msg void OnUpdateCalibrationStart(CCmdUI* pCmdUI);
    afx_msg void OnCalibrationOpen();
    afx_msg void OnIntersectionStart();
    afx_msg void OnUpdateIntersectionStart(CCmdUI* pCmdUI);
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

//{{AFX_INSERT_LOCATION}}

#endif //
!defined(AFX_CALIBRATE1DOC_H__8627DBEB_DE89_11D1_9B78_C802892FDE4F__INCLUDED_)
```

File: Calibration.h
Purpose: Defines the CCalibrationData class.

```
#if !defined( CALIBRATION_H )
#define CALIBRATION_H

#include "stdafx.h"
#include <math.h>
#include "Definitions.h"

class CCalibrationData : public CCmdTarget
{
private:
    double  m_urtip,
            m_omega,
            m_phi,
            m_kappa,
            m_f,
            m_tx,
            m_ty,
            m_tz;

    DECLARE_SERIAL( CCalibrationData )

public:
    CCalibrationData():     m_urtip( 0.0 ),
                            m_omega( 0.0 ),
                            m_phi( 0.0 ),
                            m_kappa( 0.0 ),
                            m_f( 0.0 ),
                            m_tx( 0.0 ),
                            m_ty( 0.0 ),
                            m_tz( 0.0 )
    {
    }

    CCalibrationData(  double urtip,
                       double omega,
                       double phi,
                       double kappa,
                       double f,
                       double tx,
                       double ty,
                       double tz ):

        m_urtip( urtip ),
        m_omega( omega ),
        m_phi( phi ),
        m_kappa( kappa ),   .
        m_f( f ),
        m_tx( tx ),
        m_ty( ty ),
        m_tz( tz )
    {
    }

    ~CCalibrationData()    { }
```

```
    double   Geturtip() const    { return m_urtip; }
    double   Getomega() const    { return m_omega; }
    double   Getphi() const      { return m_phi; }
    double   Getkappa() const    { return m_kappa; }
    double   Getf() const        { return m_f; }
    double   Gettx() const       { return m_tx; }
    double   Getty() const       { return m_ty; }
    double   Gettz() const       { return m_tz; }

    void     Seturtip( double urtip )      { m_urtip = urtip; }
    void     Setomega( double omega )      { m_omega = omega; }
    void     Setphi( double phi )          { m_phi = phi; }
    void     Setkappa( double kappa )      { m_kappa = kappa; }
    void     Setf( double f )              { m_f = f; }
    void     Settx( double tx )            { m_tx = tx; }
    void     Setty( double ty )            { m_ty = ty; }
    void     Settz( double tz )            { m_tz = tz; }

    void     ConstructRotationMatrix( CCalibrationData&, double** );
    int      CorrectAndTest( CCalibrationData&, double*, UINT );
    void     Serialize( CArchive& archive );
    UINT     LoadCorrData( double* );
};

// The sole global functions:
inline double DegToRad( double x )    { return (PI * x / 180.0); }
inline double RadToDeg( double x )    { return (x * 180.0 / PI); }

#endif
```

File: ObjectSpace.h
Purpose: Defines the CObjectSpace class.

```
#if !defined( OBJECTSPACE_H )
#define OBJECTSPACE_H

#include "Calibration.h"

class CObjectSpace : public CCalibrationData
{
private:
    UINT     m_Pt;
    double   m_Xos,
             m_Yos,
             m_Zos;

public:
    CObjectSpace():   m_Pt( 0 ),
                      m_Xos( 0.0 ),
                      m_Yos( 0.0 ),
                      m_Zos( 0.0 )
    {
    }

    ~CObjectSpace()   { }
```

```
    void SetObjCoords( double z, double y, double x, UINT p )
    {
        m_Pt = p;
        m_Xos = x * 1e-3;
        m_Yos = y * 1e-3;
        m_Zos = z * 1e-3;
    }

    UINT    Getp() const   { return m_Pt; }
    double  Getx() const   { return m_Xos; }
    double  Gety() const   { return m_Yos; }
    double  Getz() const   { return m_Zos; }

    void    ShiftY( double factor )   { m_Yos -= factor; m_Yos *= -1.0; }
    BOOL    ConstructJx( CCalibrationData&, CObjectSpace*, double**, double**, UINT, UINT* );
    void    ConstructJy( CCalibrationData&, CObjectSpace*, double**, double**, UINT );
    BOOL    VerifyOrthonormality( double** );
};

#endif
```

File: FrameBuffer.h
Purpose: Defines the CFrameBuffer class.

```
#if !defined( FRAMEBUFFER_H )
#define FRAMEBUFFER_H

#include "Calibration.h"
#include "ObjectSpace.h"

class CFrameBuffer : public CCalibrationData
{
private:
    UINT    m_Pt;
    double  m_xfL,
            m_yfL,
            m_xfR,
            m_yfR;

public:
    CFrameBuffer():  m_Pt( 0 ),
                     m_xfL( 0.0 ),
                     m_yfL( 0.0 ),
                     m_xfR( 0.0 ),
                     m_yfR( 0.0 )
    {
    }

    ~CFrameBuffer()  { }

    void SetFrmCoords( double yfR, double xfR, double yfL, double xfL, UINT Pt )
    {
        m_Pt = Pt;
        m_xfL = xfL;
        m_yfL = yfL;
        m_xfR = xfR;
```

```
        m_yfR = yfR;
    }

    UINT    Getp()   const    { return m_Pt; }
    double  Getxl()  const    { return m_xfL; }
    double  Getyl()  const    { return m_yfL; }
    double  Getxr()  const    { return m_xfR; }
    double  Getyr()  const    { return m_yfR; }

    void Swap( double txfR, double tyfR, double a, double b, UINT Pt )
    {
        m_Pt = Pt;
        m_xfL = txfR;
        m_yfL = tyfR;
        m_xfR = a;
        m_yfR = b;
    }

    void CalcFrmBufferCoords( CCalibrationData&,
                              double*,
                              double**,
                              CObjectSpace*,
                              UINT,
                              UINT* );

    void ConstructF( double*, double*, CFrameBuffer*, UINT );
};

#endif
```

File: Iintersection.h
Purpose: Defines the CIntersection class.

```
#if !defined( INTERSECTION_H )
#define INTERSECTION_H

#include "stdafx.h"
#include <math.h>
#include "Definitions.h"
#include "Operations.h"
#include "calibration.h"

typedef struct _FRMBUFFER     {
    double xfL, yfL, xfR, yfR;
} FRMBUFFER;

typedef struct _OBJSPACE   {
    double Xos, Yos, Zos;
} OBJSPACE;

typedef struct _CND           {
    UINT znL, znR;                    // zone information
} CND;

typedef struct _UCR           {
    double dXpos, dYpos, dZpos;       // uncertainties
```

```
    double dXneg, dYneg, dZneg;
    double dR;
    double dA;
    double reXfL, reYfL, reXfR, reYfR;      // re-projected coordinates
} UCR;


class CIntersection : public CObject
{
private:
    FRMBUFFER       m_FrmBuffer;
    OBJSPACE        m_ObjSpace;
    CND             m_Cnd;

public:
    CIntersection() {}
    ~CIntersection() {}

    void    SetFrm( double, double, double, double );
    void    SetObj( double, double, double );
    void    SetCnd( UINT, UINT );

    FRMBUFFER       GetFrm() const      { return m_FrmBuffer; }
    OBJSPACE        GetObj() const      { return m_ObjSpace; }
    CND             GetCnd() const      { return m_Cnd; }

    int   ImproveIt(    CCalibrationData&,
                        CCalibrationData&,
                        double**,
                        double**,
                        OBJSPACE*,
                        CND* );

    UCR Uncertainty( CCalibrationData&,
                        CCalibrationData&,
                        double**,
                        double**,
                        OBJSPACE*,
                        CND* );
};

#endif
```

File: Operations.h

Purpose: Defines the COperations class. Refer to the book "Numerical Recipes in C: the Art of Scientific Computing" for the function definititions.

```
#if !defined( OPERATIONS_H )
#define OPERATIONS_H

#include  "stdafx.h"
#include  <math.h>
#include  "Definitions.h"

class COperations
{
```

```
public:
    COperations()     { }
    ~COperations()    { }

    double      DPythag( double, double );
    double**    DMatrix( UINT, UINT, UINT, UINT );
    void        FreeDMatrix( double**, UINT, UINT );
    void        OperationError( char* error )
    {
        AfxMessageBox( error, MB_ICONSTOP );
    }
    void        SvdBksb( double**, double*, double**, int, int, double*, double* );
    void        SvdDcmp( double**, int, int, double*, double** );
    void        Svdvar( double**, int, double*, double** );
};

#endif
```

File: Definitions.h

```
// Constants and macros

#if !defined( DEFINITIONS_H )
#define DEFINITIONS_H

#define     IMGC        254.0       // Y axis image centre
#define     APT         13e-6       // Photosite aperture
#define     CONVLIMIT   1e-13       // Convergence limit
#define     DIVLIMIT    1e5         // Divergence limit
#define     UNKS        8           // No of calibration unknowns
#define     ELMS        3           // No of unknowns in rotation matrices
#define     MAXITS      99          // Max number of iterations
#define     MAXPOINTS   100         // Maximum number of control points
#define     YSHIFT      0.13        // Object space Y axis shift

#define     PI          3.14159265358979323846264338327 95

#define     CONVERGED   0
#define     DIVERGED    -1
#define     CONTINUE    1
#define     OK          0
#define     END         1
#define     SQR(x)      ((x) * (x))

#define     iUNKS       3
#define     iEQUS       4
#define     N           3048        // Number of lines grabbed
#define     iITER       29          // Maximum number of iterations
#define     iDIV_LIMIT  100.0       // Divergence threshold
#define     iCONV_LIMIT 1e-12       // Convergence threshold
#define     ITER1       1e8         // Max quantisation error calc iterations
#define     MAXRADIUS   0.2         // Max intersection radius

static double dsqrarg;
#define DSQR(a) ((dsqrarg=(a)) == 0.0 ? 0.0 : dsqrarg*dsqrarg)
```

```
static double dmaxarg1, dmaxarg2;
#define DMAX(a,b) (dmaxarg1=(a), dmaxarg2=(b), (dmaxarg1) > (dmaxarg2) ?\
        (dmaxarg1) : (dmaxarg2))

static int iminarg1, iminarg2;
#define IMIN(a,b) (iminarg1=(a), iminarg2=(b), (iminarg1) < (iminarg2) ?\
        (iminarg1) : (iminarg2))

#define SIGN(a,b) ((b) >= 0.0 ? fabs(a) : -fabs(a))

#endif
```

File: Calibrate1Doc.cpp

```
// calibrate1Doc.cpp : implementation of the CCalibrate1Doc class
//

#include "calibrate1Doc.h"
#include "calibrate1View.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CCalibrate1Doc

IMPLEMENT_DYNCREATE(CCalibrate1Doc, CDocument)

BEGIN_MESSAGE_MAP(CCalibrate1Doc, CDocument)
    //{{AFX_MSG_MAP(CCalibrate1Doc)
    ON_COMMAND(ID_CALIBRATION_STARTCALIBRATION, OnCalibrationStart)
    ON_UPDATE_COMMAND_UI(ID_CALIBRATION_STARTCALIBRATION, OnUpdateCalibrationStart)
    ON_COMMAND(ID_CALIBRATION_OPEN, OnCalibrationOpen)
    ON_COMMAND(ID_INTERSECTION_START, OnIntersectionStart)
    ON_UPDATE_COMMAND_UI(ID_INTERSECTION_START, OnUpdateIntersectionStart)
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CCalibrate1Doc construction/destruction

CCalibrate1Doc::CCalibrate1Doc()
{
}


CCalibrate1Doc::~CCalibrate1Doc()
{
}


BOOL CCalibrate1Doc::OnNewDocument()
{
```

```
    if (!CDocument::OnNewDocument())
        return FALSE;

    m_bFlagL = m_bFlagR = m_bAllocFlag = FALSE;
    m_nTotPts = m_nItL = m_nItR = m_nSngL = m_nSngR = 0;
    m_dNormL = m_dNormR = 0.0;
    m_dSthL = m_dSthR = 1.0e-12;

    return TRUE;
}
/////////////////////////////////////////////////////////////


void CCalibrate1Doc::OnCloseDocument()
{
    // Release objects, vectors etc.
    if( m_bAllocFlag )
    {
        delete    m_pInitCalibL;
        delete    m_pInitCalibR;
        delete    m_pCalibDataL;
        delete    m_pCalibDataR;
        delete    m_pCalibResL;
        delete    m_pCalibResR;

        m_pOperations->FreeDMatrix( m_ppRL, 1, 1 );
        m_pOperations->FreeDMatrix( m_ppRR, 1, 1 );
        m_pOperations->FreeDMatrix( m_ppJ, 1, 1 );
        m_pOperations->FreeDMatrix( m_ppV, 1, 1 );
        m_pOperations->FreeDMatrix( m_ppCvmL, 1, 1 );
        m_pOperations->FreeDMatrix( m_ppCvmR, 1, 1 );
        delete m_pOperations;

        delete []  m_pnvZoneL;
        delete []  m_pnvZoneR;
        delete []  m_pObjectSpace;
        delete []  m_pFrameBuffer;
        delete []  m_pFrameBufferS;
        delete []  m_pdvW;
        delete []  m_pdvF;
        delete []  m_pdvX;
        delete []  m_pdvFrmCalc;
    }
    CDocument::OnCloseDocument();
}

/////////////////////////////////////////////////////////////////////////////
// CCalibrate1Doc serialization

void CCalibrate1Doc::Serialize(CArchive& ar)
{
    m_pCalibDataL->Serialize( ar );
    m_pCalibDataR->Serialize( ar );
}

/////////////////////////////////////////////////////////////////////////////
// CCalibrate1Doc diagnostics
```

```
#ifdef _DEBUG
void CCalibrate1Doc::AssertValid() const
{
    CDocument::AssertValid();
}


void CCalibrate1Doc::Dump(CDumpContext& dc) const
{
    CDocument::Dump(dc);
}
#endif //_DEBUG

//////////////////////////////////////////////////////////////////
// CCalibrate1Doc commands


void CCalibrate1Doc::OnCalibrationStart()
{
    BeginWaitCursor();

    m_nSngL = m_nSngR = m_nItL = m_nItR = 0;
    m_dNormL = m_dNormR = 0.0;
    m_bFlagL = m_bFlagR = FALSE;
    BOOL bOSFlag = TRUE;
    int nCheckL, nCheckR;

    // Initial estimates:
    m_pCalibDataL->Seturtip( m_pInitCalibL->Geturtip() );
    m_pCalibDataL->Setomega( m_pInitCalibL->Getomega() );
    m_pCalibDataL->Setphi( m_pInitCalibL->Getphi() );
    m_pCalibDataL->Setkappa( m_pInitCalibL->Getkappa() );
    m_pCalibDataL->Setf( m_pInitCalibL->Getf() );
    m_pCalibDataL->Settx( m_pInitCalibL->Gettx() );
    m_pCalibDataL->Setty( m_pInitCalibL->Getty() );
    m_pCalibDataL->Settz( m_pInitCalibL->Gettz() );

    m_pCalibDataR->Seturtip( m_pInitCalibR->Geturtip() );
    m_pCalibDataR->Setomega( m_pInitCalibR->Getomega() );
    m_pCalibDataR->Setphi( m_pInitCalibR->Getphi() );
    m_pCalibDataR->Setkappa( m_pInitCalibR->Getkappa() );
    m_pCalibDataR->Setf( m_pInitCalibR->Getf() );
    m_pCalibDataR->Settx( m_pInitCalibR->Gettx() );
    m_pCalibDataR->Setty( m_pInitCalibR->Getty() );
    m_pCalibDataR->Settz( m_pInitCalibR->Gettz() );

    ClearMatrices();
    UINT nRow, nCol;
    for( nRow = 1; nRow <= UNKS; nRow++ )
        for( nCol = 1; nCol <= UNKS; nCol++ )
            m_ppCvmL[nRow][nCol] = m_ppCvmR[nRow][nCol] = 0.0;

    for( nRow = 1; nRow <= ELMS; nRow++ )
        for( nCol = 1; nCol <= ELMS; nCol++ )
            m_ppRL[nRow][nCol] = m_ppRR[nRow][nCol] = 0.0;

    ////////////////////////////////////////////////////
    //    Calibrate left camera:
```

```
do
{
    m_nItL++;

    //    Evaluate rotation matrix:
    m_pCalibDataL->ConstructRotationMatrix( *m_pCalibDataL, m_ppRL );

    //    Calculate current X-axis partial derivatives:
    bOSFlag = m_pObjectSpace->ConstructJx( *m_pCalibDataL, m_pObjectSpace,
                        m_ppJ, m_ppRL, m_nTotPts, m_pnvZoneL );

    if( !bOSFlag ) {    // bad news
        AfxMessageBox("  Diverged from CObjectSpace - L  ", MB_ICONSTOP );
        break;
    }

    //    Calculate current Y-axis partial derivatives:
    m_pObjectSpace->ConstructJy( *m_pCalibDataL, m_pObjectSpace, m_ppJ, m_ppRL, m_nTotPts );

    m_pFrameBuffer->CalcFrmBufferCoords( *m_pCalibDataL,
        m_pdvFrmCalc, m_ppRL, m_pObjectSpace, m_nTotPts, m_pnvZoneL );

    //    Construct function F (to be minimised):
    m_pFrameBuffer->ConstructF( m_pdvF, m_pdvFrmCalc, m_pFrameBuffer, m_nTotPts );

    //    Singular Value Decomposition & Backsubstitution:
    m_pOperations->SvdDcmp( m_ppJ, m_nTotPts * 2, UNKS, m_pdvW, m_ppV );

    double dWmax = 0.0;
    for( UINT nCntr = 1; nCntr <= UNKS; nCntr++ )
        if( m_pdvW[nCntr] > dWmax ) dWmax = m_pdvW[nCntr];

    double dWmin = dWmax * m_dSthL;

    //    Discard offending equations:
    for( nCntr = 1; nCntr <= UNKS; nCntr++ )
        if( m_pdvW[nCntr] < dWmin ) {
        m_pdvW[nCntr] = 0.0;
        m_nSngL++;
    }

    m_pOperations->SvdBksb( m_ppJ, m_pdvW, m_ppV, m_nTotPts * 2, UNKS, m_pdvF, m_pdvX );

    //    Process monitor:
    nCheckL = m_pCalibDataL->CorrectAndTest( *m_pCalibDataL, m_pdvX, m_nItL );

    if( nCheckL == DIVERGED )      // acted funny
    {
        MessageBeep( DWORD(-1) );
        AfxMessageBox( "  Controlled divergence in left camera.\n\n  Process failed.\t",
            MB_OK | MB_ICONSTOP );
    }
}
while( nCheckL != DIVERGED && nCheckL != CONVERGED && m_nItL <= MAXITS );
// End of Newton-Raphson loop

//    Orthonormality flag:
```

```
m_bFlagL = m_pObjectSpace->VerifyOrthonormality( m_ppRL );

m_pOperations->Svdvar( m_ppV, UNKS, m_pdvW, m_ppCvmL );

if( m_nItL == MAXITS + 1 )
    AfxMessageBox( "  WARNING: Left camera process stopped artificially...\t", MB_ICONSTOP );

if(   m_nSngL > 0 )      {
    m_nSngL /= m_nItL;
    CString sLs;
    sLs.Format( "%d singular value(s) detected...(L)", m_nSngL );
    AfxMessageBox( sLs, MB_ICONSTOP );
}

//    Calculate norm of residual vector:
for( UINT nInd = 1; nInd <= UNKS; nInd++ )
    m_dNormL += DSQR( m_pdvX[nInd] );

m_dNormL = sqrt( m_dNormL );
m_sNormL.Format( "%.3e", m_dNormL );
m_sSngL.Format( "%d", m_nSngL );

m_pCalibResL->Seturtip( m_pCalibDataL->Geturtip() );
m_pCalibResL->Setomega( m_pCalibDataL->Getomega() );
m_pCalibResL->Setphi( m_pCalibDataL->Getphi() );
m_pCalibResL->Setkappa( m_pCalibDataL->Getkappa() );
m_pCalibResL->Setf( m_pCalibDataL->Getf() );
m_pCalibResL->Settx( m_pCalibDataL->Gettx() );
m_pCalibResL->Setty( m_pCalibDataL->Getty() );
m_pCalibResL->Settz( m_pCalibDataL->Gettz() );

////////////////////////////////////////////////////////
//    Calibrate right camera:

for( UINT nPt = 1; nPt <= m_nTotPts; nPt++ )
    m_pFrameBufferS[nPt].Swap(    m_pFrameBuffer[nPt].Getxr(),
                                  m_pFrameBuffer[nPt].Getyr(),
                                  0.0, 0.0,   // don't care
                                  m_pFrameBuffer[nPt].Getp() );

ClearMatrices();

do
{
    m_nItR++;

    //    Evaluate rotation matrix:
    m_pCalibDataR->ConstructRotationMatrix( *m_pCalibDataR, m_ppRR );

    //    Calculate current X-axis partial derivatives:
    bOSFlag = m_pObjectSpace->ConstructJx( *m_pCalibDataR, m_pObjectSpace,
                        m_ppJ, m_ppRR, m_nTotPts, m_pnvZoneR );

    if( !bOSFlag ) {
        AfxMessageBox("  Diverged from CObjectSpace - R  ", MB_ICONSTOP );
        break;
    }
```

```
        //    Calculate current Y-axis partial derivatives:
        m_pObjectSpace->ConstructJy( *m_pCalibDataR, m_pObjectSpace, m_ppJ, m_ppRR, m_nTotPts );

        m_pFrameBufferS->CalcFrmBufferCoords( *m_pCalibDataR,
              m_pdvFrmCalc, m_ppRR, m_pObjectSpace, m_nTotPts, m_pnvZoneR );

        //    Construct function F (to be minimised):
        m_pFrameBufferS->ConstructF( m_pdvF, m_pdvFrmCalc, m_pFrameBufferS, m_nTotPts );

        //    Singular Value Decomposition & Backsubstitution:
        m_pOperations->SvdDcmp( m_ppJ, m_nTotPts * 2, UNKS, m_pdvW, m_ppV );

        double dWmax = 0.0;
        for( UINT nCntr = 1; nCntr <= UNKS; nCntr++ )
              if( m_pdvW[nCntr] > dWmax ) dWmax = m_pdvW[nCntr];

        double dWmin = dWmax * m_dSthR;

        //    Discard offending equations:
        for( nCntr = 1; nCntr <= UNKS; nCntr++ )
              if( m_pdvW[nCntr] < dWmin ) {
              m_pdvW[nCntr] = 0.0;
              m_nSngR++;
        }

        m_pOperations->SvdBksb( m_ppJ, m_pdvW, m_ppV, m_nTotPts * 2, UNKS, m_pdvF, m_pdvX );

        //    Process monitor:
        nCheckR = m_pCalibDataR->CorrectAndTest( *m_pCalibDataR, m_pdvX, m_nItR );

        if( nCheckR == DIVERGED )      // acted funny
        {
              MessageBeep( DWORD(-1) );
              AfxMessageBox( "  Controlled divergence in right camera.\n\n  Process failed.\t",
                    MB_OK | MB_ICONSTOP );
        }
  }
  while( nCheckR != DIVERGED && nCheckR != CONVERGED && m_nItR <= MAXITS );

  // End of Newton-Raphson loop
  ////////////////////////////////////////////////////


  //    Orthonormality flag:
  m_bFlagR = m_pObjectSpace->VerifyOrthonormality( m_ppRR );

  m_pOperations->Svdvar( m_ppV, UNKS, m_pdvW, m_ppCvmR );

  if( m_nItR == MAXITS + 1 )
        AfxMessageBox( "  WARNING: Right camera process stopped artificially...\t", MB_ICONSTOP );

  if(   m_nSngR > 0 )      {
        m_nSngR /= m_nItR;
        CString sRs;
        sRs.Format( "%d singular value(s) detected...(R)", m_nSngR );
        AfxMessageBox( sRs, MB_ICONSTOP );
  }
```

```
    //    Residual vector's norm:
    for( nInd = 1; nInd <= UNKS; nInd++ )
        m_dNormR += DSQR( m_pdvX[nInd] );

    m_dNormR = sqrt( m_dNormR );
    m_sNormR.Format( "%.3e", m_dNormR );
    m_sSngR.Format( "%d", m_nSngR );

    m_pCalibResR->Seturtip( m_pCalibDataR->Geturtip() );
    m_pCalibResR->Setomega( m_pCalibDataR->Getomega() );
    m_pCalibResR->Setphi( m_pCalibDataR->Getphi() );
    m_pCalibResR->Setkappa( m_pCalibDataR->Getkappa() );
    m_pCalibResR->Setf( m_pCalibDataR->Getf() );
    m_pCalibResR->Settx( m_pCalibDataR->Gettx() );
    m_pCalibResR->Setty( m_pCalibDataR->Getty() );
    m_pCalibResR->Settz( m_pCalibDataR->Gettz() );

    EndWaitCursor();

    int nCom = IDNO;
    if( nCheckL == CONVERGED && nCheckR == CONVERGED && !m_nSngL && !m_nSngR )
    {
        nCom = AfxMessageBox( "  Process finished.\n\n  Would you like to view the calibration results? ",
            MB_YESNO | MB_ICONEXCLAMATION );
    }
    else
    {
        nCom = AfxMessageBox(
            "  Calibrate detected problems in the calibration procedure.\n\n  Would you like to view the
calibration results anyway? ",
            MB_YESNO | MB_ICONQUESTION );
    }

    if( nCom == IDYES )    {
        CFrameWnd* pFrameWnd = STATIC_DOWNCAST(CFrameWnd, AfxGetMainWnd());
        CCalibrate1View* pView = (CCalibrate1View*)pFrameWnd->GetActiveFrame()-
>GetActiveView();
        pView->DisplayPropSheet();
    }
}


BOOL CCalibrate1Doc::ClearMatrices()
{
    UINT nRow, nCol;
    for( nRow = 1; nRow <= m_nTotPts * 2; nRow++ )  {
        m_pdvF[nRow] = m_pdvFrmCalc[nRow] = 0.0;
        for( nCol = 1; nCol <= UNKS; nCol++ )
            m_ppJ[nRow][nCol] = 0.0;
    }

    for( nRow = 1; nRow <= UNKS; nRow++ )
        for( nCol = 1; nCol <= UNKS; nCol++ )
            m_ppV[nRow][nCol] = 0.0;

    for( nRow = 1; nRow <= UNKS; nRow++ )
        m_pdvX[nRow] = m_pdvW[nRow] = 0.0;
```

```
    return TRUE;
}


void CCalibrate1Doc::OnUpdateCalibrationStart(CCmdUI* pCmdUI)
{
    pCmdUI->SetCheck( FALSE );
    pCmdUI->Enable( (BOOL)m_nTotPts );
}


void CCalibrate1Doc::OnCalibrationOpen()
{
    // Begin the begin:
    m_pInitCalibL = new CCalibrationData(   0.02,     // Urtip
                                            0.0,      // omega
                                            0.0,      // phi
                                            0.0,      // kappa
                                            0.050,    // f
                                            0.015,    // tx
                                            0.05,     // ty
                                            1.0 );    // tz

    m_pInitCalibR = new CCalibrationData(   0.02,     // Urtip
                                            0.0,      // omega
                                            0.0,      // phi
                                            0.0,      // kappa
                                            0.050,    // f
                                            -0.015,   // tx
                                            0.05,     // ty
                                            1.0 );    // tz

    m_pCalibDataL = new CCalibrationData;
    m_pCalibDataR = new CCalibrationData;

    m_pCalibResL = new CCalibrationData;
    m_pCalibResR = new CCalibrationData;

    double* m_pdvData = new double[ MAXPOINTS * 9 + 9 ];

    m_nCP = 0;
    m_nCP = m_pCalibDataL->LoadCorrData( m_pdvData );
    m_nTotPts = m_nCP;

    CInputDataDialog InDlg;
    UINT nZL, nZR;

    InDlg.m_nZL = InDlg.m_nZR = 36;     // zone #2 after this

    if( InDlg.DoModal() == IDOK )   {
        nZL = InDlg.m_nZL;
        nZR = InDlg.m_nZR;
    }
    else return;

    m_pnvZoneL = new UINT[ m_nTotPts + 1 ];
    m_pnvZoneR = new UINT[ m_nTotPts + 1 ];
```

```
//    Zone conditions
for( UINT nPt = 1; nPt <= m_nTotPts; nPt++ ) {
      m_pnvZoneL[nPt] = m_pnvZoneR[nPt] = 1;      // default

      if( nPt >= nZL )     m_pnvZoneL[nPt] = 2;
      if( nPt >= nZR )     m_pnvZoneR[nPt] = 2;
}

//////////////////////////////////////////////////
// Allocate objects on the heap:

m_pObjectSpace = new CObjectSpace[ m_nTotPts + 1 ];
m_pFrameBuffer = new CFrameBuffer[ m_nTotPts + 1 ];
m_pFrameBufferS = new CFrameBuffer[ m_nTotPts + 1 ];
m_pOperations = new COperations;

m_ppRL = m_pOperations->DMatrix( 1, ELMS, 1, ELMS );
m_ppRR = m_pOperations->DMatrix( 1, ELMS, 1, ELMS );
m_ppCvmL = m_pOperations->DMatrix( 1, UNKS, 1, UNKS );
m_ppCvmR = m_pOperations->DMatrix( 1, UNKS, 1, UNKS );
m_ppJ = m_pOperations->DMatrix( 1, m_nTotPts * 2, 1, UNKS );

m_pdvFrmCalc = new double[ m_nTotPts * 2 + 1 ];
m_pdvX = new double[ UNKS + 1 ];
m_pdvW = new double[ UNKS + 1 ];
m_pdvF = new double[ m_nTotPts * 2 + 1 ];
m_ppV =  m_pOperations->DMatrix( 1, UNKS, 1, UNKS );

m_bAllocFlag = TRUE; // allocation flag

for( UINT nRow = 1; nRow <= ELMS; nRow++ )
      for( UINT nCol = 1; nCol <= ELMS; nCol++ )
            m_ppRL[nRow][nCol] = m_ppRR[nRow][nCol] = 0.0;

if( m_nCP )
{
      UINT nAddElm = 0, nPt;

      for( nPt = 1; nPt <= m_nTotPts; nPt++ )
            m_pObjectSpace[nPt].SetObjCoords(
                        m_pdvData[ ++nAddElm ],
                        m_pdvData[ ++nAddElm ],
                        m_pdvData[ ++nAddElm ],
                        UINT( m_pdvData[ ++nAddElm ] ) );

      nAddElm = 0; // Reset counter

      for( nPt = 1; nPt <= m_nTotPts; nPt++ )
            m_pFrameBuffer[nPt].SetFrmCoords(
                        m_pdvData[ ++nAddElm + m_nTotPts * 4 ],
                        m_pdvData[ ++nAddElm + m_nTotPts * 4 ],
                        m_pdvData[ ++nAddElm + m_nTotPts * 4 ],
                        m_pdvData[ ++nAddElm + m_nTotPts * 4 ],
                        UINT( m_pdvData[ ++nAddElm + m_nTotPts * 4 ] ) );

      for( nPt = 1; nPt <= m_nTotPts; nPt++ )
            m_pObjectSpace[nPt].ShiftY( YSHIFT );     // translate origin 130mm
}
```

```
    delete [] m_pdvData;

    CString Message = "The Control Correspondence was successfully loaded.";
    Message += "\n\nThe calibration procedure may now be invoked.";
    AfxMessageBox( Message, MB_ICONINFORMATION );
}


void CCalibrate1Doc::OnIntersectionStart()
{
    m_pInter = new CInterseetion;

    CIntersectCorr InterDialog;
    if( InterDialog.DoModal() == IDOK )
        m_pInter->SetFrm(     InterDialog.m_dInterXL,
                              InterDialog.m_dInterYL,
                              InterDialog.m_dInterXR,
                              InterDialog.m_dInterYR );

    // ------------------------------------------------------------
    //  Estimate the unknown point's radius, angle and zone conditions:

    // local bits and pieces (radius estimate)
    double sxL, sxR, xi, dMinRad;

    sxL = 30.0 / ( PI * m_pCalibResL->Geturtip() );
    sxR = 30.0 / ( PI * m_pCalibResR->Geturtip() );

    xi = m_pInter->GetFrm().xfL / sxL - m_pInter->GetFrm().xfR / sxR;

    if( fabs( m_pCalibResL->Gettx() ) > fabs( m_pCalibResR->Gettx() ) )
        dMinRad = fabs( m_pCalibResL->Gettx() );
    else
        dMinRad = fabs( m_pCalibResR->Gettx() );

    for( m_dAppRadius = dMinRad; m_dAppRadius < MAXRADIUS + 1e-3; m_dAppRadius += 1e-6 )
    {
        if( fabs( fabs( xi - fabs( m_pCalibResL->Getphi() ) -
            m_pCalibResR->Getphi() ) -
            asin( fabs( m_pCalibResL->Gettx() ) / m_dAppRadius ) -
            asin( fabs( m_pCalibResR->Gettx() ) / m_dAppRadius ) ) < 1e-5 )
                break;
    }

    if( m_dAppRadius > MAXRADIUS )    // Struggling - relax criterion
    {
        for( double rdn = dMinRad; rdn < MAXRADIUS + 1e-3; rdn += 1e-6 )
        {
            if( fabs( fabs( xi - fabs( m_pCalibResL->Getphi() ) -
                m_pCalibResR->Getphi() ) -
                asin( fabs( m_pCalibResL->Gettx() ) / rdn ) -
                asin( fabs( m_pCalibResR->Gettx() ) / rdn ) ) < 1e-4 )
                    break;
        }
        m_dAppRadius = rdn;
        AfxMessageBox( " Radius convergence criterion has been relaxed. ", MB_ICONSTOP );
    }
```

```
if( m_dAppRadius > MAXRADIUS ) {
    AfxMessageBox( " Could not determine radius - input data are unreliable.\n\n Aborting process. ",
        MB_ICONSTOP );
    return;
}

CND     zone;    // zone conditions
OBJSPACE apxOS;  // to hold approximate solution

// local bits and pieces (zone condition estimates)
double wRefL, wRefR, wRef, wZoneL, wZoneR;

wRefL = ( m_pCalibResL->Getphi() + m_pCalibResR->Getphi() ) / 2.0 -
        fabs( ( m_pCalibResL->Getphi() - m_pCalibResR->Getphi() ) / 2.0 ) +
        m_pInter->GetFrm().xfL / sxL + asin( m_pCalibResL->Gettx() / m_dAppRadius );

wRefR = ( m_pCalibResL->Getphi() + m_pCalibResR->Getphi() ) / 2.0 +
        fabs( ( m_pCalibResL->Getphi() - m_pCalibResR->Getphi() ) / 2.0 ) +
        m_pInter->GetFrm().xfR / sxR + asin( m_pCalibResR->Gettx() / m_dAppRadius );

wRef = 0.5 * ( wRefL + wRefR );

wZoneL = wRef - m_pCalibResL->Getphi();
wZoneR = wRef - m_pCalibResR->Getphi();

apxOS.Xos = m_dAppRadius * sin( wRef );  // Cartesian solution rotated
apxOS.Zos = m_dAppRadius * cos( wRef );  // by 90 degrees

// local bits and pieces (Y coordinate estimate)
double yDenL, yCamL, yWldL, yDenR, yCamR, yWldR;

yDenL = m_pCalibResL->Gettz() - sqrt( SQR( m_dAppRadius ) - SQR( m_pCalibResL->Gettx() ) );

yCamL = yDenL * ( APT / m_pCalibResL->Getf() ) * ( m_pInter->GetFrm().yfL - IMGC );

yWldL = ( yCamL + m_pCalibResL->Getty() -
          m_ppRL[2][1] * apxOS.Xos - m_ppRL[2][3] * apxOS.Zos ) / m_ppRL[2][2];

yDenR = m_pCalibResR->Gettz() - sqrt( SQR( m_dAppRadius ) - SQR( m_pCalibResR->Gettx() ) );

yCamR = yDenR * ( APT / m_pCalibResR->Getf() ) * ( m_pInter->GetFrm().yfR - IMGC );

yWldR = ( yCamR + m_pCalibResR->Getty() -
          m_ppRR[2][1] * apxOS.Xos - m_ppRR[2][3] * apxOS.Zos ) / m_ppRR[2][2];

apxOS.Yos = 0.5 * ( yWldL + yWldR );

if( wZoneL > 0 && wZoneL < PI ) zone.znL = 1;
else zone.znL = 2;

if( wZoneL > 2 * PI )    {
    AfxMessageBox( " Point exceeds stereo FOV... (+ve)\n\nAborting process. ", MB_ICONSTOP );
    return;
}

if( wZoneR < PI ) zone.znR = 1;   else zone.znR = 2;
```

```
    CInterDiagnostics DiagnDialog;

    DiagnDialog.m_dLAngle     = RadToDeg( wRefL );
    DiagnDialog.m_dRAngle     = RadToDeg( wRefR );
    DiagnDialog.m_dLAngleZn   = RadToDeg( wZoneL );
    DiagnDialog.m_dRAngleZn   = RadToDeg( wZoneR );
    DiagnDialog.m_dRadius     = 1e3 * m_dAppRadius;
    DiagnDialog.m_dX          = 1e3 * apxOS.Xos;
    DiagnDialog.m_dY          = 1e3 * ( YSHIFT - apxOS.Yos );
    DiagnDialog.m_dZ          = 1e3 * apxOS.Zos;
    DiagnDialog.m_nLZone      = zone.znL;
    DiagnDialog.m_nRZone      = zone.znR;

    DiagnDialog.DoModal();

    //-------------------------------------------------------------
//  zone.znL = 1; // MANUAL OVERRIDE OF ZONE CONDITIONS (DEBUG)
//  zone.znR = 1;
    //-------------------------------------------------------------

    OBJSPACE    imprOS = apxOS;  // preserve approximate solution

    //    Attempt non-linear improvement of approximate solution:
    int bIFlag = m_pInter->ImproveIt(     *m_pCalibResL, *m_pCalibResR,
                                          m_ppRL, m_ppRR,
                                          &imprOS,
                                          &zone );

    if( bIFlag != DIVERGED )
    {
        //    a simple test:
        double dTstRadius = sqrt( SQR( imprOS.Xos ) + SQR( imprOS.Zos ) );

        if( ( ( dTstRadius / m_dAppRadius ) > 1.1 || ( dTstRadius / m_dAppRadius ) < 0.9 ) )    {
            AfxMessageBox( " Singular Value Decomposition fails - input data may be unreliable. ",
                MB_ICONSTOP );
            m_finOS = apxOS;
        }
        else m_finOS = imprOS;      // all is OK
    }
    else {
        AfxMessageBox( " SVD diverged. ", MB_ICONSTOP );
        m_finOS = apxOS;
    }

    m_ucr = m_pInter->Uncertainty( *m_pCalibResL, *m_pCalibResR,
                                   m_ppRL, m_ppRR,
                                   &m_finOS,
                                   &zone );

    m_finOS.Yos = YSHIFT - m_finOS.Yos;

    CFrameWnd* pFrameWnd = STATIC_DOWNCAST(CFrameWnd, AfxGetMainWnd());
    CCalibrate1View* pView = (CCalibrate1View*)pFrameWnd->GetActiveFrame()->GetActiveView();
    pView->IntersectionSolution();

    delete m_pInter;
}
```

```
void CCalibrate1Doc::OnUpdateIntersectionStart(CCmdUI* pCmdUI)
{
    pCmdUI->SetCheck( FALSE );
    pCmdUI->Enable( (BOOL)m_nItL );
}
```

File: Calibration.cpp

```
#include "calibration.h"

void CCalibrationData::Serialize( CArchive& archive )
{
    CObject::Serialize(archive);

    if( archive.IsStoring() )
    {
        archive << m_urtip;
        archive << m_omega;
        archive << m_phi;
        archive << m_kappa;
        archive << m_f;
        archive << m_tx;
        archive << m_ty;
        archive << m_tz;
    }
    else
    {
        archive >> m_urtip;
        archive >> m_omega;
        archive >> m_phi;
        archive >> m_kappa;
        archive >> m_f;
        archive >> m_tx;
        archive >> m_ty;
        archive >> m_tz;
    }
}

IMPLEMENT_SERIAL( CCalibrationData, CObject, 0 );


//    Construct the rotation matrix
void CCalibrationData::ConstructRotationMatrix( CCalibrationData& cd, double** ppR )
{
    ppR[1][1] = cos( cd.m_phi ) * cos( cd.m_kappa );
    ppR[1][2] = sin( cd.m_omega ) * sin( cd.m_phi ) * cos( cd.m_kappa ) +
                cos( cd.m_omega ) * sin( cd.m_kappa );
    ppR[1][3] = -cos( cd.m_omega ) * sin( cd.m_phi ) * cos( cd.m_kappa ) +
                sin( cd.m_omega ) * sin( cd.m_kappa );

    ppR[2][1] = -cos( cd.m_phi ) * sin( cd.m_kappa );
    ppR[2][2] = -sin( cd.m_omega ) * sin( cd.m_phi ) * sin( cd.m_kappa ) +
                cos( cd.m_omega ) * cos( cd.m_kappa );
    ppR[2][3] = cos( cd.m_omega ) * sin( cd.m_phi ) * sin( cd.m_kappa ) +
```

```
                    sin( cd.m_omega ) * cos( cd.m_kappa );

    ppR[3][1] = sin( cd.m_phi );
    ppR[3][2] = -sin( cd.m_omega ) * cos( cd.m_phi );
    ppR[3][3] = cos( cd.m_omega ) * cos( cd.m_phi );
}


//  Correct initial approximations and check for convergence / divergence:
int CCalibrationData::CorrectAndTest( CCalibrationData& cd,
                                      double* pdvX,     // corrections' vector
                                      UINT iter )
{
    static double dConvLimit;
    if( iter == 1 ) dConvLimit = CONVLIMIT;  // Initialise conv_limit

//  Gradually relax initial convergence criterion:
    if( ( iter % 5 == 1 ) && ( iter > 15 ) )  dConvLimit *= 3.3;

    //  Add corrections
    cd.m_urtip += pdvX[1];
    cd.m_omega += pdvX[2];
    cd.m_phi   += pdvX[3];
    cd.m_kappa += pdvX[4];
    cd.m_f     += pdvX[5];
    cd.m_tx    += pdvX[6];
    cd.m_ty    += pdvX[7];
    cd.m_tz    += pdvX[8];  .

//  Check for convergence:
    double rs_sum = 0.0;
    rs_sum = sqrt(      DSQR( pdvX[1] / cd.m_urtip      ) +
                        DSQR( pdvX[2] / cd.m_omega ) +
                        DSQR( pdvX[3] / cd.m_phi  ) +
                        DSQR( pdvX[4] / cd.m_kappa ) +
                        DSQR( pdvX[5] / cd.m_f     ) +
                        DSQR( pdvX[6] / cd.m_tx    ) +
                        DSQR( pdvX[7] / cd.m_ty    ) +
                        DSQR( pdvX[8] / cd.m_tz    ) );

    if( rs_sum < dConvLimit )     return CONVERGED;

//  Check for divergence:
    for( UINT c = 1; c <= UNKS; c++ )
        if( pdvX[c] > DIVLIMIT )     {
            AfxMessageBox( "Normal divergence detected. ", MB_ICONSTOP );
            return DIVERGED;
        }

    return CONTINUE;       // Neither converged nor diverged - continue;
}


UINT CCalibrationData::LoadCorrData( double* pdData )
{
    CFile theFile;
```

```
    static char BASED_CODE szFilter[] = "Control Correspondence Files (*.dat)|*.dat|All Files
(*.*)|*.*||";

    CFileDialog dlg( TRUE, "dat", "*.dat", OFN_FILEMUSTEXIST | OFN_HIDEREADONLY, szFilter );

    if( dlg.DoModal() == IDOK )
        theFile.Open( dlg.GetPathName(), CFile::modeRead );
    else
        return 0;

    BeginWaitCursor();
    char* psBuf = new char[ MAXPOINTS * 9 + 10 ];
    char* psFin = new char[ MAXPOINTS * 9 + 10 ];
    psBuf[0] = psFin[0] = '\0';

    LONG lCount = 0L;
    BOOL bEnd = FALSE;
    UINT i = 0, l = 1;

    while( !bEnd )
    {
        do   // load number to buffer
        {
            do   // ignore irrelevant characters ( CR, LF, etc. )
            {
                theFile.Seek( lCount++, CFile::begin );
                theFile.Read( &psBuf[i++], 1 );
            }
            while( ( psBuf[i-1] < 44 ) || ( psBuf[i-1] > 57 ) );

            // if the character is numeric copy it:
            if( ( psBuf[i-1] >= 46 ) && ( psBuf[i-1] <= 57 ) )
                lstrcpy( psFin, psBuf );
        }
        while( psBuf[i-1] != COMMA  && (DWORD)lCount < theFile.GetLength() );

        // stop here or crash and burn:
        if( (DWORD)lCount == theFile.GetLength() )    bEnd = TRUE;

        psBuf[i] = psFin[i] = '\0'; // null terminate
        pdData[l] = atof( psFin );    // convert ascii to double

        for(UINT k=0; k<=i; k++)   psBuf[k] = psFin[k] = '\0';

        i = 0;      // reset pointer
        l++;        // next number
    }
    theFile.Close();

    delete [] psBuf;
    delete [] psFin;

    EndWaitCursor();
    return (UINT)pdData[l-5];   // return number of control points
}
```

File: Objectspace.cpp

```
#include "ObjectSpace.h"

// Adds X-axis Jacobian elements from the partial derivatives
BOOL CObjectSpace::ConstructJx(   CCalibrationData& cd, // the Calibration object
                                  CObjectSpace* pOs,    // the ObjectSpace object
                                  double** ppJ,         // the Jacobian matrix
                                  double** ppR,         // the Rotation matrix
                                  UINT pts,                  // number of control points
                                  UINT* pnvZone )       // Zone conditions
{
    double   *pdXc, *pdZc, *pdAlpha, *pdBeta, *pdGamma, *pdDelta, *pdEpsilon, *pdRd;
    UINT     *pnV;

    pdXc =       new double[pts+1];
    pdZc =       new double[pts+1];
    pdAlpha =    new double[pts+1];
    pdBeta =     new double[pts+1];
    pdGamma =    new double[pts+1];
    pdDelta =    new double[pts+1];
    pdEpsilon =  new double[pts+1];
    pdRd =       new double[pts+1];
    pnV =        new UINT[pts+1];

    for ( UINT nControl = 1; nControl <= pts; nControl++ )
    {
        pnV[nControl] = pOs[nControl].m_Pt + pOs[nControl-1].m_Pt;
        // pnV[nControl] points to odd-numbered rows

        pdXc[nControl] =  ppR[1][1] * pOs[nControl].Getx() +
                          ppR[1][2] * pOs[nControl].Gety() +
                          ppR[1][3] * pOs[nControl].Getz();

        pdZc[nControl] =  ppR[3][1] * pOs[nControl].Getx() +
                          ppR[3][2] * pOs[nControl].Gety() +
                          ppR[3][3] * pOs[nControl].Getz();

        pdRd[nControl] = sqrt( SQR( pdXc[nControl] ) + SQR( pdZc[nControl] ) );

        pdAlpha[nControl] = 30.0 / ( PI * cd.Geturtip() * pdRd[nControl] );

        pdBeta[nControl] = 1.0 / sqrt( 1.0 - SQR( pdZc[nControl] ) / SQR( pdRd[nControl] ) );

        if ( pnvZone[nControl] == 2 )      pdBeta[nControl] *= -1.0;

        if(   SQR( cd.Gettx() ) / SQR( pdRd[nControl] ) > 1.0 )
        {
            delete [] pdXc;
            delete [] pdZc;
            delete [] pdAlpha;
            delete [] pdBeta;
            delete [] pdGamma;
            delete [] pdDelta;
            delete [] pdEpsilon;
            delete [] pdRd;
            delete [] pnV;
            AfxMessageBox( "  Fatal Error in OBJSPC.\n\n  Aborting Calibration...", MB_ICONSTOP );
```

```
            return FALSE;
        }

        pdGamma[nControl] = cd.Gettx() / sqrt( 1.0 - SQR( cd.Gettx() ) / SQR( pdRd[nControl] ) );

        pdDelta[nControl] = ( pdGamma[nControl] + pdBeta[nControl] * pdZc[nControl] ) / SQR(
pdRd[nControl] );

        pdEpsilon[nControl] =  cos( cd.Getphi() ) * pOs[nControl].Getx() +
                               sin( cd.Getomega() ) * sin( cd.Getphi() ) * pOs[nControl].Gety() -
                               cos( cd.Getomega() ) * sin( cd.Getphi() ) * pOs[nControl].Getz();

//      wrt urtip:
        if ( pnvZone[nControl] == 1 )     {
            ppJ[ pnV[nControl] ][1] = - 30.0 / ( PI * SQR( cd.Geturtip() ) ) * (
                                      PI / 2.0 - asin( cd.Gettx() / pdRd[nControl] ) -
                                      asin( pdZc[nControl] / pdRd[nControl] ) );
        }
        else {
            ppJ[ pnV[nControl] ][1] = - 30.0 / ( PI * SQR( cd.Geturtip() ) ) * (
                                      3 * PI / 2.0 - asin( cd.Gettx() / pdRd[nControl] ) +
                                      asin( pdZc[nControl] / pdRd[nControl] ) );
        }

//      wrt omega:
        ppJ[ pnV[nControl] ][2] = pdAlpha[nControl] * ( - pdBeta[nControl] *
                ( - ppR[3][3] * pOs[nControl].Gety() + ppR[3][2] * pOs[nControl].Getz() ) +
                pdDelta[nControl] * (
                pdXc[nControl] * ( - ppR[1][3] * pOs[nControl].Gety() + ppR[1][2] *
pOs[nControl].Getz() ) +
                pdZc[nControl] * ( - ppR[3][3] * pOs[nControl].Gety() + ppR[3][2] *
pOs[nControl].Getz() ) ) );

//      wrt phi:
        ppJ[ pnV[nControl] ][3] = pdAlpha[nControl] * ( pdDelta[nControl] * pdZc[nControl] * (
                pdEpsilon[nControl] - pdXc[nControl] * cos( cd.Getkappa() ) ) -
                pdBeta[nControl] * pdEpsilon[nControl] );

//      wrt kappa:
        ppJ[ pnV[nControl] ][4] = pdAlpha[nControl] * pdDelta[nControl] *
                pdXc[nControl] * ( ppR[2][1] * pOs[nControl].Getx() +
                ppR[2][2] * pOs[nControl].Gety() + ppR[2][3] * pOs[nControl].Getz() );

//      wrt tx:
        ppJ[ pnV[nControl] ][6] = - pdAlpha[nControl] * pdGamma[nControl] / cd.Gettx();
    }
    delete [] pdXc;
    delete [] pdZc;
    delete [] pdAlpha;
    delete [] pdBeta;
    delete [] pdGamma;
    delete [] pdDelta;
    delete [] pdEpsilon;
    delete [] pdRd;
    delete [] pnV;

    return TRUE;
}
```

```
// Adds Y-axis Jacobian elements from the partial derivatives
void CObjectSpace::ConstructJy( CCalibrationData& cd,
                                CObjectSpace* pOs,
                                double** ppJ,
                                double** ppR,
                                UINT pts )
{
    double *pdXc, *pdYc, *pdZc, *pdRd, *pdAlpha, *pdBeta, *pdGamma;
    UINT   *pnVb;

    pdXc    =    new double[pts+1];
    pdYc    =    new double[pts+1];
    pdZc    =    new double[pts+1];
    pdAlpha =    new double[pts+1];
    pdBeta  =    new double[pts+1];
    pdGamma =    new double[pts+1];
    pdRd    =    new double[pts+1];
    pnVb    =    new UINT[pts+1];

    for ( UINT nControl = 1; nControl <= pts; nControl++ )
    {
        pnVb[nControl] = pOs[nControl].m_Pt + pOs[nControl-1].m_Pt + 1 ;

        pdXc[nControl] =  ppR[1][1] * pOs[nControl].Getx() +
                          ppR[1][2] * pOs[nControl].Gety() +
                          ppR[1][3] * pOs[nControl].Getz();

        pdYc[nControl] =  ppR[2][1] * pOs[nControl].Getx() +
                          ppR[2][2] * pOs[nControl].Gety() +
                          ppR[2][3] * pOs[nControl].Getz() - cd.Getty();

        pdZc[nControl] =  ppR[3][1] * pOs[nControl].Getx() +
                          ppR[3][2] * pOs[nControl].Gety() +
                          ppR[3][3] * pOs[nControl].Getz();

        pdRd[nControl] = sqrt( SQR( pdXc[nControl] ) + SQR( pdZc[nControl] ) );

        pdAlpha[nControl] = sqrt( SQR( pdRd[nControl] ) - SQR( cd.Gettx() ) );

        pdBeta[nControl] = ( cd.Getf() / APT ) * ( 1.0 / ( cd.Gettz() - pdAlpha[nControl] ) );

        pdGamma[nControl] = pdYc[nControl] / ( pdAlpha[nControl] * ( cd.Gettz() - pdAlpha[nControl] )
);

//      wrt omega:
        ppJ[ pnVb[nControl] ][2] = pdBeta[nControl] * (
                ( - ppR[2][3] * pOs[nControl].Gety() + ppR[2][2] * pOs[nControl].Getz() ) +
                pdGamma[nControl] * (
                pdXc[nControl] * ( - ppR[1][3] * pOs[nControl].Gety() + ppR[1][2] *
pOs[nControl].Getz() ) +
                pdZc[nControl] * ( - ppR[3][3] * pOs[nControl].Gety() + ppR[3][2] *
pOs[nControl].Getz() ) ) );

//      wrt phi:
        ppJ[ pnVb[nControl] ][3] = pdBeta[nControl] * pdZc[nControl] * (
                sin( cd.Getkappa() ) + pdGamma[nControl] * (
                - pdXc[nControl] * cos( cd.Getkappa() ) + (
```

```
                    cos( cd.Getphi() ) * pOs[nControl].Getx() +
                    sin( cd.Getomega() ) * sin( cd.Getphi() ) * pOs[nControl].Gety() -
                    cos( cd.Getomega() ) * sin( cd.Getphi() ) * pOs[nControl].Getz() ) );

//          wrt kappa:
            ppJ[ pnVb[nControl] ][4] = pdBeta[nControl] * pdXc[nControl] * (
                    - 1.0 + pdGamma[nControl] * ( pdYc[nControl] + cd.Getty() ) );

//          wrt f:
            ppJ[ pnVb[nControl] ][5] = pdBeta[nControl] * pdYc[nControl] / cd.Getf();

//          wrt tx:
            ppJ[ pnVb[nControl] ][6] = - pdBeta[nControl] * pdGamma[nControl] * cd.Gettx();

//          wrt ty:
            ppJ[ pnVb[nControl] ][7] = - pdBeta[nControl];

//          wrt tz:
            ppJ[ pnVb[nControl] ][8] = - pdBeta[nControl] * pdYc[nControl] / ( cd.Gettz() -
pdAlpha[nControl] );
        }
        delete [] pdXc;
        delete [] pdYc;
        delete [] pdZc;
        delete [] pdAlpha;
        delete [] pdBeta;
        delete [] pdGamma;
        delete [] pdRd;
        delete [] pnVb;
}


// Verify orthonormality of rotation matrices
BOOL CObjectSpace::VerifyOrthonormality( double** ppR )
{
        double    dInter[4] = { 0, 0, 0, 0 };

        for( UINT i = 1; i < 4; i++ )
                for( UINT j = 1; j < 4; j++ )
                        dInter[i] += SQR( ppR[i][j] );

        double    dVer = 0.0;
        for( UINT n = 1; n < 4; n++ )       dVer += ( sqrt( dInter[n] ) );

        // roundoff / truncation errors
        if ( ( dVer < 3.0 + 1e-15 ) && ( dVer > 3.0 - 1e-15 ) )
                return     TRUE;
        else
                return FALSE;
}
```

File: Framebuffer.cpp

```
#include "FrameBuffer.h"
#include "ObjectSpace.h"
```

```
// Calculates frame buffer coordinates:
void CFrameBuffer::CalcFrmBufferCoords( CCalibrationData& cd,
                                        double* pdvCalc,
                                        double** ppR,
                                        CObjectSpace* pOs,
                                        UINT nPoints,
                                        UINT* pnvZone )
{
    double   *pdXc, *pdYc, *pdZc, *pdRc;
    UINT     *pnVa,  *pnVb;

    pdXc = new double[nPoints+1];
    pdYc = new double[nPoints+1];
    pdZc = new double[nPoints+1];
    pdRc = new double[nPoints+1];
    pnVa = new UINT[nPoints+1];
    pnVb = new UINT[nPoints+1];

    double sx = 30.0 / ( PI * cd.Geturtip() );

    for ( UINT nControl = 1; nControl <= nPoints; nControl++ )
    {
        pnVa[nControl] = (UINT)pOs[nControl].Getp() + (UINT)pOs[nControl-1].Getp();
        pnVb[nControl] = pnVa[nControl] + 1;

        pdXc[nControl] =  ppR[1][1] * pOs[nControl].Getx() +
                          ppR[1][2] * pOs[nControl].Gety() +
                          ppR[1][3] * pOs[nControl].Getz();

        pdYc[nControl] =  ppR[2][1] * pOs[nControl].Getx() +
                          ppR[2][2] * pOs[nControl].Gety() +
                          ppR[2][3] * pOs[nControl].Getz() - cd.Getty();

        pdZc[nControl] =  ppR[3][1] * pOs[nControl].Getx() +
                          ppR[3][2] * pOs[nControl].Gety() +
                          ppR[3][3] * pOs[nControl].Getz();

        pdRc[nControl] = sqrt( SQR( pdXc[nControl] ) + SQR( pdZc[nControl] ) );

        if ( pnvZone[nControl] == 1 )
        {
            pdvCalc[ pnVa[nControl] ] = sx * ( PI / 2.0 - asin( cd.Gettx() / pdRc[nControl] ) -
                    asin( pdZc[nControl] / pdRc[nControl] ) );
        }
        else
        {
            pdvCalc[ pnVa[nControl] ] = sx * ( 3 * PI / 2.0 - asin( cd.Gettx() / pdRc[nControl] ) +
                    asin( pdZc[nControl] / pdRc[nControl] ) );
        }

        pdvCalc[ pnVb[nControl] ] = IMGC + ( cd.Getf() / APT ) * (
                pdYc[nControl] / ( cd.Gettz() - sqrt( SQR( pdRc[nControl] ) - SQR( cd.Gettx() ) ) ) );

    }
    delete [] pdXc;
    delete [] pdYc;
    delete [] pdZc;
    delete [] pdRc;
```

```
    delete [] pnVa;
    delete [] pnVb;
}

// Construct Vector F:
void CFrameBuffer::ConstructF( double* pdvF,
                               double* pdvCalc,
                               CFrameBuffer* pObserved,
                               UINT nPoints )
{
    for ( UINT nCount = 1; nCount <= nPoints; nCount++ )
    {
        int xvar = 2 * nCount - 1;
        int yvar = xvar + 1;

        pdvF[xvar] = pObserved[nCount].Getxl() - pdvCalc[xvar];
        pdvF[yvar] = pObserved[nCount].Getyl() - pdvCalc[yvar];
    }
}
```

File: Intersection.cpp

```
#include "Intersection.h"

void CIntersection::SetFrm( double xfL, double yfL, double xfR, double yfR )
{
    m_FrmBuffer.xfL = xfL;
    m_FrmBuffer.yfL = yfL;
    m_FrmBuffer.xfR = xfR;
    m_FrmBuffer.yfR = yfR;
}

void CIntersection::SetObj( double Xos, double Yos, double Zos )
{
    m_ObjSpace.Xos = Xos;
    m_ObjSpace.Yos = Yos;
    m_ObjSpace.Zos = Zos;
}

void CIntersection::SetCnd( UINT znL, UINT znR )
{
    m_Cnd.znL = znL;
    m_Cnd.znR = znR;
}

int CIntersection::ImproveIt( CCalibrationData& rCL, CCalibrationData& rCR,
                              double** ppRL, double** ppRR,
                              OBJSPACE* os,
                              CND* zone )
{
    int iter;
    double aM[iEQUS+1][iUNKS+1], bM[iEQUS+1];

    COperations* pOP = new COperations;
```

```
double** ia;
double** iu;
double** iv;

ia = pOP->DMatrix( 1, iEQUS, 1, iUNKS );
iu = pOP->DMatrix( 1, iEQUS, 1, iEQUS );
iv = pOP->DMatrix( 1, iUNKS, 1, iUNKS );

double* iw = new double[iUNKS+1];
double* ib = new double[iEQUS+1];
double* ix = new double[iUNKS+1];

double   XcL, XcR, YcL, YcR, ZcL, ZcR, rcL, rcR, FxL, FyL, FzL, FxR, FyR, FzR,
         alphaXL, alphaXR, alphaYL, alphaYR, betaXL, betaXR, betaYL, betaYR,
         gammaL, gammaR, deltaL, deltaR, sxL, sxR;

for( iter = 0; iter <= iITER; iter++ )
{
     XcL = ppRL[1][1] * os->Xos + ppRL[1][2] * os->Yos + ppRL[1][3] * os->Zos;
     YcL = ppRL[2][1] * os->Xos + ppRL[2][2] * os->Yos + ppRL[2][3] * os->Zos - rCL.Getty();
     ZcL = ppRL[3][1] * os->Xos + ppRL[3][2] * os->Yos + ppRL[3][3] * os->Zos;

     XcR = ppRR[1][1] * os->Xos + ppRR[1][2] * os->Yos + ppRR[1][3] * os->Zos;
     YcR = ppRR[2][1] * os->Xos + ppRR[2][2] * os->Yos + ppRR[2][3] * os->Zos - rCR.Getty();
     ZcR = ppRR[3][1] * os->Xos + ppRR[3][2] * os->Yos + ppRR[3][3] * os->Zos;

     FxL = ( ppRL[1][1] * ppRL[1][1] + ppRL[3][1] * ppRL[3][1] ) * os->Xos +
           ( ppRL[1][1] * ppRL[1][2] + ppRL[3][1] * ppRL[3][2] ) * os->Yos +
           ( ppRL[1][1] * ppRL[1][3] + ppRL[3][1] * ppRL[3][3] ) * os->Zos;

     FyL = ( ppRL[1][1] * ppRL[1][2] + ppRL[3][1] * ppRL[3][2] ) * os->Xos +
           ( ppRL[1][2] * ppRL[1][2] + ppRL[3][2] * ppRL[3][2] ) * os->Yos +
           ( ppRL[1][2] * ppRL[1][3] + ppRL[3][2] * ppRL[3][3] ) * os->Zos;

     FzL = ( ppRL[1][1] * ppRL[1][3] + ppRL[3][1] * ppRL[3][3] ) * os->Xos +
           ( ppRL[1][2] * ppRL[1][3] + ppRL[3][2] * ppRL[3][3] ) * os->Yos +
           ( ppRL[1][3] * ppRL[1][3] + ppRL[3][3] * ppRL[3][3] ) * os->Zos;

     FxR = ( ppRR[1][1] * ppRR[1][1] + ppRR[3][1] * ppRR[3][1] ) * os->Xos +
           ( ppRR[1][1] * ppRR[1][2] + ppRR[3][1] * ppRR[3][2] ) * os->Yos +
           ( ppRR[1][1] * ppRR[1][3] + ppRR[3][1] * ppRR[3][3] ) * os->Zos;

     FyR = ( ppRR[1][1] * ppRR[1][2] + ppRR[3][1] * ppRR[3][2] ) * os->Xos +
           ( ppRR[1][2] * ppRR[1][2] + ppRR[3][2] * ppRR[3][2] ) * os->Yos +
           ( ppRR[1][2] * ppRR[1][3] + ppRR[3][2] * ppRR[3][3] ) * os->Zos;

     FzR = ( ppRR[1][1] * ppRR[1][3] + ppRR[3][1] * ppRR[3][3] ) * os->Xos +
           ( ppRR[1][2] * ppRR[1][3] + ppRR[3][2] * ppRR[3][3] ) * os->Yos +
           ( ppRR[1][3] * ppRR[1][3] + ppRR[3][3] * ppRR[3][3] ) * os->Zos;

     rcL = sqrt( SQR( XcL ) + SQR( ZcL ) );
     rcR = sqrt( SQR( XcR ) + SQR( ZcR ) );

     alphaXL = 30.0 / ( PI * rCL.Geturtip() * rcL );
     alphaXR = 30.0 / ( PI * rCR.Geturtip() * rcR );

     betaXL = -1.0 / sqrt( 1.0 - ( SQR( ZcL ) / SQR( rcL ) ) );
```

```
betaXR = -1.0 / sqrt( 1.0 - ( SQR( ZcR ) / SQR( rcR ) ) );

if( zone->znL == 2 )    betaXL = - betaXL;
if( zone->znR == 2 )    betaXR = - betaXR;

gammaL = rCL.Gettx() / sqrt( 1.0 - ( SQR( rCL.Gettx() ) / SQR( rcL ) ) );
gammaR = rCR.Gettx() / sqrt( 1.0 - ( SQR( rCR.Gettx() ) / SQR( rcR ) ) );

deltaL = ( betaXL * ZcL - gammaL ) / SQR( rcL );
deltaR = ( betaXR * ZcR - gammaR ) / SQR( rcR );

alphaYL = ( rCL.Getf() / APT ) / ( rCL.Gettz() - sqrt( SQR( rcL ) - SQR( rCL.Gettx() ) ) );

betaYL = YcL / ( ( rCL.Gettz() - sqrt(
         SQR( rcL ) - SQR( rCL.Gettx() ) ) ) * ( sqrt( SQR( rcL ) - SQR( rCL.Gettx() ) ) ) );

alphaYR = ( rCR.Getf() / APT ) / ( rCR.Gettz() - sqrt( SQR( rcR ) - SQR( rCR.Gettx() ) ) );

betaYR = YcR / ( ( rCR.Gettz() - sqrt(
         SQR( rcR ) - SQR( rCR.Gettx() ) ) ) * ( sqrt( SQR( rcR ) - SQR( rCR.Gettx() ) ) ) );

aM[1][1] = alphaXL * ( ppRL[3][1] * betaXL - deltaL * FxL );
aM[1][2] = alphaXL * ( ppRL[3][2] * betaXL - deltaL * FyL );
aM[1][3] = alphaXL * ( ppRL[3][3] * betaXL - deltaL * FzL );

aM[2][1] = alphaYL * ( ppRL[2][1] + betaYL * ( ppRL[1][1] * XcL + ppRL[3][1] * ZcL ) );
aM[2][2] = alphaYL * ( ppRL[2][2] + betaYL * ( ppRL[1][2] * XcL + ppRL[3][2] * ZcL ) );
aM[2][3] = alphaYL * ( ppRL[2][3] + betaYL * ( ppRL[1][3] * XcL + ppRL[3][3] * ZcL ) );

aM[3][1] = alphaXR * ( ppRR[3][1] * betaXR - deltaR * FxR );
aM[3][2] = alphaXR * ( ppRR[3][2] * betaXR - deltaR * FyR );
aM[3][3] = alphaXR * ( ppRR[3][3] * betaXR - deltaR * FzR );

aM[4][1] = alphaYR * ( ppRR[2][1] + betaYR *( ppRR[1][1] * XcR + ppRR[3][1] * ZcR ) );
aM[4][2] = alphaYR * ( ppRR[2][2] + betaYR *( ppRR[1][2] * XcR + ppRR[3][2] * ZcR ) );
aM[4][3] = alphaYR * ( ppRR[2][3] + betaYR *( ppRR[1][3] * XcR + ppRR[3][3] * ZcR ) );

// construct vB:

sxL = 30.0 / ( PI * rCL.Geturtip() );
sxR = 30.0 / ( PI * rCR.Geturtip() );

bM[1] = GetFrm().xfL - ( sxL * ( PI / 2.0 - asin( rCL.Gettx() / rcL ) - asin( ZcL / rcL ) ) );

bM[3] = GetFrm().xfR - ( sxR * ( PI / 2.0 - asin( rCR.Gettx() / rcR ) - asin( ZcR / rcR ) ) );

if( zone->znL == 2 )
    bM[1] = GetFrm().xfL - ( sxL * ( 3 * PI / 2.0 - asin( rCL.Gettx() / rcL ) + asin( ZcL / rcL ) ) );

if( zone->znR == 2 )
    bM[3] = GetFrm().xfR - ( sxR * ( 3 * PI / 2.0 - asin( rCR.Gettx() / rcR ) + asin( ZcR / rcR ) ) );

bM[2] = GetFrm().yfL - ( IMGC + ( rCL.Getf() / APT ) * (
         YcL / ( rCL.Gettz() - sqrt( SQR( rcL ) - SQR( rCL.Gettx() ) ) ) ) );

bM[4] = GetFrm().yfR - ( IMGC + ( rCR.Getf() / APT ) * (
         YcR / ( rCR.Gettz() - sqrt( SQR( rcR ) - SQR( rCR.Gettx() ) ) ) ) );
```

```
        // vA and vB calculated - continue:

        for( UINT i = 1; i <= iEQUS; i++ )
            for( UINT j = 1; j <= iUNKS; j++ )
                iu[i][j] = aM[i][j];

        for( i = 1; i <= iEQUS; i++ )
            ib[i] = bM[i];

        pOP->SvdDcmp( iu, iEQUS, iUNKS, iw, iv );
        pOP->SvdBksb( iu, iw, iv, iEQUS, iUNKS, ib, ix );

        os->Xos += ix[1];
        os->Yos += ix[2];
        os->Zos += ix[3];

        if(  fabs( ix[1] / os->Xos ) < iCONV_LIMIT &&
            fabs( ix[2] / os->Yos ) < iCONV_LIMIT &&
            fabs( ix[3] / os->Zos ) < iCONV_LIMIT )
        {
            delete [] ix;
            delete [] ib;
            delete [] iw;
            pOP->FreeDMatrix( iv, 1, 1 );
            pOP->FreeDMatrix( iu, 1, 1 );
            pOP->FreeDMatrix( ia, 1, 1 );
            delete pOP;

            return iter;     // Converged - stop searching
        }

        if( ( fabs( ix[1] ) + fabs( ix[2] ) + fabs( ix[3] ) ) > iDIV_LIMIT )
        {
            delete [] ix;
            delete [] ib;
            delete [] iw;
            pOP->FreeDMatrix(iv, 1, 1);
            pOP->FreeDMatrix(iu, 1, 1);
            pOP->FreeDMatrix(ia, 1, 1);
            delete pOP;
            AfxMessageBox( " Divergence detected...aborting process.", MB_ICONSTOP );
            return DIVERGED;
        }
    }

    delete [] ix;    // significant residuals
    delete [] ib;
    delete [] iw;
    pOP->FreeDMatrix(iv, 1, 1);
    pOP->FreeDMatrix(iu, 1, 1);
    pOP->FreeDMatrix(ia, 1, 1);
    delete pOP;

    return iter;
}
```

```
UCR CIntersection::Uncertainty( CCalibrationData& rCL, CCalibrationData& rCR,
                                double** ppRL, double** ppRR,
                                OBJSPACE* absCoords,
                                CND* zone )
{
    // Re-project solution:
    double   xcBackL, ycBackL, zcBackL, xcBackR, ycBackR, zcBackR,
             rcBackL, rcBackR, xfBackL, yfBackL, xfBackR, yfBackR, sxL, sxR;

    xcBackL =   ppRL[1][1] * absCoords->Xos +
                ppRL[1][2] * absCoords->Yos +
                ppRL[1][3] * absCoords->Zos;
    ycBackL =   ppRL[2][1] * absCoords->Xos +
                ppRL[2][2] * absCoords->Yos +
                ppRL[2][3] * absCoords->Zos - rCL.Getty();
    zcBackL =   ppRL[3][1] * absCoords->Xos +
                ppRL[3][2] * absCoords->Yos +
                ppRL[3][3] * absCoords->Zos;

    xcBackR =   ppRR[1][1] * absCoords->Xos +
                ppRR[1][2] * absCoords->Yos +
                ppRR[1][3] * absCoords->Zos;
    ycBackR =   ppRR[2][1] * absCoords->Xos +
                ppRR[2][2] * absCoords->Yos +
                ppRR[2][3] * absCoords->Zos - rCR.Getty();
    zcBackR =   ppRR[3][1] * absCoords->Xos +
                ppRR[3][2] * absCoords->Yos +
                ppRR[3][3] * absCoords->Zos;

    rcBackL = sqrt( SQR( xcBackL ) + SQR( zcBackL ) );
    rcBackR = sqrt( SQR( xcBackR ) + SQR( zcBackR ) );

    sxL = 30.0 / ( PI * rCL.Geturtip() );
    sxR = 30.0 / ( PI * rCR.Geturtip() );

    if( zone->znL == 2 )
        xfBackL = sxL * ( 3.0 * PI / 2.0 + asin( zcBackL / rcBackL ) - asin( rCL.Gettx() / rcBackL ) );
    else
        xfBackL = sxL * ( PI / 2.0 - asin( zcBackL / rcBackL ) - asin( rCL.Gettx() / rcBackL ) );


    if( zone->znR == 2 )
        xfBackR = sxR * ( 3.0 * PI / 2.0 + asin( zcBackR / rcBackR ) - asin( rCR.Gettx() / rcBackR ) );
    else
        xfBackR = sxR * ( PI / 2.0 - asin( zcBackR / rcBackR ) - asin( rCR.Gettx() / rcBackR ) );

    yfBackL = IMGC + ( rCL.Getf() / APT ) * ( ycBackL / ( rCL.Gettz() - sqrt(
              SQR( rcBackL ) - SQR( rCL.Gettx() ) ) ) );

    yfBackR = IMGC + ( rCR.Getf() / APT ) * ( ycBackR / ( rCR.Gettz() - sqrt(
              SQR( rcBackR ) - SQR( rCR.Gettx() ) ) ) );

//-------------------------------------------------------------------------
// The following calculates the spatial quantisation error:

    double   xcL, ycL, zcL, xcR, ycR, zcR, rcL, rcR, xCalcL, xCalcR, yCalcL, yCalcR,
             xResPos, yResPos, zResPos, xResNeg, yResNeg, zResNeg,
             storedXL, storedXR, storedYL, storedYR;
```

```
        OBJSPACE solution = *absCoords;     // preserve object space solution

        double Xw = absCoords->Xos;
        double Yw = absCoords->Yos;
        double Zw = absCoords->Zos;

        double Xwnew = Xw;
        double Ywnew = Yw;
        double Zwnew = Zw;

//  ------------------------------------------------------------------------
//      Calculate radial uncertainty:
        xcL = ppRL[1][1] * Xwnew + ppRL[1][2] * Ywnew + ppRL[1][3] * Zwnew;
        zcL = ppRL[3][1] * Xwnew + ppRL[3][2] * Ywnew + ppRL[3][3] * Zwnew;
        rcL = sqrt( SQR( xcL ) + SQR( zcL ) );

        xcR = ppRR[1][1] * Xwnew + ppRR[1][2] * Ywnew + ppRR[1][3] * Zwnew;
        zcR = ppRR[3][1] * Xwnew + ppRR[3][2] * Ywnew + ppRR[3][3] * Zwnew;
        rcR = sqrt( SQR( xcR ) + SQR( zcR ) );

        double angleL = asin( zcL / rcL );
        double angleR = asin( zcR / rcR );

        double rStepL = rcL;            // Fix radius
        double rStepR = rcR;

        double xfL, xfR;
        if( zone->znL == 2 )
            xfL = sxL * ( 3 * PI / 2.0 + angleL - asin( rCL.Gettx() / rStepL ) );
        else
            xfL = sxL * ( PI / 2.0 - angleL - asin( rCL.Gettx() / rStepL ) );

        if( zone->znR == 2 )
            xfR = sxR * ( 3 * PI / 2.0 + angleR - asin( rCR.Gettx() / rStepR ) );
        else
            xfR = sxR * ( PI / 2.0 - angleR - asin( rCR.Gettx() / rStepR ) );

        double disp = xfL - xfR;
        double tDisp, RadialUncertainty;

        for( long iter = 0; iter <= ITER1; iter++ )
        {
            if( zone->znL == 2 )
                xfL = sxL * ( 3.0 * PI / 2.0 + angleL - asin( rCL.Gettx() / rStepL ) );
            else
                xfL = sxL * ( PI / 2.0 - angleL - asin( rCL.Gettx() / rStepL ) );

            if( zone->znR == 2 )
                xfR = sxR * ( 3.0 * PI / 2.0 + angleR - asin( rCR.Gettx() / rStepR ) );
            else
                xfR = sxR * ( PI / 2.0 - angleR - asin( rCR.Gettx() / rStepR ) );

            tDisp = xfL - xfR;

            if( fabs( tDisp - disp) >= 1.0 ) {
                RadialUncertainty = rStepL - rcL;
                break;
```

```
        }
        rStepL += 1e-6;  // 1 micron
        rStepR += 1e-6;  // 1 micron
    }

// ------------------------------------------------------------------------
//    Calculate angular uncertainty:
    angleL = asin( zcL / rcL );
    angleR = asin( zcR / rcR );

    if( zone->znL == 2 )
        angleL = 3 * PI / 2.0 + angleL - asin( rCL.Gettx() / rcL );
    else
        angleL = PI / 2.0 - angleL - asin( rCL.Gettx() / rcL );

    if( zone->znR == 2 )
        angleR = 3 * PI / 2.0 + angleR - asin( rCR.Gettx() / rcR );
    else
        angleR = PI / 2.0 - angleR - asin( rCR.Gettx() / rcR );

    double angStL = angleL;      // Fix angle
    double angStR = angleR;

    xfL = sxL * angleL;
    xfR = sxR * angleR;

    double AngularUncertainty, xfLnew, xfRnew;

    for( iter = 0; iter <= ITER1; iter++ )
    {
        if( zone->znL == 2 )
            xfLnew = sxL * angStL;
        else
            xfLnew = sxL * angStL;

        if( zone->znR == 2 )
            xfRnew = sxR * angStR;
        else
            xfRnew = sxR *.angStR;

        if( ( fabs( xfLnew - xfL ) >= 1.0 ) && ( fabs( xfRnew - xfR ) >= 1.0 ) )     {
                AngularUncertainty = angStL - angleL;
                break;
        }
        angStL += 1e-6;    // 1 microrad
        angStR += 1e-6;
    }

//    NOTE:The routines for calculating the Cartesian uncertainties are similar to that of the radial
//    uncertainty given above. In order to conserve space, they will not be provided in the thesis.

    UNC unc;
    unc.dXpos = 1e3 * xResPos;
    unc.dYpos = 1e3 * yResPos;
    unc.dZpos = 1e3 * zResPos;
    unc.dXneg = 1e3 * xResNeg;
    unc.dYneg = 1e3 * yResNeg;
    unc.dZneg = 1e3 * zResNeg;
```

```
    unc.dR = 1e3 * RadialUncertainty;
    unc.dA = RadToDeg( AngularUncertainty );
    unc.reXfL = xfLBack;
    unc.reYfL = yfLBack;
    unc.reXfR = xfRBack;
    unc.reYfR = yfRBack;

    return unc;
}
```

Appendix V

Control Field Calibration Data

TABLE A *Control field calibration data.*

Target	Radius (mm)	Angle (^{0})	Y (mm)
A2	93.437	0.239	130.106
A3	94.636	67.569	130.784
A4	93.286	90.291	129.926
A5	94.002	112.615	130.003
A6	93.854	165.372	130.320
A7	92.717	195.436	130.685
A1	93.574	270.681	130.318
B2	83.218	0.249	110.080
B3	84.973	67.551	110.720
B4	84.369	90.225	109.965
B5	84.214	112.627	110.023
B6	83.926	165.442	110.321
B7	83.395	195.459	110.541
B1	84.317	270.667	110.29
C2	74.026	0.278	90.176
C3	74.821	67.505	90.626
C4	73.769	90.212	89.894
C5	74.099	112.659	90.020
C6	73.531	165.431	90.111
C7	73.876	195.542	90.569
C1	74.155	270.617	90.238
D2	64.192	0.252	70.021
D3	64.068	67.496	70.601
D4	64.913	90.152	69.822
D5	63.671	112.671	70.115
D6	64.621	165.439	70.093
D7	63.886	195.608	70.530
D1	63.170	270.591	70.133
E2	54.084	0.158	49.996
E3	54.057	67.392	50.686
E4	54.058	90.116	49.786
E5	53.728	112.699	49.891
E6	54.324	165.555	50.199
E7	53.350	195.756	50.414
E1	53.626	270.489	50.092
F2	44.164	0.207	29.935
F3	44.533	67.371	30.678
F4	44.243	90.025	29.791
F5	44.032	112.573	30.055
F6	44.007	165.644	30.127
F7	43.735	195.901	30.439
F1	43.662	270.357	29.956
G2	34.096	0.162	9.915
G3	33.954	67.319	10.653
G4	34.998	90.026	9.929
G5	34.501	112.662	9.937
G6	34.208	165.725	10.053
G7	33.572	196.215	10.424
G1	35.130	270.225	10.016