

Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be “used for any purpose other than private study, scholarship, or research.” If a user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of “fair use” that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select “Pages from: first page # to: last page #” on the print dialog screen



The Van Houten library has removed some of the personal information and all signatures from the approval page and biographical sketches of theses and dissertations in order to protect the identity of NJIT graduates and faculty.

ABSTRACT

A STEREOSCOPIC SYSTEM USING KNOWLEDGE PROPAGATION TO ACHIEVE ACCURATE DEPTH CALCULATION

**by
Chaitali Mulay**

With the advent of the twenty first century, stereoscopic systems have found a widespread use in the engineering industry. Several biomechanical analyses utilize this concept for efficient information extraction. Some examples of its applications are gait analysis, hand shape recognition, facial surface recognition etc.

The primary goal of this thesis was to optimize the existing stereoscopic system, to increase accuracy and precision of the depth information extracted. The process included redesign of the existing equipment set-up, automation of image acquisition unit and modification of conventionally used correspondence test to achieve higher accuracy. The acquired data was used to estimate depth of the object used for the study.

It was found that the correspondence information for a pair of adjacent cameras had high accuracy. In addition, the plots of the correspondences exhibited similarity in the trend. An attempt was made to use this information for predicting the values for regions on the curves having inconsistencies. Depth estimation using triangulation was performed on the correspondences found for adjacent pairs. It was found that the row was reconstructed as anticipated using the algorithm. With further development of the algorithm and successful implementation of knowledge propagation, this system can demonstrate efficient shape recovery.

**A STEREOSCOPIC SYSTEM USING KNOWLEDGE PROPAGATION TO
ACHIEVE ACCURATE DEPTH CALCULATION**

**by
Chaitali Mulay**

**A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Biomedical Engineering**

Department of Biomedical Engineering

January 2008

Blank Page

APPROVAL PAGE

**A STEREOSCOPIC SYSTEM USING KNOWLEDGE PROPAGATION TO
ACHIEVE ACCURATE DEPTH CALCULATION**

Chaitali Mulay

Dr. Richard Foulds, Thesis Advisor

Date

Associate Professor of Biomedical Engineering, NJIT

Dr. Sergei Adamovich

Date

Assistant Professor of Biomedical Engineering, NJIT

Dr. Bruno Mantilla

Date

Special Lecturer of Biomedical Engineering, NJIT

BIOGRAPHICAL SKETCH

Author: Chaitali Mulay
Degree: Master of Science
Date: January 2008

Undergraduate and Graduate Education:

- Master of Science in Biomedical Engineering,
New Jersey Institute of Technology, Newark, NJ, 2008
- Bachelor of Engineering in Biomedical Engineering,
Mumbai University, Mumbai, India, 2006

Major: Biomedical Engineering

To my beloved parents,
without whom this journey would have been
impossible.

ACKNOWLEDGMENT

I would like to take this opportunity to express my deep gratitude to Dr. Richard Foulds, who not only acted as my thesis advisor, providing valuable information, technical guidance and insight into the project but also for his patience, encouragement and moral support that he gave me throughout the project. A special thanks to Dr. Sergei Adamovich and Dr. Bruno Mantilla who not only acted as my committee members but also provided encouragement and moral support throughout the period of this project.

I would like to thank all my professors at NJIT for their mentorship, which strengthened my theoretical concepts and practical knowledge. I would also like to extend my thanks to Amanda Irving, Diego Ramirez and Katherine Swift for helping me troubleshoot all the technical problems I faced while the project was in process. Very special thanks to my dearest parents, who have stood by me through every phase of life and provided constant guidance and encouragement. I also wish to thank my friends, Geeta Pamidi, Charu Chande, Kunal Doshi, Namrata Patel and Ashwini Nikam, who have been supportive throughout my master's study at NJIT.

TABLE OF CONTENTS

Chapter		Page
1	INTRODUCTION.....	1
	1.1 Objective.....	1
	1.2 The Human Visual Cortex.....	1
	1.3 Stereovision.....	2
2	PRINCIPLES OF STEREOVISION.....	4
	2.1 Camera model.....	4
	2.1.1 Perspective camera model.....	4
	2.1.2 Coplanar stereo geometry.....	5
	2.2 Calibration.....	7
	2.2.1 Intrinsic Parameters.....	9
	2.2.2 Extrinsic Parameters.....	9
	2.3 Rectification.....	10
	2.4 Correspondence matching	12
	2.4.1 Intensity based stereo analysis.....	12
	2.4.2 Feature based stereo analysis.....	13
	2.4.3 1D correspondence search.....	13
	2.5 Triangulation.....	14
3	EXPERIMENTAL SET-UP AND PROCEDURE.....	16
	3.1 Experimental set-up.....	16
	3.1.1 Optimum base distance calculation.....	16
	3.1.2 Equipment set-up.....	17
	3.2 Hardware profile.....	19
	3.2.1 Cameras.....	20
	3.2.2 Computer System.....	21
	3.3 Software profile.....	21
	3.3.1 MATLAB (version R2007a).....	21
	3.3.2 Camera calibration toolbox for MATLAB.....	23

TABLE OF CONTENTS
(Continued)

Chapter	Page
3.4 Procedure.....	24
3.4.1 Image acquisition.....	24
3.4.2 Camera Calibration.....	25
3.4.3 Image rectification.....	31
3.4.4 Correspondence analysis.....	32
3.4.5 Depth Estimation.....	36
4 RESULTS.....	37
4.1 Correspondence Matching.....	37
4.2 Row-wise Correspondence Test.....	39
4.3 Depth Information Extraction.....	43
5 CONCLUSION AND FUTURE WORK.....	48
APPENDIX A CAMERA CALIBRATION RESULTS.....	49
APPENDIX B MATLAB CODE USED FOR THESIS.....	53
REFERENCES.....	60

LIST OF TABLES

Table		Page
4.1	The results of robustness test performed on the modified intensity-based correspondence matching algorithm.....	38
4.2	Coordinates of the first and last pixel for the rows tested in row-wise correspondence analysis.....	40
4.3	Comparative chart of the expected and detected end points.....	43

LIST OF FIGURES

Figure	Page
2.1 The pinhole camera model.....	5
2.2 Two pinhole camera models are used to illustrate coplanar stereo geometry.....	6
2.3 Two examples of calibration object (A) Checkerboard (B) Open Cube.....	8
2.4 An illustration of epipolar geometry.....	11
3.1 The experimental set-up being used for the project.....	18
3.2 Daisy chained cameras connected to a computer.....	19
3.3 A Pixelink A642 camera.....	20
3.4 A typical MATLAB environment.....	22
3.5 The graphical user interface of the camera calibration toolbox.....	23
3.6 The graphical user interface for the stereo camera calibration toolbox.....	24
3.7 Planar checkerboard placed for calibration process.....	26
3.8 Images captured by camera 11262 for calibration.....	27
3.9 (A) Manual grid corners selection (B) Automated corner extraction (C) Automated square counting by the camera calibration toolbox.....	28
3.10 The intrinsic parameters for camera 11262 displayed at the command window in MATLAB.....	29
3.11 The output of calibration results for camera 11248 and 11247.....	30
3.12 The graphical depiction of the extrinsics of a stereo rig computed for camera 11248 and 11247.....	30
3.13 A subplot of rectified images for cameras 11248 and 11247 has been shown with row 535 highlighted for illustration.....	31
3.14 The series of prompts that appear on the command window after <i>corr_code</i> is executed.....	33

LIST OF FIGURES
(Continued)

Figure	Page
3.15 (A) and (C) Block surrounding test pixel at (376,601). (B) Block surrounding corresponding pixel at (376,640) in second image. (D) Histogram equalized block surrounding corresponding pixel at (376,640).....	33
3.16 The result of the correspondence match test for pixel at (376, 601) is displayed in the command window.....	34
3.17 The plot of correlation values along row 376 shows a large spike at $y = 640$	34
3.18 Visual inspection of the point yields the matching point at (376,639).....	35
3.19 Trend of the conjugates observed in row 490 for camera pair 11262 and 11218.....	35
3.20 3D reconstruction of the midline of the object using data from cameras 11262 and 11218.....	36
4.1 Five points chosen for testing the modified intensity-based correspondence matching algorithm.....	37
4.2 Graph showing the trend for variation of y coordinates of the conjugate pair for cameras 11248 and 11262.....	40
4.3 Graph showing the trend for variation of y coordinates of the conjugate pair for cameras 11262 and 11218.....	40
4.4 Graph showing the trend for variation of y coordinates of the conjugate pair for cameras 11218 and 08719.....	41
4.5 Graph showing the trend for variation of y coordinates of the conjugate pair for cameras 08719 and 11247.....	41
4.6 Graph showing the trend for variation of y coordinates of the conjugate pair for cameras 11248 and 11247.....	41
4.7 The section of the image that was reconstructed using triangulation.....	44
4.8 Depth estimation using correspondence data from cameras 11248 and 11262.....	45
4.9 Depth estimation using correspondence data from cameras 11262 and 11218.....	45

LIST OF FIGURES
(Continued)

Figure	Page
4.10 3D reconstruction of the nose of the mask using depth estimation algorithm developed.....	46
4.11 Different views of the reconstructed region along with image of the region.....	47
A.1 Calibration results for camera 11248.....	49
A.2 Calibration results for camera 11262.....	49
A.3 Calibration results for camera 11218.....	49
A.4 Calibration results for camera 08719.....	50
A.5 Calibration results for camera 11247.....	50
A.6 Stereo calibration results for cameras 11248 and 11262.....	50
A.7 Stereo calibration results for cameras 11262 and 11218.....	51
A.8 Stereo calibration results for cameras 11218 and 08719.....	51
A.9 Stereo calibration results for cameras 08719 and 11247.....	52
A.10 Stereo calibration results for cameras 11248 and 11247.....	52

CHAPTER 1

INTRODUCTION

1.1 Objective

With the advent of the twenty first century, stereoscopic systems have found a widespread use in the engineering industry. Several biomechanical analyses utilize this concept for efficient information extraction. Some examples of its applications are gait analysis, hand shape recognition and facial surface recognition [1].

The primary goal of this thesis was to optimize the existing stereoscopic system, to increase accuracy and precision of the depth information extracted. The process included redesigning the existing equipment set-up, automation of image acquisition unit and modification of conventionally used correspondence test to achieve higher accuracy. The acquired data was used to estimate depth of the object in this study.

The set-up consisted of an array of five cameras surrounding the object being used as a model for reconstruction. The cameras with high image resolution were used to facilitate the image processing tasks. The computing environment chosen for realization of the image processing algorithms was MATLAB. It is a user-friendly environment that incorporates several of the basic image processing algorithms in form of functions. This obviated the need for implementing certain basic functions required during the process. All the results generated during the process were displayed using the graphical options provided by MATLAB.

1.2 The Human Visual Cortex

The concept of a stereoscopic system stems from the model of the visual cortex, which is responsible for binocular vision in primates. The primary visual cortex is said to have a well-defined map of the spatial information. This information is used by primates to perform coordinated motions in the three-dimensional world.

As cited in [2], visual perception is the primary form of cognitive contact between primates and the world around them. Each eye, by virtue of its location, acquires information that is slightly different from the other. This difference is often referred to as the disparity in the scene. The visual cortex is able to make fine depth discriminations from parallax provided by the two eyes.

1.3 Stereovision

Stereovision is derived from the Greek word *stereos*, which means relating to space. Hence, the word essentially means vision in relation with space. This fact was first described by the English genius Charles Wheatstone in 1838. To prove his theory, Charles Wheatstone invented a simple device dubbed as 'Stereoscope'. This device displayed pictures to the left and right eyes separately. It was based on the fact that each individual eye views the same object placed at different distances from slightly different horizontal positions. This in turn gives the depth cue of horizontal disparity to the brain and allows us to perceive depth by triangulation. .

A basic stereoscopic system has, essentially, four modules. They are as follows,

1. Image acquisition
2. Image rectification
3. Correspondence matching

4. Depth estimation

Each module has been discussed in the subsequent chapter. The underlying theory has also been elaborated on.

CHAPTER 2

PRINCIPLES OF STEREOVISION

2.1 Camera Model

Any image processing task is normally preceded by image acquisition. The acquired image is influenced by several factors such as sensor characteristics of the camera, optical characteristics of the lens, characteristics of the light source, material and reflection characteristics of the object surfaces recorded in the scene and last but not the least the geometric laws that govern the image acquisition process [3]. This section reviews the influence of camera geometry on a captured image. An exact knowledge of the camera geometry is crucial for surface reconstruction using static stereo analysis. A mathematical model helps in understanding the fundamentals of this geometry has been presented in the following sub-section. It is called the perspective camera model or the pinhole model.

2.1.1 Perspective Camera Model

As quoted by Olivier Faugeras [4], the perspective (pinhole) camera model (see Figure 2.1) explains a camera from a geometric standpoint. It consists of two screens. The first screen has a small hole punched through allowing a thin ray of light when illuminated. The ray of light forms an inverted image on the second screen, which is placed parallel to the first screen. The first screen represents the *focal plane*, plane F, and the second screen represents the *retinal plane*, plane R. The pinhole present on the first screen is equivalent to the *optical center*, point C, and the ray of light passing through it forms the *optical*

axis. The distance between the two planes is equal to the *focal length* of the optical system.

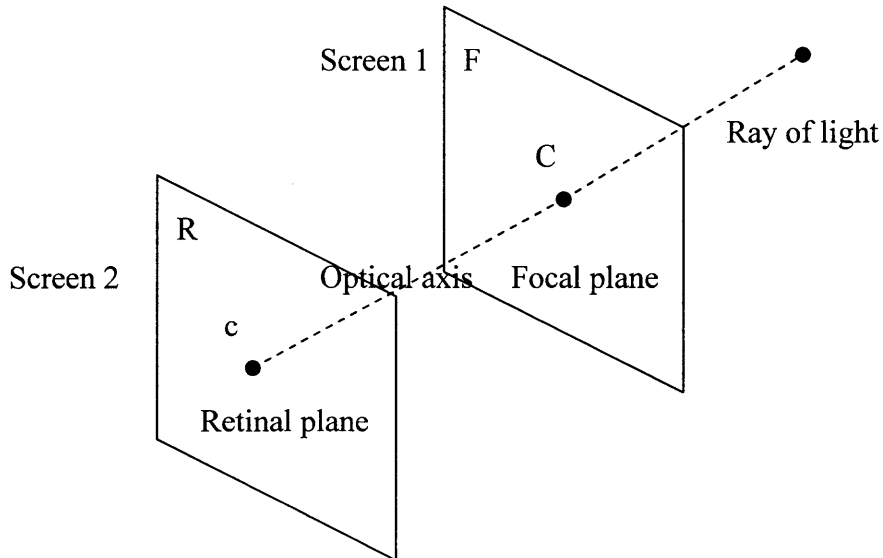


Figure 2.1 The pinhole camera model [4].

2.1.2 Coplanar Stereo Geometry

With the knowledge of the pinhole model, the concept of coplanar stereo geometry can now be explored. Two such models are used to illustrate the concept (see Figure 2.2) [3]. They are assumed to represent left and right cameras. A point $P(X, Y, Z)$ is assumed to lie in the field of view of both the cameras. The optical axes of the cameras run parallel to Z -axis with a direction towards the point. The xy image coordinate axes for both the cameras are parallel to each other. In addition, xy image planes are arranged such that all the rows are identical. The focal point is assumed to lie in the XY plane. Since the orthogonal distance between the focal point and an image plane is the focal length f , the two image planes are now defined by $Z = f$.

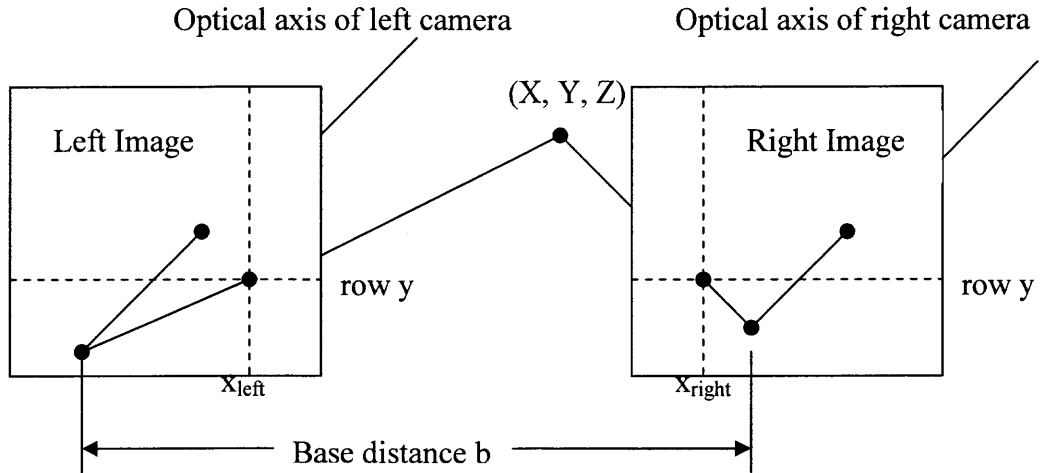


Figure 2.2 Two pinhole camera models are used to illustrate coplanar stereo geometry [3].

Now it is assumed that the XYZ-coordinate system is oriented at the left camera. Thus, the focal point of the left camera lies at $O_{left} (0, 0, 0)$ and that of right camera lies at $O_{right} (b, 0, 0)$. In general, the pinhole cameras will project point P on the two image planes at $p_{left} (x_{left}, y_{left})$ and $p_{right} (x_{right}, y_{right})$, respectively. A new term called *disparity* can now be introduced. It is a measure of difference between the coordinates of two points with one point held as reference. For instance, the *disparity* with respect to p_{left} is found to be $\Delta (x_{left}, y_{left}) = (x_{left} - x_{right}, y_{left} - y_{right})$. This term will be very useful while discussing correspondence and triangulation issues.

Recall that the two image planes were said to lie at $Z=f$ in the world coordinate system. All lines coming from point P are cut by a pair of parallel lines in a certain ratio, which now a relationship between (x, y) , (X, Y, Z) and f .

$$\mathbf{p} = (x, y) = (f \cdot X/Z, f \cdot Y/Z) \quad \dots 2.1$$

Equation 2.1 is applied for the two projected points, namely \mathbf{p}_{left} and $\mathbf{p}_{\text{right}}$, and the resultant is,

$$\mathbf{p}_{\text{left}} = (x_{\text{left}}, y) = (f \cdot X/Z, f \cdot Y/Z) \quad \dots 2.2$$

$$\mathbf{p}_{\text{right}} = (x_{\text{right}}, y) = (f \cdot (X-b)/Z, f \cdot Y/Z) \quad \dots 2.3$$

(Note: since the images are row identical $y_{\text{left}} = y_{\text{right}} = y$)

Notice that for equation 2.3, X has been shifted by b . This is because the optical center of the right image plane is away from the left image plane by distance ‘ b ’ in X direction. Thus, a relationship between the world’s coordinates and the projection coordinates has been established. The above equations will once again be encountered in Section 2.5 when the concept of triangulation is discussed.

2.2 Calibration

As mentioned in the previous section, it is essential that one has prior knowledge of intrinsic and extrinsic camera parameters for an image processing task such as shape recovery. A direct measurement of these parameters is usually technically impossible or difficult. Hence, an indirect measurement of these parameters is required to be made. This process is commonly known as *calibration* [3]. As cited in [4], the aim of calibration is to achieve the best possible correspondence between the used camera model

(parameterized by observed or calculated parameters) and the realized image acquisition with the given camera.

Calibration requires use of objects that have a set of points with known coordinates. These points are usually realized by physically present marks such as dots or crosses. Various examples of a calibration object (see Figure 2.3) are grid, checkerboard, surfaces with equally spaced dots, etc.

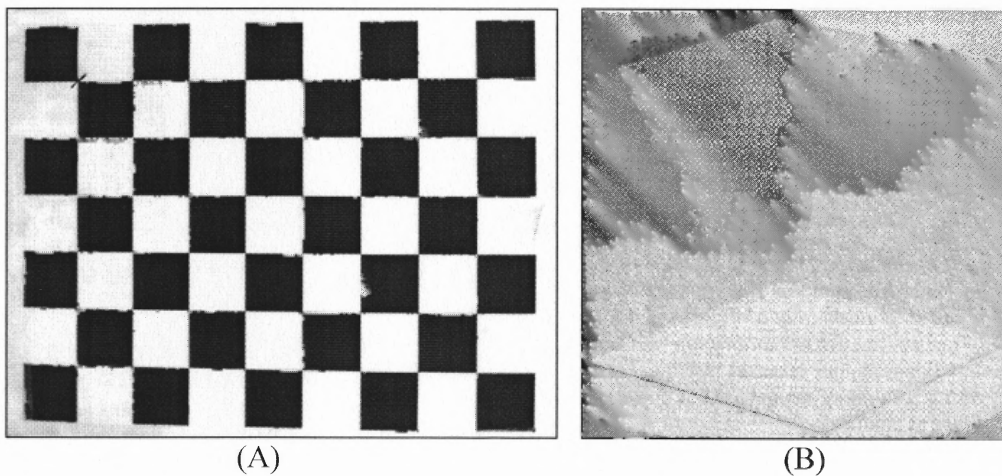


Figure 2.3 Two examples of calibration object (A) Checkerboard (B) Open Cube [3].

It is important that a calibration object is chosen such that it supports the working range of the camera. In other words, all the known points or the calibration points on that object should be visible and distinguishable in the digital images taken by the camera or the image acquisition system used. Sometimes images are taken in several orientations. In such cases, it must be ensured that all the calibration points are visible in every image that has been taken.

Once an image has been acquired, the projected row and column positions of the calibration points are determined. This can be done either by manual selection of points

or by an automated method. After the required information has been extracted, a suitable calibration technique is used for computation of camera parameters. Various techniques exist that serve the purpose. *Direct Linear Transformation* is one of the most widely used techniques [3]. Other very well known method is the *Tsai's Calibration Method* [3]. A method proposed by Janne Heikkilä and Olli Silvén of University of Oulu, Finland has been incorporated in calibration software designed by Dr. Jean-Yves Bouguet of California Institute of Technology [5].

2.2.1 Intrinsic Parameters

As quoted in [3], the *intrinsic parameters* of the camera determine the projective behavior of the camera. These parameters are focal length, principal point, skew coefficient, etc [7] . The definition of the focal length was given in Section 2.1.1. The *principal point* is defined as the intersection of the optical axis and the image plane [3]. The *skew coefficient* accounts for non-orthogonal coordinate axes of an image plane in case of lens distortion.

2.2.2 Extrinsic Parameters

The extrinsic parameters model the transformation from the object coordinates to the camera coordinates [5]. This transformation is described by a set of rotations preceded by translation of the object axes. On calibrating a camera, these alterations are presented in form of a rotation *matrix* (R) and a *translation vector* (T). For a given a set of world

coordinates (X_w, Y_w, Z_w) and camera coordinates (X_c, Y_c, Z_c) , the relation that exists between them can be written as,

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = \mathbf{R} \cdot \begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} + \mathbf{T} \quad \dots 2.4$$

2.3 Rectification

In Section 2.1.2, when the coplanar stereo geometry was discussed, an assumption of row identicalness was made. However, in practical systems such an assumption is not valid. The image is required to undergo some process to achieve the said identicalness. This process is usually known as *Rectification* in a stereo analysis system [4]. With the aid of rectification, the practical image planes can be treated as the ideal case discussed in Section 2.1.2.

To understand how rectification can be realized, one must first get acquainted with the underlying geometrical concept, known as *epipolar geometry*. An illustration of epipolar geometry is provided in Figure 2.4. Subsequently, an explanation of the epipolar geometry is provided. Certain terminology pertaining to this concept very commonly found in literature is explained. The practical application of this concept will be realized in succeeding chapter when the issue of information extraction from given two images is addressed.

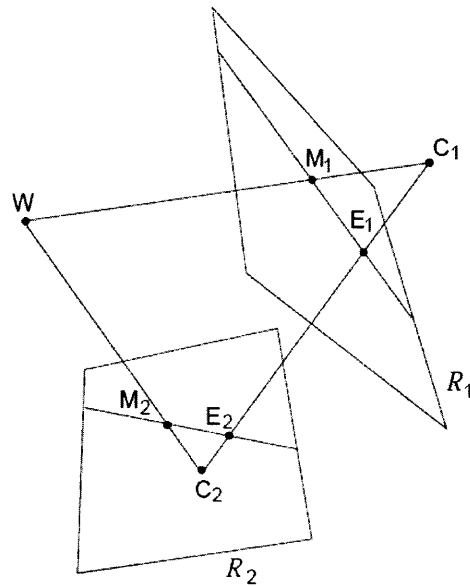


Figure 2.4 An illustration of epipolar geometry [6].

A stereo rig composed of two pinhole models has been assumed. C_1 and C_2 form the optical centers of left and right cameras, respectively. The centers intersect the image planes at points E_1 and E_2 , which are also known as the *epipoles*. A 3D point W is projected on the two image planes; where M_1 and M_2 are the projected points. Lines E_1M_1 and E_2M_2 are known as the *epipolar lines*. It is apparent that any arbitrary point lying on line WC_1 will be constrained to be projected on line E_2M_2 . This constraint is termed as the *epipolar constraint*. When C_1 and C_2 lie in the focal planes for both cameras, the epipolar lines form a bundle of parallel lines in both images. The procedure of transforming the epipolar lines to a bundle of parallel and horizontal lines is known as *Rectification*. Any pair of images can be rectified if the camera parameters are known.

2.4 Correspondence Matching

While discussing stereo geometry, the concept of paired projection points was repeatedly encountered. The importance of this pair was realized in Section 2.1.2, where equations 2.2 and 2.3 established a relation between the said pair and world coordinates. It should be noted that the only parameters required, besides focal length and base distance, to completely recover world coordinates of a point are projected coordinates and disparity. Hence, if one is able to determine the corresponding pairs for every point on an image, the shape of the object captured can be reconstructed. This analysis is, often, referred to as *correspondence matching*. Since this stage leads to actual shape recovery, it can be regarded as the prime stage of stereo analysis.

The basic premise of correspondence matching is to find pair of points that have a unique match. In other words, for a point on one of the given set of image planes, there should be one and only one point found on the analyzed plane that will be declared as the conjugate of the assumed point. For practical implementation of this analysis, one needs to extract information that is unique to every point on the image. Two methods have assumed wide recognition for this form of information extraction [3]; namely intensity-based correspondence analysis and feature-based analysis.

2.4.1 Intensity Based Correspondence Analysis

For correspondence analysis, it is assumed that corresponding points or *pixels* in given two images have similar intensities. However, such similarity may exist between several of the pixels present on the image. Hence, a better approach would be the use of neighboring pixels along with the point for comparison. Now an intensity measure of this

block of pixels can be used for search of corresponding value. Usually, the sum of intensities of all the pixels present in the block is used as the measure. This unique identification measure used is also referred to as a *token* [4]. This measure can now be compared with a congruent block formed around every pixel of the conjugate image. The block with the best match is supposed to contain the corresponding pixel.

2.4.2 Feature Based Correspondence Analysis

The basic concept of a feature based analysis is the same, except that the token used is feature information of the block. Features such as edges, intensity variations are, often, used for comparison. These features are, however, not very apparent in an image. Therefore, the image is pre-processed to make them more evident. Edge detection or high pass filtering are normally the techniques used to enhance the features.

2.4.3 1D Correspondence Search

The search for correspondence over an entire image is an onerous task. It consumes a huge amount of processing time. In addition to that, the probability of arriving at a false match increases. For a practical stereo system, numerous such searches are required to be performed for shape recovery. The process time and number of false matches is, thereby, multiplied. This is where the application of rectification is realized. Since rectification aligns rows of the image planes, the search for correspondence is now reduced to 1D. For a given point, the conjugate is now searched along one row, which is given by the y coordinate of the reference point.

2.5 Triangulation

Triangulation marks the culmination of shape recovery. Some processing may take place later but it is merely to improve the appearance of the reconstructed object. To understand the concept of triangulation, the coplanar stereo geometry model reviewed in Section 2.1.2 is used once again.

Equations 2.2 and 2.3 are used to establish a relation between the world coordinates and projected image coordinates [3]. If the values of Z obtained in the equations above are compared, the result is,

$$Z = f \cdot X / x_{\text{left}} = f \cdot (X - b) / x_{\text{right}}$$

And therefore,

$$X = b \cdot x_{\text{left}} / (x_{\text{left}} - x_{\text{right}}) \quad \text{or} \quad X = b \cdot x_{\text{left}} / d \quad \dots 2.5$$

Where d = disparity of the coordinates.

Using equation 2.5, it can now write,

$$Z = b \cdot f / d \quad \dots 2.6$$

These results are now substituted in equation 2.1, which gives the value of Y .

$$Y = b \cdot y / d \quad \dots 2.7$$

Equations 2.5, 2.6 and 2.7 give the mathematical relation between point P in the world coordinate system and projected points. It can be seen that coordinates of a point P

are a function of the focal length, baseline, disparity and projected point on the left image plane. Knowledge of all the above mentioned parameters allows one to recover the original location of the projected points and, hence, recover shape. It should be noted that the focal length is an intrinsic property of the camera optical system and is constant for a given camera system. The base distance, also referred to as the *baseline*, is required to be set for a particular study. To determine the disparity for a set of points, one must have good knowledge of their corresponding points in the adjacent image plane. This is an issue of correspondence matching and was dealt with in Section 2.4.

CHAPTER 3

EXPERIMENTAL SET-UP AND PROCEDURE

3.1 Experimental Set-Up

3.1.1 Optimum Base Distance Calculation

In the previous chapter, the concept of stereovision was discussed along with its underlying theory. It is evident that the transformation of any point in the world coordinate system to a camera coordinate system is a function of the intrinsic and extrinsic parameters of the camera. Two parameters explicitly seen in the equations of triangulation are focal length and base distance. Focal length of a camera is always constant. It is an inherent property of the camera optical system and cannot be changed. However, the base distance is required to be designed before the apparatus is arranged.

The design of the base distance is very critical, since its magnitude affects two important aspects of the stereovision process; namely correspondence determination and depth estimation. A wide baseline provides better depth estimation with a compromise on the accuracy of correspondence match. The effect is reversed when the base distance is reduced [8].

In this section, the base distance design for camera arrangement used for this thesis has been provided. The first step of the design is to pick two points such that the desired depth resolution is acquired. For this thesis, it is desired that a depth of 1 mm is represented by a disparity difference of 1 pixel. The design is done iteratively and begins with an initial guess of base distance.

Let the base distance for the first trial be 50cm between cameras located at the extreme ends. Two points are assumed for the calculation; point A located at 75 cm from center of the set-up and point B located 1mm away from point A in z-direction. If the disparities found for points A and B are d_A and d_B , respectively, then the corresponding Z-depths will be given as,

$$Z_A = b.f / d_A \quad \dots 3.1$$

$$Z_B = b.f / d_B \quad \dots 3.2$$

Since the values of b , f , Z_A and Z_B are known, they can be substituted in Equations 3.1 and 3.2 to determine the magnitudes of the disparities. These values are found to be 10mm and 9.98668mm, respectively. The difference between the disparities is, hence, found to be 0.01331mm. The pixel pitch for a Pixelink A642 camera is found to be $6.7\mu\text{m}$. Therefore, the disparity in pixels is found to be approximately 2. However, it is desired that the disparity be 1 pixel for a 1mm change. Hence, the baseline is reduced to 25cm. Consequently, the disparity change is found to be closer to 1 pixel. Thus, it is inferred that a baseline between 25cm and 50cm is optimal for the given set-up. The choice should be made taking the camera dimensions into consideration. The entire set-up, including the chosen baseline, has been discussed in Section 3.1.2.

3.1.2 Equipment Set-Up

The experimental set-up (see Figure 3.1) consists of an array of five cameras (1, 2, 3, 4 and 5) firmly mounted on metal brackets. They are arranged such that they surround a Halloween mask, which is the object being used for 3D reconstruction. The perpendicular

distance between the mask and center camera is about 750mm (29.52”) and that between the two cameras on extreme ends is approximately 362mm (14.25”). The perpendicular distance between two adjacent cameras is 90mm (3.5”). The cameras are oriented such that the object appears in the center of FOV (field of view) of every camera.

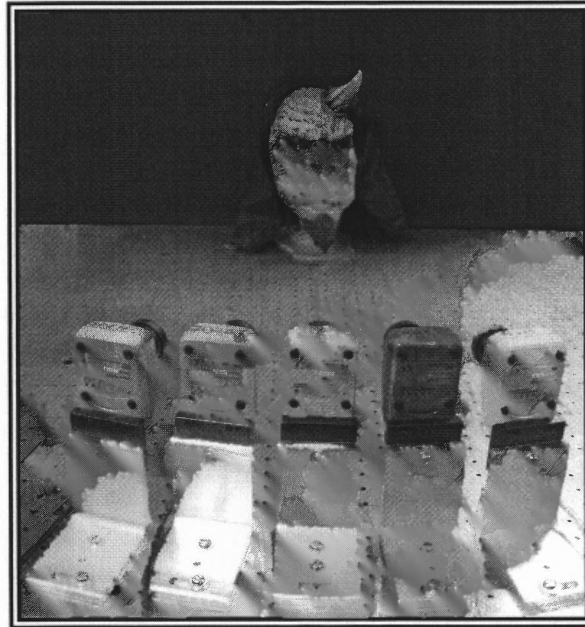


Figure 3.1 The experimental set-up being used for the project.

The cameras are daisy chained (see Figure 3.2) and connected to an IEEE 1394a (FireWire) port present on a high processor computer. The captured images are acquired using algorithms written in programming environment known as MATLAB (version 2007a) created by Mathworks Inc., MA, USA. The algorithms use a software interface between the cameras and MATLAB developed by M.A.E Bakker and L.I. Oei of Czech Technical University in Prague, Czech Republic, which is available on the internet for free download.

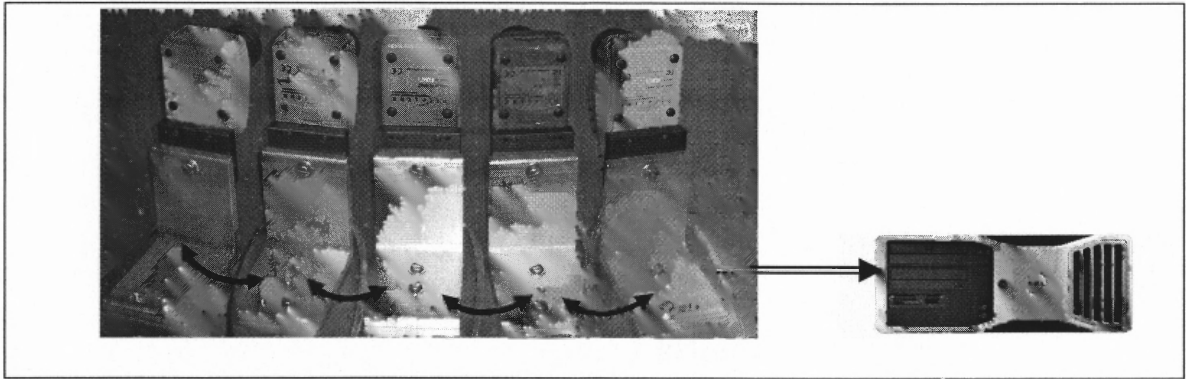


Figure 3.2 Daisy chained cameras connected to a computer.

Each camera was calibrated individually and in adjacent pairs before acquiring images. Calibration was performed using the Camera Calibration toolbox for MATLAB designed by Professor Jean-Yves Bouguet of the California Institute of Technology. This eliminated the need of using time consuming calibration techniques providing quick and reliable results. The toolbox also allows rectification and warping of the images.

3.2 Hardware Profile

This section describes the hardware used for the experiment, namely the cameras and the computer system. The first sub-section (see Section 3.2.1) talks about the features and the computer hardware requirements for the camera. The following sub-section (see Section 3.2.2) talks about the computer system incorporated in the experiment.

3.2.1 Cameras

The camera model used for this experiment is a Pixelink A642 (see Figure 3.3) manufactured by Vitana Corp., Ottawa, Canada. The imager resolution of this model is 1.3 Megapixel (1280 x 1024) and can capture images in 10-bit RGB or monochrome. An integrated infra-red filter is present over the image sensor. Each camera has two FireWire connectors. Because of which a camera can either be connected solely or in a daisy chain with other cameras. The focal length of the lens is 16mm.

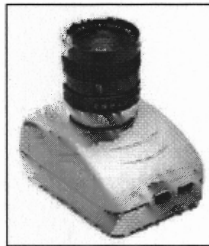


Figure 3.3 A Pixelink A642 camera.

The recommended hardware configuration of the computer for optimal functioning of the camera is as follows:

1. Microprocessor speed: 1.5 GHz.
2. Random access memory: 128 MB.
3. Video card: 24-bit/32-bit with true color graphics capability with at least 8 MB of video memory.
4. Hard drive space: 30 MB (minimum).
5. Desktop resolution: 1280 x 1240.
6. A built-in FireWire port or a preinstalled FireWire PCI card.

3.2.2 Computer System

The computer model used in this study was Dell Precision 390. This machine is equipped with an Intel ® core™ 2 duo processor with a clock speed of 2.4 GHz. A 3.25 GB of RAM allows multiple processes to be run without any issue. The hard drive capacity is very high and the desktop screen resolution is optimum. However, the machine has just one IEEE 1394 (Fire Wire) port. Additional ports were made available by installing a Fire Wire PCI card in the machine.

3.3 Software Profile

In this section, the software packages used at various stages of the thesis have been introduced. First we take a look at the main computing environment used for the development of all the algorithms used in this experiment. This is followed by the Camera calibration toolbox used at the preliminary stage of calibration.

3.3.1 MATLAB (version R2007a)

MATLAB, short for matrix laboratory, is a numerical computing environment and computing language developed by Mathworks Inc. at MA, USA. The basis of this software is matrix manipulation, which allows us to perform a wide range of mathematical computations on given data by expressing it in form of a matrix or a vector. A wide array of built-in functions allows a user to acquire, analyze and plot various forms of data. One can also implement a user-defined algorithm using the various functions

present. Besides basic data processing functions, MATLAB also incorporates dedicated sets of functions known as toolboxes. Each toolbox contains functions pertaining to a specific acquisition or processing method. For instance, the image processing toolbox consists of functions that perform edge enhancement, 2D filtering of images, histogram manipulations, etc. Other toolboxes that are often found are signal processing, statistical analysis, data acquisition, control systems, virtual reality, etc.

A typical MATLAB environment (see Figure 3.4) shows a command window, where all the functions present can be invoked. A history of such commands can be seen in the command history window. All the variables created during the process are shown in the section of workspace.

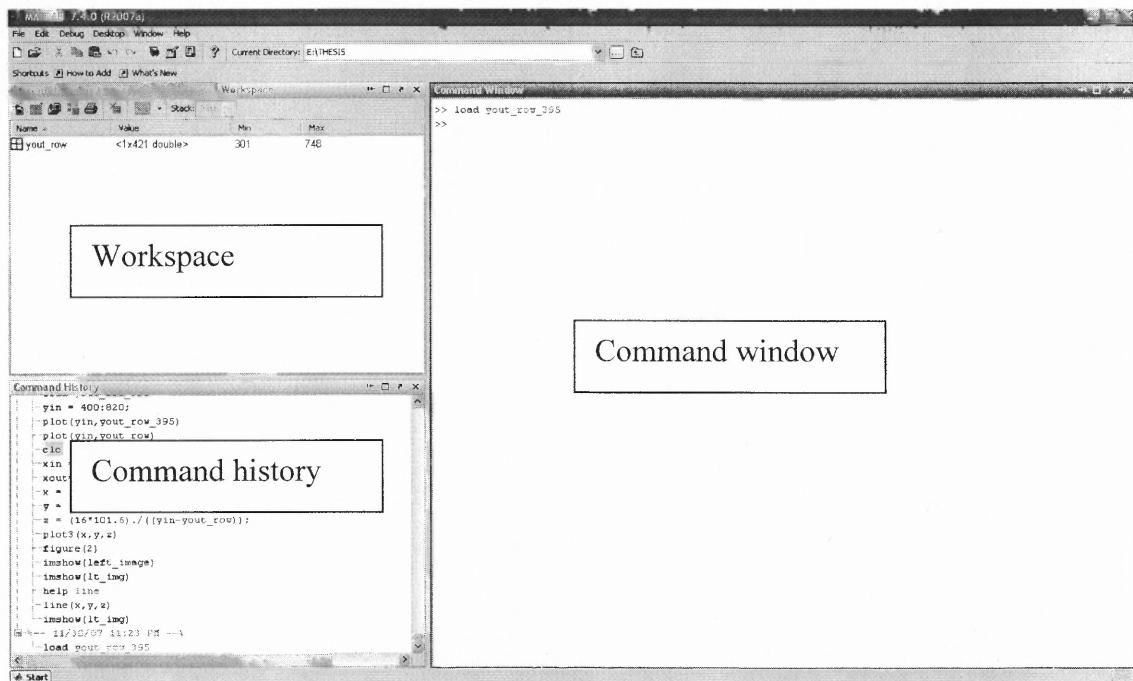


Figure 3.4 A typical MATLAB environment.

3.3.2 Camera Calibration Toolbox for MATLAB

The camera calibration toolbox for MATLAB designed by Professor Jean- Yves Bouquet is a simple and a user-friendly toolbox, which is easy to use and produces good results. The toolbox works very well with MATLAB 5.x and higher versions. It does not require any specific toolbox for its functioning.

This toolbox reads images, extracts corners and computes the intrinsic and extrinsic parameters. It has a graphical user interface (GUI) (see Figure 3.5) that makes selection of each function very simple. This GUI can be invoked by typing `calib_gui` at the command prompt of MATLAB command window. The calibration results are stored in a .mat form in the current directory of MATLAB allowing the user to access them when needed. This toolbox is essentially used for calibration of individual cameras.

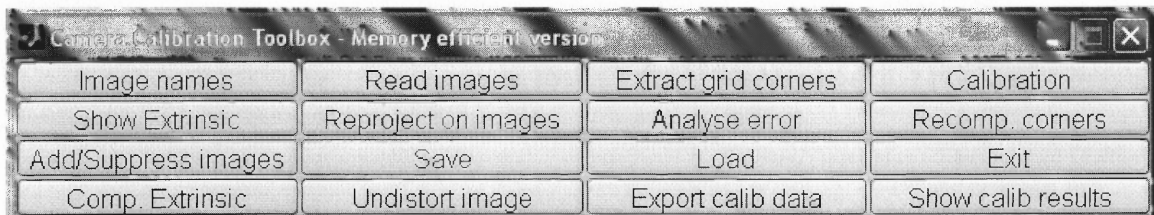


Figure 3.5 The graphical user interface of the camera calibration toolbox.

In order to calibrate a pair of cameras, the stereo calibration toolbox (see Figure 3.6) is required to be used. This toolbox has also been designed by providers of camera calibration toolbox. The GUI for the same can be called by typing `stereo_gui` at the command window. Whilst performing stereo calibration, the .mat files generated during calibration of individual cameras are read. The user has to ensure that correct entries for left and right calibration results are made. Clicking on the run stereo calibration button

performs the calibration using the read calibration files. In addition to the aforementioned functions, this toolbox can also be used for rectification of images.

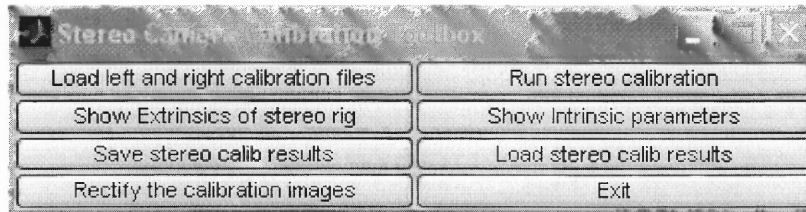


Figure 3.6 The graphical user interface for the stereo camera calibration toolbox.

3.4 Procedure

The aim of a static stereo analysis is essentially to extract depth information and recover shape. The process begins with acquisition of images, which can be followed by either calibration of cameras or rectification of images. Later case is usually encountered when the cameras have been pre-calibrated. For most static systems, calibration is not required for every reconstruction and hence, the process begins with rectification. The rectified images are then analyzed for correspondence matching using a suitable relation technique. The process is concluded by using the correspondence data to estimate depth and recover the shape of the scene. The subsequent sub-sections address each of the processing steps.

3.4.1 Image Acquisition

Any image processing technique is always preceded by image acquisition. There are various modes of acquiring images. They can either be captured by an analog camera and then converted to a digital format or by a digital camera and transferred to a computer.

Here the camera system is directly interfaced with the computer system and a software package is required to be used to trigger image capture. The Pixelink A642 comes with a software bundle that allows the user to acquire images. The software identifies each camera by its hardware serial number allowing the user to access one camera at a time for acquisition. Thus, several images can be captured accessing the cameras sequentially.

However, a stereo vision system requires that all the cameras capture the scene simultaneously, especially for calibration. Hence, a script was developed in MATLAB using the Pixelink interface to do the needful. In previous studies, the Pixelink capture software was used for the purpose, thereby compromising on the accuracy of the results. To use the above mentioned script, one needs to type *grab_image* at the command prompt. The function needs the hardware serial number to be passed as parameter. On execution, the function opens the camera, sets the parameters required for a good quality image and grabs it. The entire function has been provided in Appendix B.

3.4.2 Camera Calibration

The purpose of calibrating cameras is to establish a mathematical model defining its behavior. The need and underlying theory has already been reviewed in Section 2.1. As mentioned before, the camera calibration toolbox for MATLAB was used for this purpose. A planar checkerboard of size 196 x 252 mm (see Figure 3.7) was used as a calibration grid.

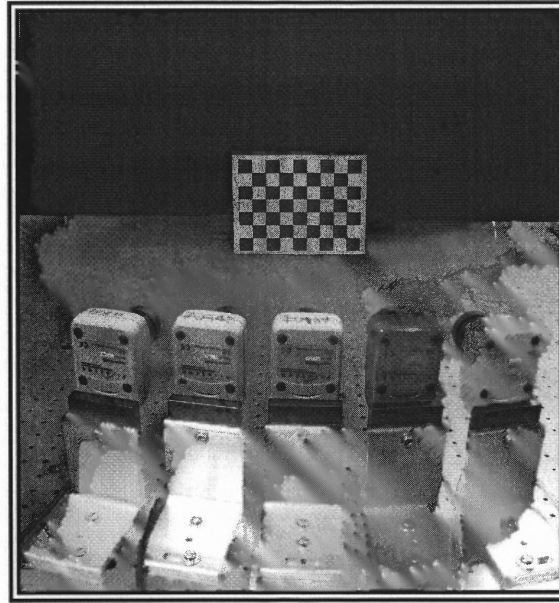


Figure 3.7 Planar checkerboard placed for calibration process.

Before initiating calibration, one must acquire several images of the grid in varied orientations. This enables allows the software to locate more corners, thereby giving output with higher accuracy. It is recommended that a minimum of fifteen images be used for the procedure. A special code for acquiring multiple images simultaneously from all cameras was developed for this experiment. This script can be called by typing *calib_images* (see Appendix B) at the command prompt of MATLAB. On execution, the code requires the user to input the number of cameras being used, the number of images required to be grabbed and the hardware serial numbers in order of their positions. After the required entries have been made, the code begins acquiring images. Thus, using this code twenty images were captured for calibration (see Figure 3.8).

Set of images used for calibration of Camera no. 2

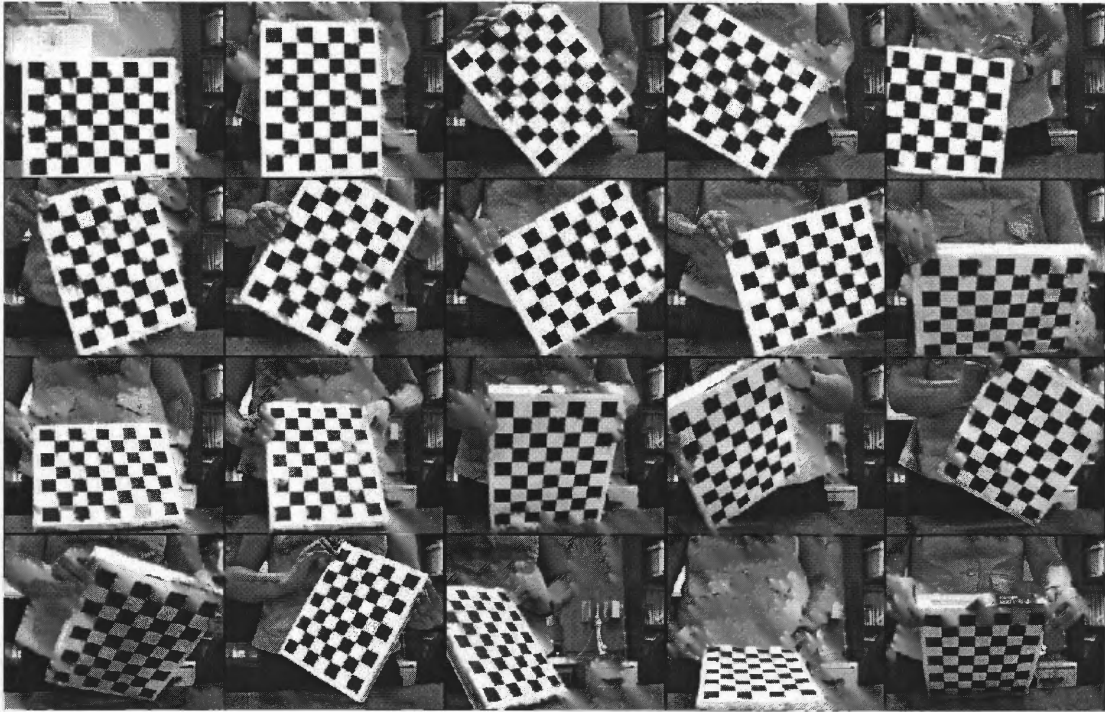


Figure 3.8 Images captured by camera 2 for calibration.

After image acquisition, the process of calibration ensues. Firstly, all the cameras are required to be calibrated individually using the camera calibration toolbox. Once that has been accomplished, the stereo calibration toolbox can be used to calibrate the cameras in pairs. As mentioned in Section 3.3.2, GUI for the toolbox can be opened by typing *calib_gui* at the command prompt. The first step in calibration is to read the acquired images. This is achieved by hitting the *Read images* button on the GUI window. The user is required to specify the base name and format of the image being read. After all the images have been read, corner extraction is performed. For this, the *Extract grid corners* button should be clicked. The user is prompted to enter the values for search window size, exact size of one square on the grid being used and number of square in X

and Y directions. The default window size of 11x11 has been used for calibration. Each image is then flashed prompting the user to select the origin and remnant three corners. It is essential that the origin chosen for the first image coincides with the origins on the subsequent images. Failure of which might result into erroneous results. After every image has been subjected to corner extraction, the user is required to click on the *Calibration* button. Now the toolbox computes the intrinsic and extrinsic parameters based on data acquired during corner extraction. At the culmination of the computation, the intrinsic parameters are displayed at the command window of MATLAB (see Figure 3.10). These parameters can also be accessed by hitting the *Show calib results* button.

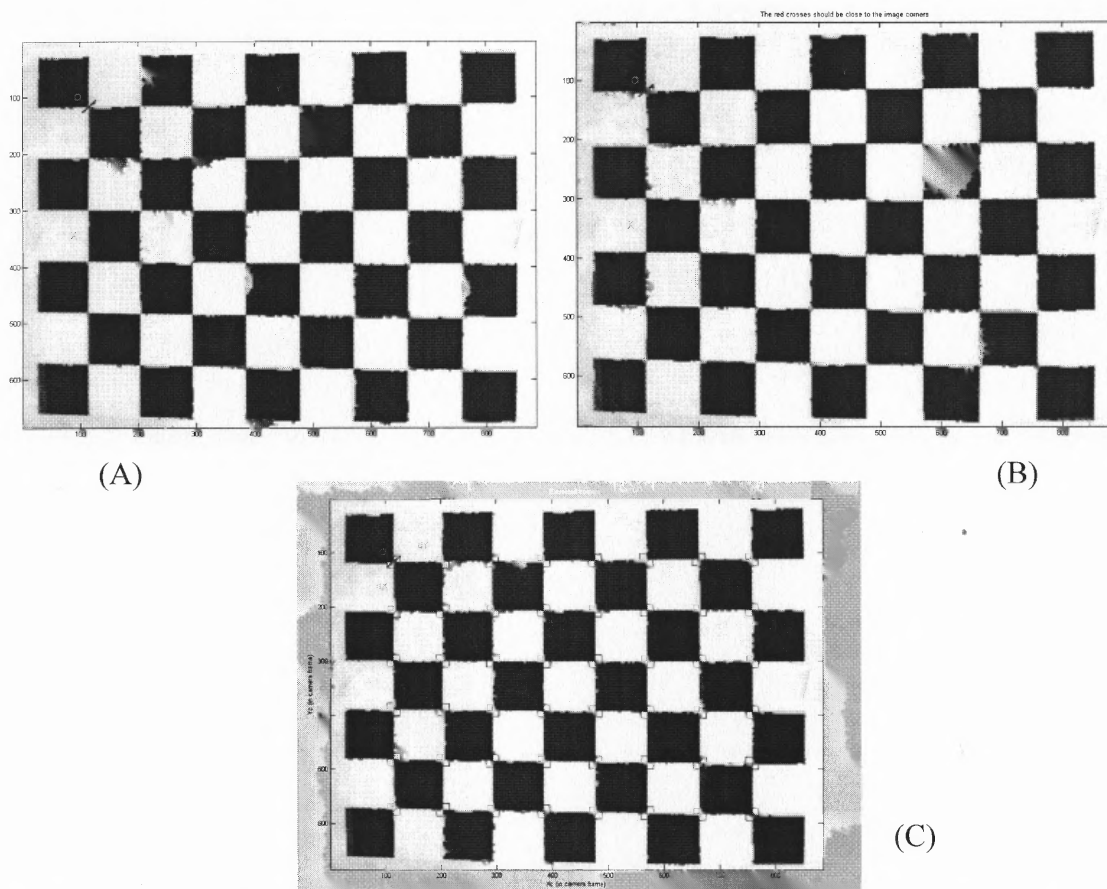


Figure 3.9 (A) Manual grid corners selection (B) Automated corner extraction (C) Automated square counting by the camera calibration toolbox.

```

Command Window

Calibration results (with uncertainties):

Focal Length:      fc = [ 2236.20153  2246.69915 ] ± [ 36.08792  38.15334 ]
Principal point:   cc = [ 693.61176  453.53580 ] ± [ 56.86187  48.79746 ]
Skew:              alpha_c = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:        kc = [ -0.32148  0.41688  -0.00377  -0.00437  0.00000 ] ± [ 0.06801  0.64909  0.00319  0.00396  0.00000 ]
Pixel error:       err = [ 1.18107  1.34656 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).

>>

```

Figure 3.10 The intrinsic parameters for camera 2 displayed at the command window in MATLAB.

Following the calibration of individual cameras, calibration of adjacent pairs should be performed. This is done by using the stereo calibration toolbox, which is launched by typing *stereo_gui* in the MATLAB command window. The process requires loading the calibration results of the camera pair being calibrated. The program prompts the user to enter the results for the left and the right camera. After having entered the appropriate files, the calibration process is initiated by clicking on *Run stereo calibration*. The program then computes a new set of parameters relevant to a stereo rig, such the rotation and translation vectors. The output is displayed on the command window (see Figure 3.11). These results are saved in working directory as well for future use. The extrinsic information of the stereo rig can be seen by hitting the *Show Extrinsic of stereo rig* button (see Figure 3.12). All the calibration results have been included in Appendix A.

```

Command Window
File Edit Debug Desktop Window Help

Stereo calibration parameters:

Intrinsic parameters of left camera:

Focal Length:      fc_left = [ 2342.76754  2337.57415 ] ± [ 48.56937  49.08763 ]
Principal point:   cc_left = [ 678.23712  513.92506 ] ± [ 75.23539  69.02691 ]
Skew:             alpha_c_left = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:       kc_left = [ -0.48939  1.91228  -0.00101  -0.00721  0.00000 ] ± [ 0.16503  2.04804  0.00487  0.00432  0.00000 ]

Intrinsic parameters of right camera:

Focal Length:      fc_right = [ 2270.54995  2276.86840 ] ± [ 44.48345  44.86809 ]
Principal point:   cc_right = [ 553.85275  374.64423 ] ± [ 75.55653  72.68366 ]
Skew:             alpha_c_right = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:       kc_right = [ -0.16525  -0.65453  -0.00015  0.00103  0.00000 ] ± [ 0.14664  1.40131  0.00656  0.00532  0.00000 ]

Extrinsic parameters (position of right camera wrt left camera):

Rotation vector:   om = [ -0.04794  0.52459  0.00116 ] ± [ 0.03390  0.04299  0.01352 ]
Translation vector: T = [ -362.99705  6.01253  81.30696 ] ± [ 7.32774  3.16600  19.58333 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).

>>

```

Figure 3.11 The output of calibration results for camera 1 and 4.

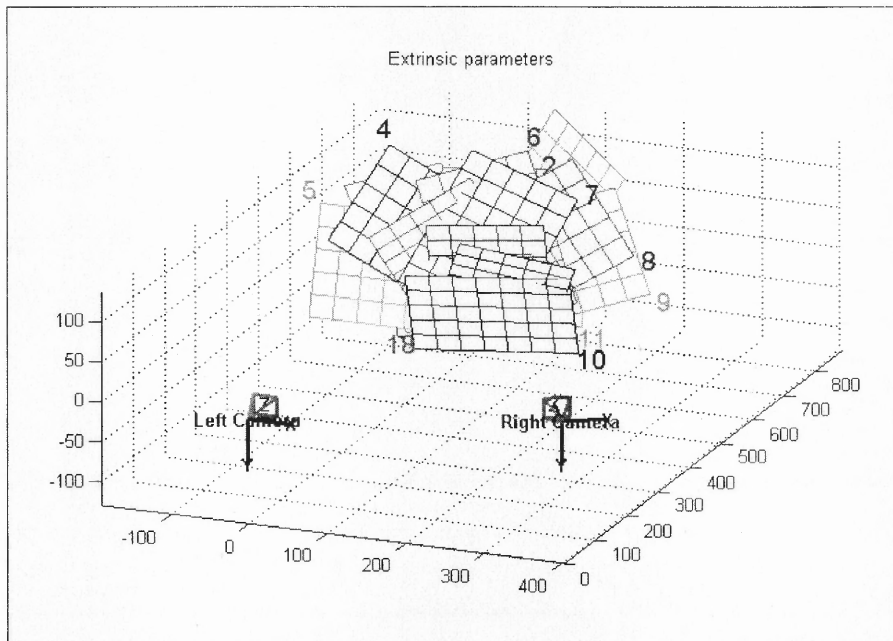


Figure 3.12 The graphical depiction of the extrinsics of a stereo rig computed for camera 1 and 4.

3.4.3 Image Rectification

Rectification is a key step in simplifying the process of 3D reconstruction. By aligning the epipolar lines of two images, the search for correspondences is reduced from 2D to 1D. Rectification of two images is achieved using the stereo calibration toolbox. It requires that the images be renamed with same base name as the images used for calibration. The process starts after clicking on *Rectify the calibration images* button. The rectified images are stored in the working directory with a suffix *_rectified* followed by the number of the image. An example of rectified images from cameras 1 and 4 has been shown in Figure 3.13.

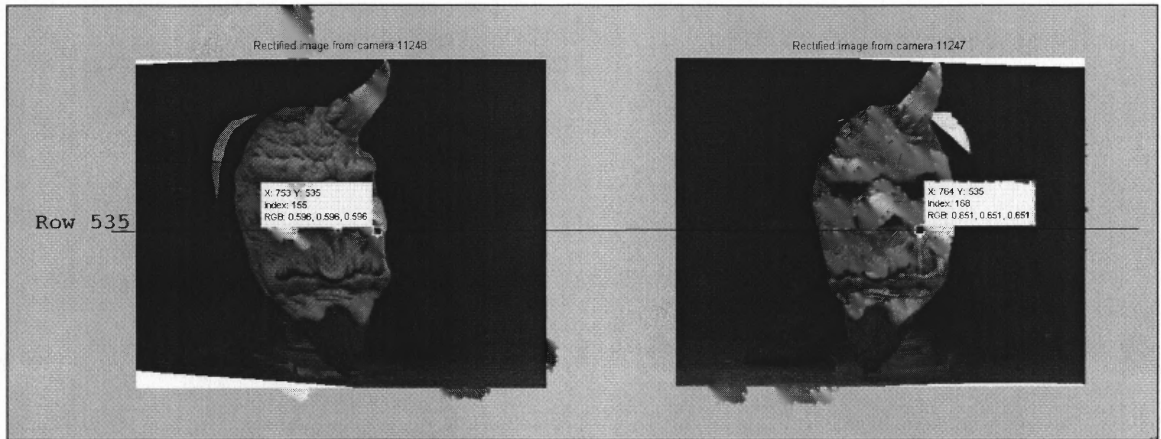


Figure 3.13 A subplot of rectified images for cameras 1 and 4 has been shown with row 535 highlighted for illustration.

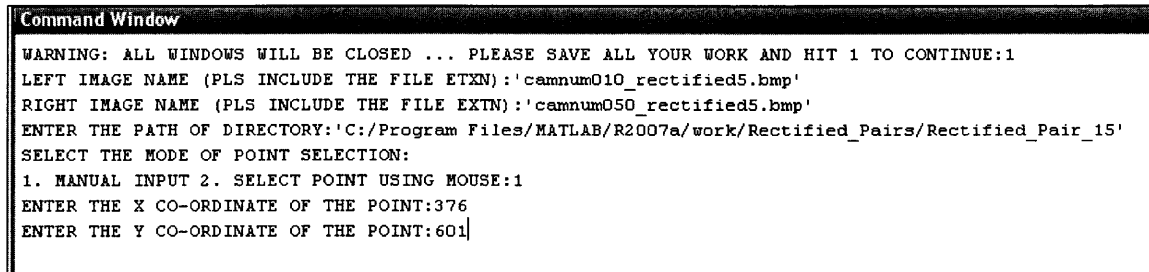
3.4.4 Correspondence Analysis

With correspondence matching begins the actual process of stereovision. The similarity between two regions being compared is the selection criterion for this analysis. As mentioned earlier, the information used for comparison or a *token* can either be the

intensity variation of the area being compared or the feature variation. Former being referred to as an area based method and later as feature based. In this thesis, the area based method has been chosen for comparison. However, it was observed that the matching blocks in given two images did not always have the same intensity dynamic range. As such, the algorithm failed to detect correspondences having similar pattern but different dynamic range. The inconsistencies found can be attributed to several factors such as non-uniform illumination in vicinity of the object, reflective surfaces present around object, etc. This issue was resolved by modifying the conventional intensity based analysis. For every comparison, the current block being tested was equalized using the histogram of the reference block. This operation resulted in comparable dynamic ranges for blocks being matched. The pattern of the pixels was now more distinctly visible. Secondly, it was found that use of window size higher than 10×10 produced results with higher precision. The technique chosen for comparison of tokens was normalized correlation. This method was chosen on the basis of previous study, conducted by Navneetha Shriram [9], which said that this particular technique provided accurate results for almost all cases tested.

In order to test the above mentioned method, two scripts were developed that tested single pixel correspondence for a pair of images. The main script is present under the name of *corr_code* and the second script is a function, *corr_image*, which performs the required computation and matching. On execution, the code prompts the user to enter the names of the images to be used for comparison as well as the path to the directory in which they are located. After MATLAB has read the images, user is required to select the point whose correspondence is to be found either by manually entering the coordinates or

by selecting the point by a mouse click (see Figure 3.14). This code also allows the user to select the window size. It is recommended that a size above 10 x 10 be used for higher accuracy in results.



```
Command Window
WARNING: ALL WINDOWS WILL BE CLOSED ... PLEASE SAVE ALL YOUR WORK AND HIT 1 TO CONTINUE:1
LEFT IMAGE NAME (PLS INCLUDE THE FILE EXTN):'camnum010_rectified5.bmp'
RIGHT IMAGE NAME (PLS INCLUDE THE FILE EXTN):'camnum050_rectified5.bmp'
ENTER THE PATH OF DIRECTORY:'C:/Program Files/MATLAB/R2007a/work/Rectified_Pairs/Rectified_Pair_15'
SELECT THE MODE OF POINT SELECTION:
1. MANUAL INPUT 2. SELECT POINT USING MOUSE:1
ENTER THE X CO-ORDINATE OF THE POINT:376
ENTER THE Y CO-ORDINATE OF THE POINT:601
```

Figure 3.14 The series of prompts that appear on the command window after *corr_code* is executed.

Once the coordinates have been provided, a window of the specified size is placed around the point of interest. Two computations are performed on this block; sum of pixel intensities and histogram of the block. Now a window of the same size is moved over epipolar row of the second image. For every pixel, the surrounding block is histogram equalized and then summed for intensities. The two sums are then checked for correlation. The pixel with highest correlation is declared as the corresponding point. Figure 3.15 depicts improvement in information due to histogram equalization.

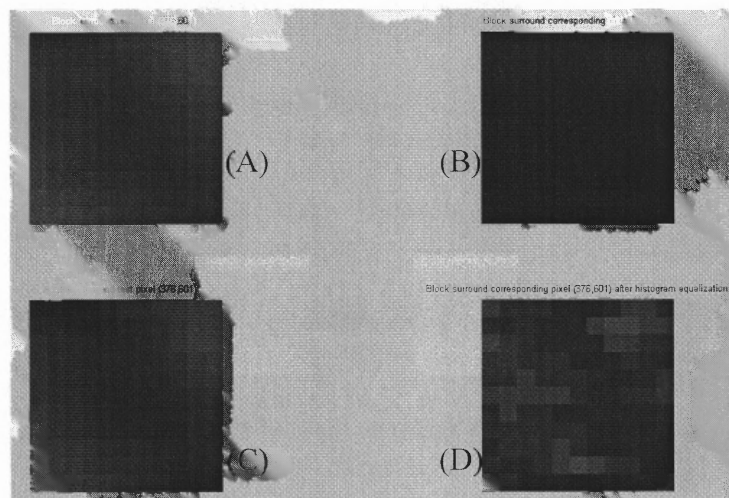


Figure 3.15 (A) and (C) Block surrounding test pixel at (376,601). (B) Block surrounding corresponding pixel at (376,640) in second image. (D) Histogram equalized block surrounding corresponding pixel at (376,640).

After the computation is concluded, the code outputs the detected match (see Figure 3.16) in the MATLAB command window. A graph with the plot of correlation values against the y coordinates is also displayed (see Figure 3.17). The result can be verified by visual selection of the corresponding point. For the given point, the matching pixel was found at (376, 639) (see Figure 3.18). This method was tested by five different points illuminated by a laser for five different pairs of cameras. The results for the same can found in Section 4.1.

```

Command Window
ENTER THE WINDOW SIZE IN X (>) DIRECTION:11
ENTER THE WINDOW SIZE IN Y (^) DIRECTION:11
Elapsed time is 15.082886 seconds.
The pixel match for (376,601) was found at (376,640)
>> |

```

Figure 3.16 The result of the correspondence match test for pixel at (376, 601) is displayed in the command window.

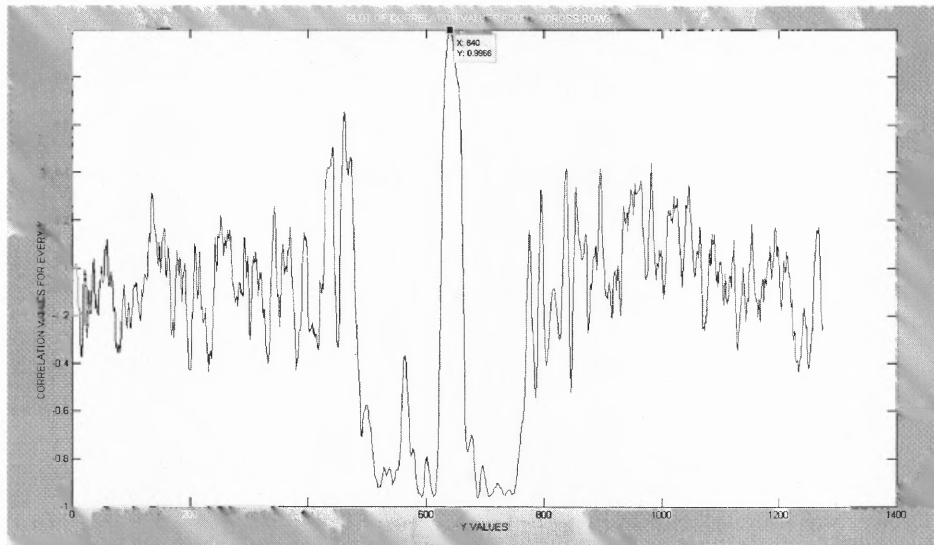


Figure 3.17 The plot of correlation values along row 376 shows a large spike at $y = 640$.

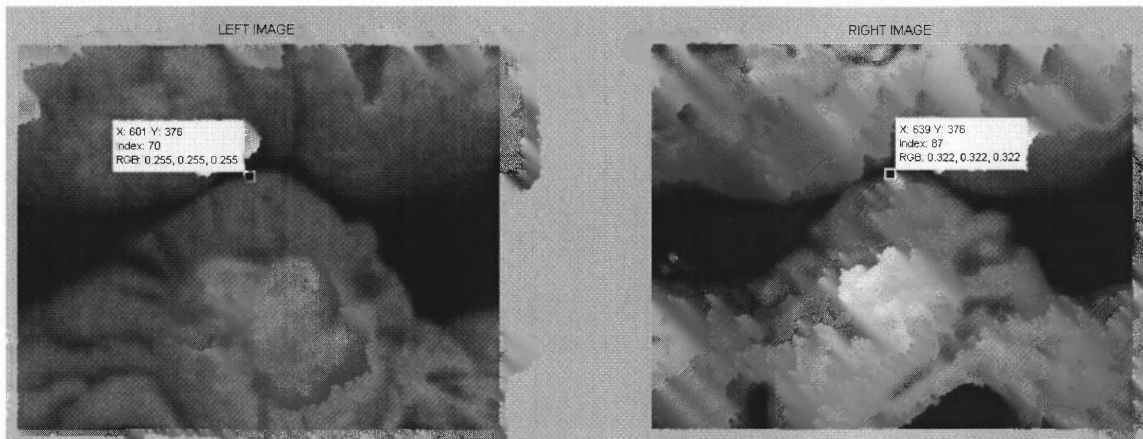


Figure 3.18 Visual inspection of the point yields the matching point at (376,639).

This analysis is further extended to determination of conjugates for every pixel present in the row being scanned. A plot of conjugate pairs is generated to visualize the trend of the curve. Figure 3.17 shows the curve obtained for camera pair 2 and 3 at row 490.

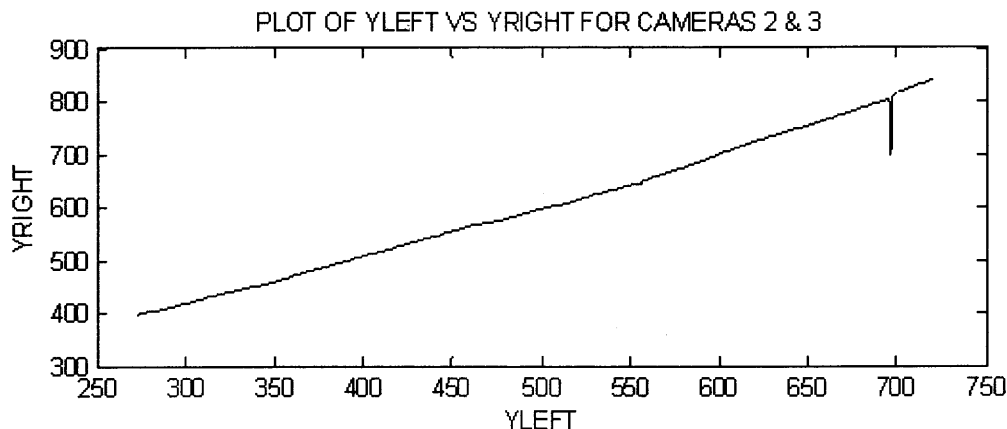


Figure 3.17 Trend of the conjugates observed in row 490 for camera pair 2 and 3.

The plot of the conjugates displays a seemingly linear trend. However, the textural variations are evident where the plot is investigated in small portions. The glitch located towards the end of the curve represents the sudden change in intensity due to a boundary. Hence, such plots would provide a mode of discriminating correspondences. It is anticipated that these trends would also help in smoothening the curves obtained for other camera pairs.

3.4.5 Depth Estimation

The final step towards surface reconstruction is depth estimation. It is realized by using the concept of triangulation discussed in Section 4.5. The complete set of parametric data required for computations involved, namely focal length, base distance and disparity, is available for use after the correspondence analysis. The necessary computations are performed by a script under the name *shape_recover* developed in MATLAB. This script can be found in Appendix B.

The script reads the data for conjugate pairs made available during correspondence analysis. It calculates the row disparity for every camera pair and substitutes it in triangulation equations to calculate the approximate depth. The accuracy of the depth is subject to the accuracy of the correspondences. Besides the depth, the corresponding column coordinates of the points are calculated. A 3D plot of the row and column coordinates against the depth gives an approximate shape of the row. Figure 3.18 shows an example of a reconstructed segment of the object.

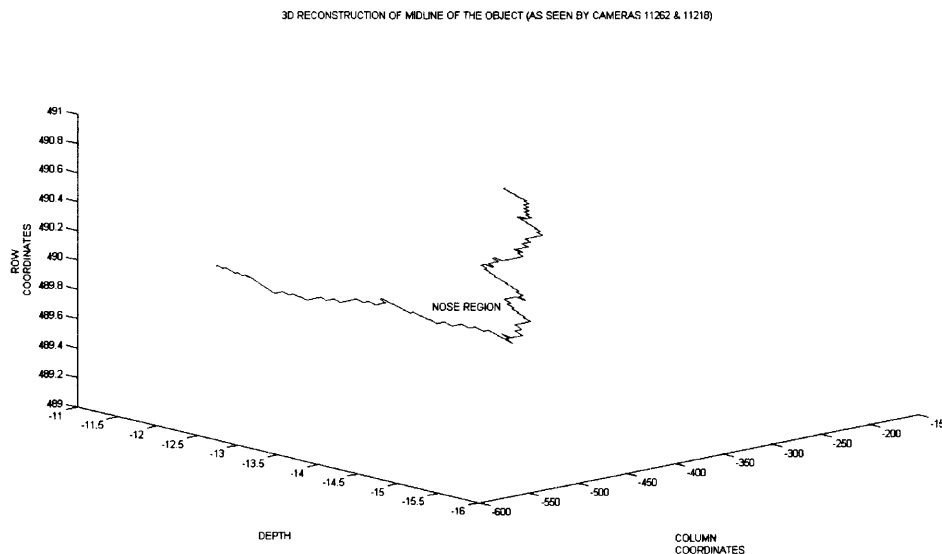


Figure 3.18 3D reconstruction of the midline of the object using data from cameras 2 and 3.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Correspondence Matching

The modified intensity-based correspondence matching algorithm, discussed in section 3.4.4, was tested for its robustness using five different spots on the mask (see Figure 4.1). The spots were illuminated by laser for ease of visual inspection. The test was performed on rectified image pairs of adjacent cameras as well as the pair of two extreme cameras. The robustness test results are presented in Table 4.1. Please note that the coordinates of points in the table are in accordance with MATLAB matrix convention.



Figure 4.1 Five points chosen for testing the modified intensity-based correspondence matching algorithm.

Table 4.1 The results of robustness test performed on the modified intensity-based correspondence matching algorithm

Camera pair	Point of interest	Expected value of y	Detected value of y
1 and 2	(546, 800)	710	710
	(379, 662)	576	574
	(676, 516)	430	431
	(546, 680)	578	578
	(947, 703)	607	601
2 and 3	(546, 675)	775	776
	(364, 537)	646	645
	(670, 382)	490	489
	(540, 534)	623	623
	(968, 549)	637	635
3 and 4	(534, 771)	809	808
	(355, 679)	680	679
	(661, 487)	537	537
	(529, 618)	648	648
	(960, 641)	659	656
4 and 5	(538, 806)	777	780
	(370, 672)	651	653
	(679, 543)	526	526
	(540, 645)	609	607
	(963, 669)	626	624
1 and 5	(549, 753)	763	762
	(376, 601)	641	640
	(681, 450)	505	504
	(550, 619)	600	595
	(964, 650)	591	593

(Note: Order of the cameras is 11248, 11262, 11218, 08719 and 11247 going from left to right)

The first column in Table 4.1 gives the camera pair being tested. The coordinates of the points of interest have been shown in the second column. The last two columns show the expected y coordinate and the detected y coordinate of the conjugate. On comparing the results, it was found that correspondences for the first four spots were within ± 2 pixels of error. The error was slightly higher for the spot located on the beard of the mask due to presence of relatively dark area with less variation in the pixel intensities. It was also observed that the pixel error was higher for cameras located on extreme ends when compared with those located adjacent. Hence, it is concluded that the algorithm is robust enough to be used for further analysis.

4.2 Row-Wise Correspondence Test

The aforementioned algorithm was further used to determine the conjugates of an entire row of points. The row passing through the center of the mask was chosen for this exercise. Due to the nature of the rectification process, corresponding row for every image pair does not coincide. Hence, it is required that the row number be determined for every image pair being used. Table 4.2 gives the coordinates of the end points for every row used for this test. It also gives the location of the end points for the conjugate row; acquired by visual inspection.

A graph was plotted for every pair, showing the trend of variation of the y coordinates, *disparity*, of the conjugate pair. The coordinate used as input has been represented as *y_{in}* whereas the corresponding coordinate has been called *y_{out}*. All the plots have been shown following Table 4.2.

Table 4.2 Coordinates of the first and last pixel for the rows tested in row-wise correspondence analysis.

Camera pair	Test row		Conjugate row (by visual inspection)	
	First pixel	Last pixel	First pixel	Last pixel
1 and 2	(500, 385)	(500, 825)	(500, 315)	(500, 752)
2 and 3	(490, 272)	(490, 720)	(490, 397)	(490, 841)
3 and 4	(482, 392)	(482, 836)	(482, 455)	(482, 896)
4 and 5	(497, 450)	(497, 893)	(497, 448)	(497, 890)
1 and 5	(498, 317)	(498, 778)	(498, 435)	(498, 872)

(Note: Order of the cameras is 11248, 11262, 11218, 08719 and 11247 going from left to right)

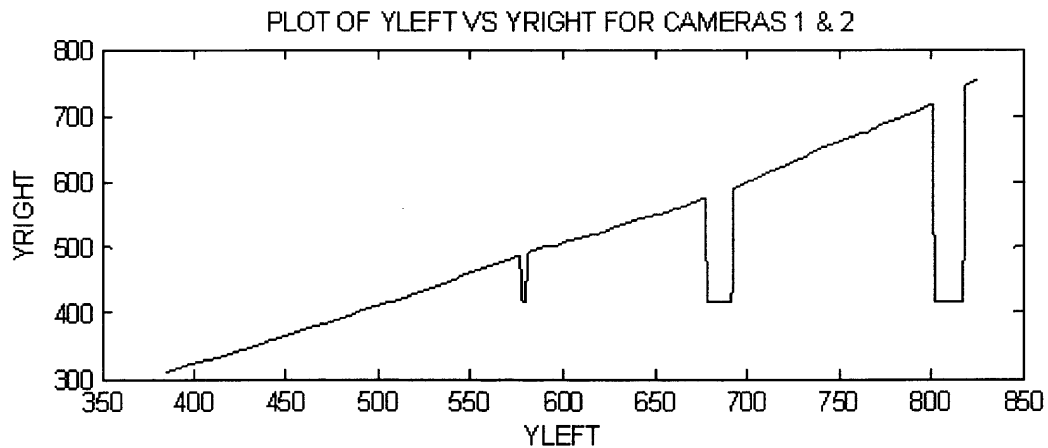


Figure 4.2 Graph showing the trend for variation of y coordinates of the conjugate pair for cameras 1 and 2.

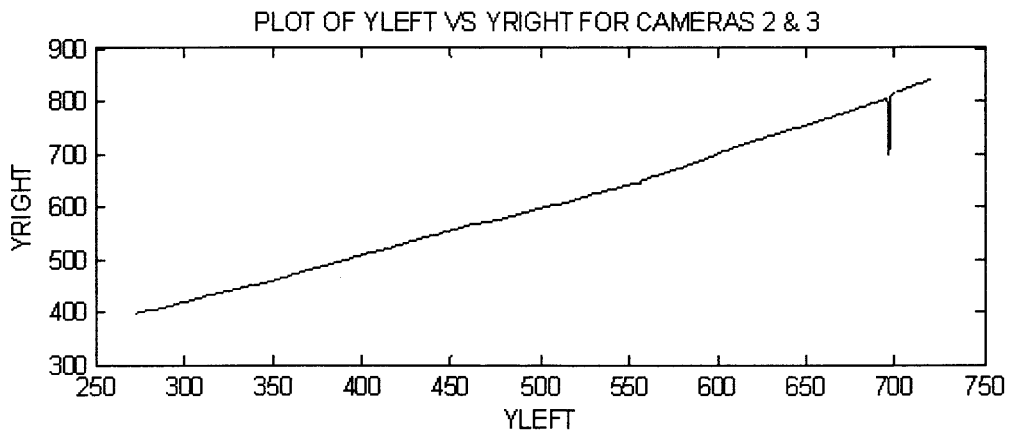


Figure 4.3 Graph showing the trend for variation of y coordinates of the conjugate pair for cameras 2 and 3.

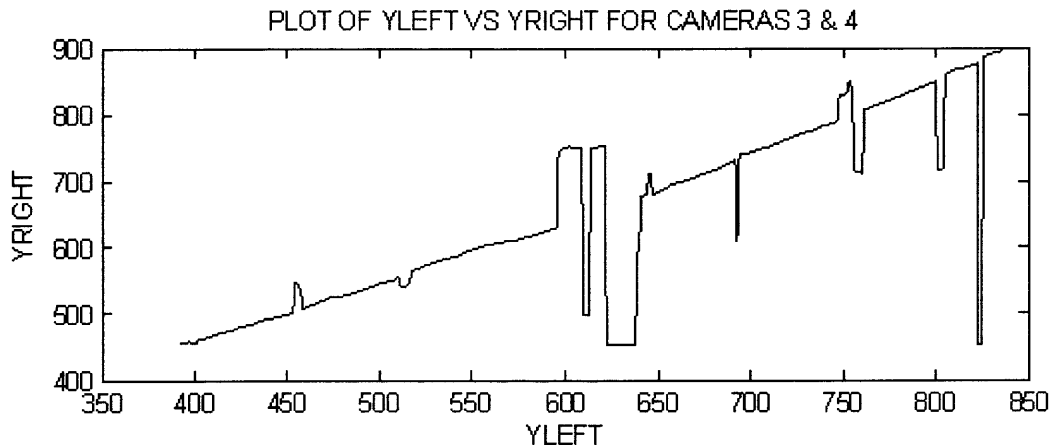


Figure 4.4 Graph showing the trend for variation of y coordinates of the conjugate pair for cameras 3 and 4.

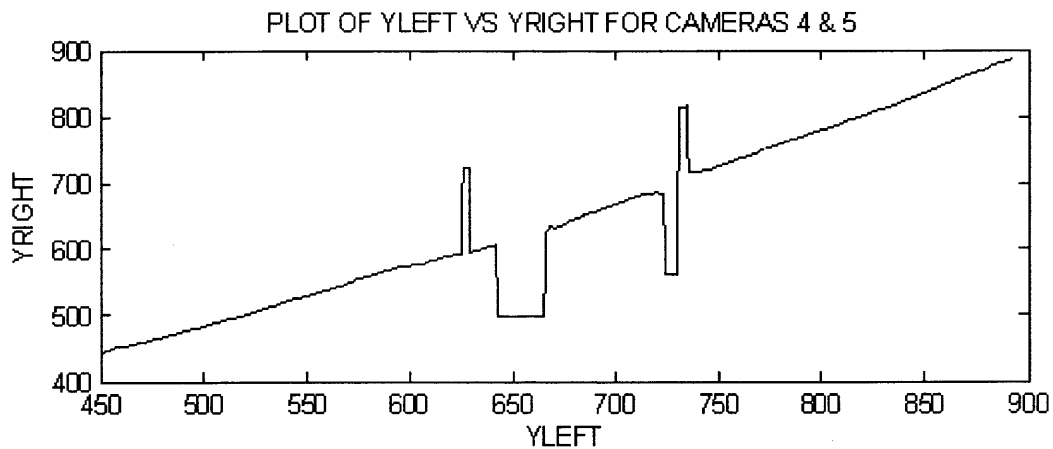


Figure 4.5 Graph showing the trend for variation of y coordinates of the conjugate pair for cameras 4 and 5.

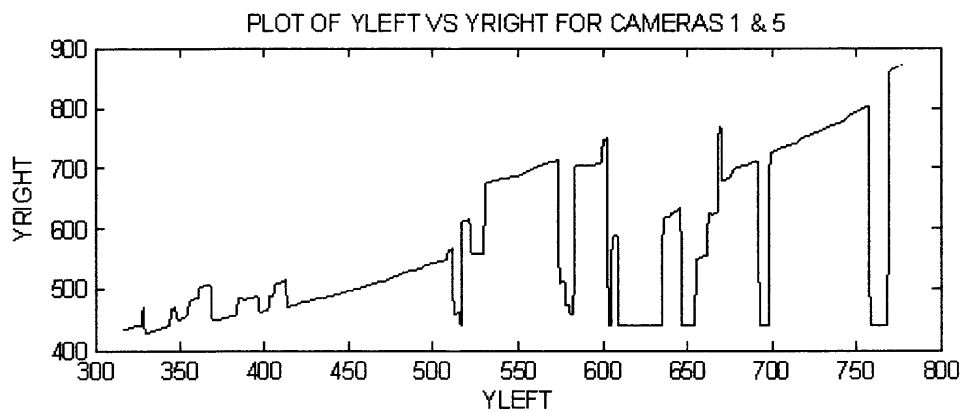


Figure 4.6 Graph showing the trend for variation of y coordinates of the conjugate pair for cameras 1 and 5.

The curves exhibit an almost linear trend. However, if smaller regions are investigated it is found that the textural changes show their presence. Glitches present towards the center of the plot represent portions with consistent gray scale value, in this case the nostril area. The sharp dips towards the end of the curves represent sudden change in the intensity due to rounding of the face. Thus, it can be inferred that these glitches usually coincide with boundaries.

All the plots indicate a similar trend in the variation of y coordinates of the conjugates. The graph is slightly convex with y_{out} rising slowly with y_{in} . The convexity for the last pair is deeper, which can be attributed to the fact that more information is acquired by the cameras at the extreme ends due to large field of view. However, for such a wide baseline, the area of scene overlap is less. As a result of which, the correspondence accuracy is low. This is manifested in the sudden variations seen towards the center of the plot.

As expected, the pairs of adjacent cameras gave a better estimate of the correspondence. This is demonstrated exceptionally well in the graph obtained for cameras 2 and 3 (see Figure 4.3). However, an exception is observed in the graph shown in Figure 4.4, where inconsistencies in the trend are higher. This aberrance may be due to extrinsic factors such as non-uniform illumination in the vicinity of the set-up, reflection from the metallic surface of the table underneath, etc.

It is anticipated that the dips and the peaks in the first four plots will be reduced by employing a suitable smoothing algorithm. The trend acquired in Figure 4.3 can also be used as a reference for predicting the intermediate points. Another proposition is to

triangulation equation is now available. The procedure for depth estimation has already been discussed in Section 3.4.5.

To demonstrate shape reconstruction, the data generated in row-wise correspondence analysis was used. It was seen in Section 4.2, that the correspondence measures obtained for camera pairs (1, 2) and (2, 3) had high precision and accuracy. As such, they were used to estimate the depth of mask at the selected row. The row that was reconstructed has been indicated in Figure 4.7 using a double arrow.



Figure 4.7 The section of the image that was reconstructed using triangulation.

The plot of the estimated depth for camera pair (1, 2) has been displayed in Figure 4.8. The bulge located towards the center of the image coincides with the nose of the mask. The slightly flatter bulges on either sides of the center bulge form the region of the cheek. Similarly for camera (2, 3), the nose and cheeks are distinctly visible in Figure 4.9. However, the depths shown differ in both the plots. This is due to the orientations of the cameras. Recall that camera 1 is located at the extreme left end and is located at an angle much lower than 90° . Camera 3 is located perpendicular to the mask. And it is apparent that camera 2 is located between the aforesaid cameras. Due to the orientation, the projected depth is relatively lower in the first camera pair. The effect of the orientation is also evident in the skewed appearance of the plot. The three dips seen on the plot

utilize all the graphs acquired for adjacent cameras for guessing the corresponding points for end cameras in regions of inconsistency.

As a measure of verification, the expected end points of the rows were compared with the detected end points. The results indicate a close match, thereby, confirming the accuracy of the algorithm. Table 4.3 gives a comparative chart of the expected and the detected end points.

Table 4.3 Comparative chart of the expected and detected end points

Camera pair	Expected		Detected	
	First pixel	Last pixel	First pixel	Last pixel
1 and 2	(500, 315)	(500, 752)	(500, 310)	(500, 753)
2 and 3	(490, 397)	(490, 841)	(490, 397)	(490, 839)
3 and 4	(482, 455)	(482, 896)	(482, 455)	(482, 896)
4 and 5	(497, 448)	(497, 890)	(497, 445)	(497, 887)
1 and 5	(498, 435)	(498, 872)	(498, 427)	(498, 870)

(Note: Order of the cameras is 11248, 11262, 11218, 08719 and 11247 going from left to right)

4.3 Depth Information Extraction

The primary goal of a stereoscopic system is reconstruction of the scene. This requires estimation of the depth for every pixel. As mentioned in Section 2.5, the parameters required for depth estimation are the *base distance*, *focal length* and the *disparity*. The base distance and the focal length were available from the very beginning of the process, However, with row-wise correspondence determination, the third component of the

measures for every tenth row starting from the apex of the nose and ending at the tip were found. The inconsistent regions on the plots acquired were replaced by predicted points using row number 490 as the reference. The three dimensional reconstruction of the nose region is shown in Figure 4.10.

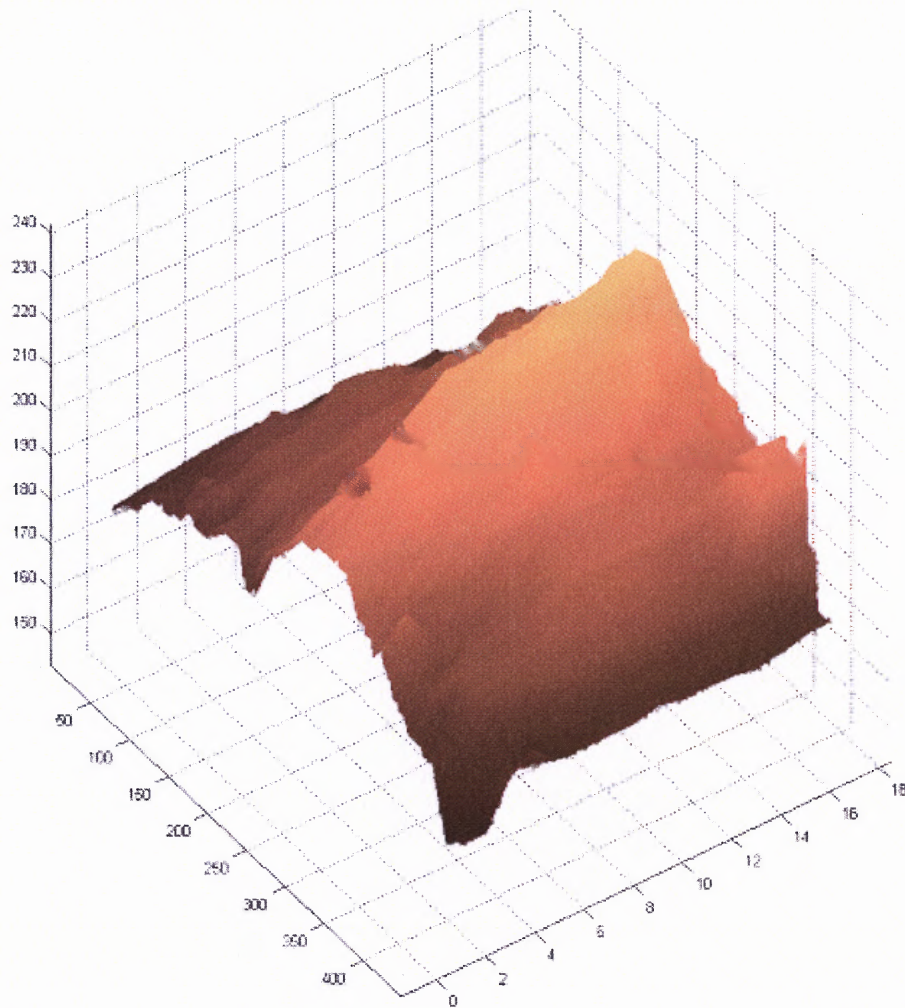


Figure 4.10 3D reconstruction of the nose of the mask using depth estimation algorithm developed.

correspond to the dips seen in Figure 4.2. And hence, they will not be visible if the correspondence measure is subjected to curve smoothing prior to depth estimation.

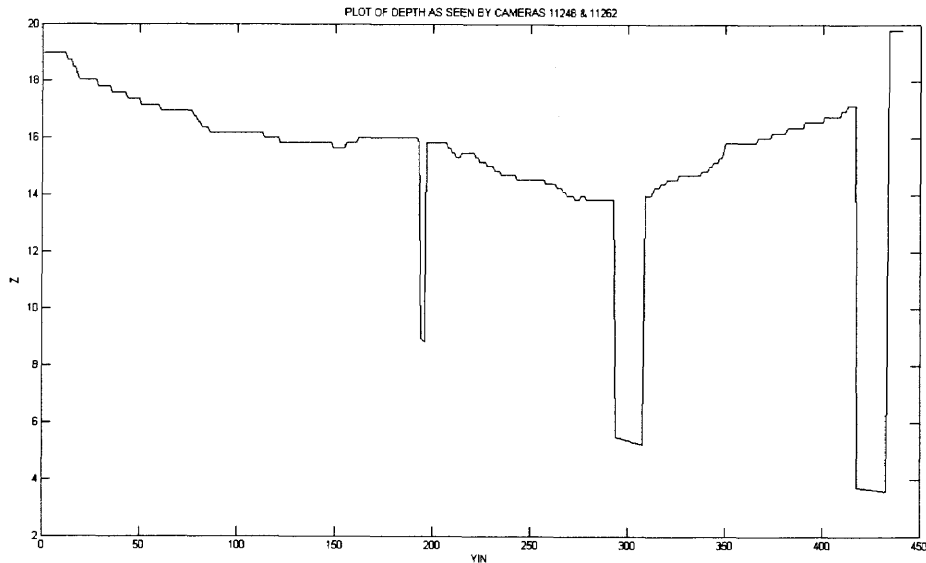


Figure 4.8 Depth estimation using correspondence data from cameras 1 and 2.

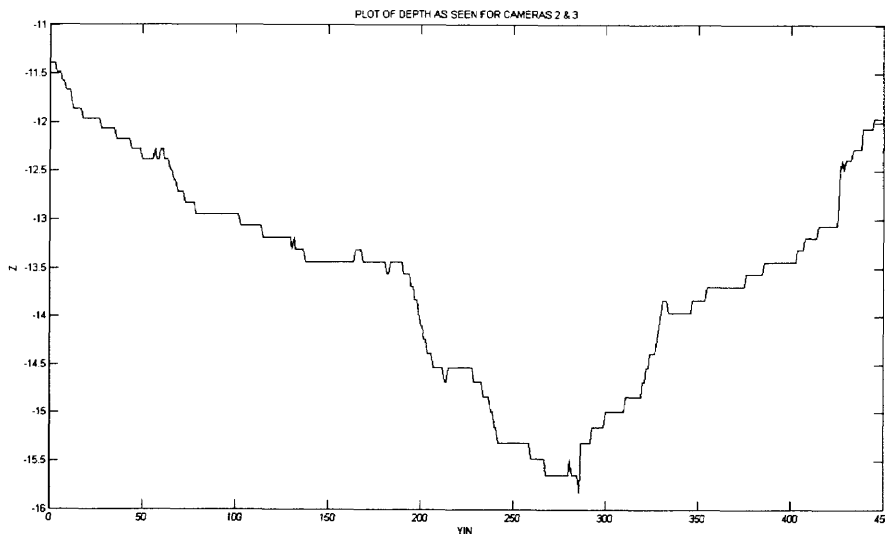
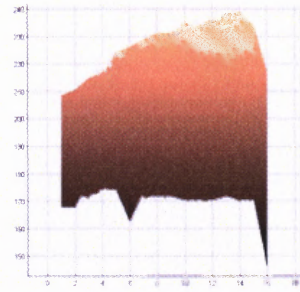
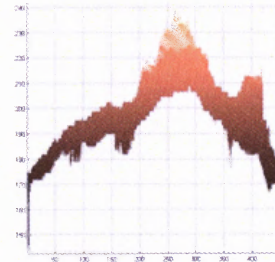


Figure 4.9 Depth estimation using correspondence data from cameras 2 and 3.

The depth estimation was further extended to reconstruction of the mask's nose. The data acquired from cameras 2 and 3 was used for this purpose. Correspondence



(A) Side View



(B) Top View



(C) Front View



(D) Region reconstructed

Figure 4.11 Different views of the reconstructed region along with image of the region.

CHAPTER 5

CONCLUSION AND FUTURE WORK

A good stereoscopic system requires optimal base distance, accurate camera calibration and good correspondence matches for truthful 3D reconstruction. To achieve the desired precision, optimum base distance was calculated and implemented for this thesis. The cameras were calibrated using the existing method. However, it is recommended that another approach for calibration be used for better results. A technique that calibrates the entire array of cameras together would be considered ideal for this case.

The issue of correspondence matching was extensively dealt with during the process. The existing method available for the purpose was modified to increase the robustness of the algorithm. The success of this approach was evident in the improved accuracy, thereby, giving us good correspondence matches. A portion of the object used for the study was reconstructed using the data acquired from an adjacent camera pair. It was found that the reconstruction resembled the actual object. It is anticipated that the reconstruction would be truthful if the depth information provided by the cameras located at the extreme ends is used. However, a good estimate of the conjugates is not acquired for such a pair. Hence, it is proposed that the trends acquired for the adjacent camera pairs be used to reduce the regions of inconsistencies.

Lastly, the system requires complete automation obviating the need of user's presence for the process. This may also eliminate errors generated due to user interference.

APPENDIX A

CALIBRATION RESULTS

All the calibration results for Pixelink A642 camera, individual and stereo, have been presented in the appendix.

```
Command Window
1 To get started, select MATLAB Help or Demos from the Help menu.

Calibration results (with uncertainties):

Focal Length:      fc = [ 2236.20153   2246.69915 ] ± [ 36.08792   38.15334 ]
Principal point:   cc = [ 693.61176   453.53580 ] ± [ 56.86187   48.79746 ]
Skew:              alpha_c = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:        kc = [ -0.32148   0.41688   -0.00377   -0.00437   0.00000 ] ± [ 0.06801   0.64909   0.00319   0.00396   0.00000 ]
Pixel error:       err = [ 1.18107   1.34656 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).

>>
```

Figure A.1 Calibration results for camera 11248.

```
Command Window
1 To get started, select MATLAB Help or Demos from the Help menu.

Calibration results (with uncertainties):

Focal Length:      fc = [ 2382.17679   2387.45101 ] ± [ 52.18236   53.59484 ]
Principal point:   cc = [ 674.72037   544.85457 ] ± [ 61.60163   49.89795 ]
Skew:              alpha_c = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:        kc = [ -0.27397   -2.28262   -0.00139   -0.00076   0.00000 ] ± [ 0.10278   1.35583   0.00480   0.00633   0.00000 ]
Pixel error:       err = [ 1.53364   2.04907 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).

>>
```

Figure A.2 Calibration results for camera 11262.

```
Command Window
1 To get started, select MATLAB Help or Demos from the Help menu.

Calibration results (with uncertainties):

Focal Length:      fc = [ 2248.28763   2249.50518 ] ± [ 31.61490   32.04073 ]
Principal point:   cc = [ 646.26117   493.06617 ] ± [ 29.11866   36.09537 ]
Skew:              alpha_c = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:        kc = [ -0.38889   1.84157   -0.00632   -0.00400   0.00000 ] ± [ 0.06573   0.71128   0.00322   0.00302   0.00000 ]
Pixel error:       err = [ 0.70215   1.46432 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).

>> |
```

Figure A.3 Calibration results for camera 11218.

```

Command Window
To get started, select MATLAB Help or Demos from the Help menu.

Calibration results (with uncertainties):

Focal Length:      fc = [ 2258.00728  2259.50839 ] ± [ 34.59449  35.72004 ]
Principal point:   cc = [ 658.05650  422.12888 ] ± [ 45.00892  45.14787 ]
Skew:             alpha_c = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:       kc = [ -0.32628  0.48145  -0.00688  0.00270  0.00000 ] ± [ 0.06233  0.54719  0.00354  0.00325  0.00000 ]
Pixel error:      err = [ 1.04349  1.46504 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).

>> |

```

Figure A.4 Calibration results for camera 08719.

```

Command Window
To get started, select MATLAB Help or Demos from the Help menu.

Calibration results (with uncertainties):

Focal Length:      fc = [ 2240.52086  2236.48064 ] ± [ 31.20122  32.39539 ]
Principal point:   cc = [ 642.17670  476.14571 ] ± [ 34.51883  37.82257 ]
Skew:             alpha_c = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:       kc = [ -0.45383  2.10604  -0.00203  -0.00193  0.00000 ] ± [ 0.06514  0.81288  0.00272  0.00347  0.00000 ]
Pixel error:      err = [ 0.83331  1.34468 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).

>>

```

Figure A.5 Calibration results for camera 11247.

```

Stereo calibration parameters after optimization:

Intrinsic parameters of left camera:

Focal Length:      fc_left = [ 2225.86952  2231.95342 ] ± [ 36.39753  37.24297 ]
Principal point:   cc_left = [ 626.52005  513.59439 ] ± [ 59.75846  50.74897 ]
Skew:             alpha_c_left = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:       kc_left = [ -0.33860  0.58794  -0.00561  -0.00275  0.00000 ] ± [ 0.11172  1.34988  0.00373  0.00366  0.00000 ]

Intrinsic parameters of right camera:

Focal Length:      fc_right = [ 2326.16132  2339.68451 ] ± [ 36.07866  36.75918 ]
Principal point:   cc_right = [ 637.17834  428.59402 ] ± [ 41.15402  49.72313 ]
Skew:             alpha_c_right = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:       kc_right = [ -0.21194  -1.50752  -0.01255  -0.00370  0.00000 ] ± [ 0.12362  1.54139  0.00395  0.00332  0.00000 ]

Extrinsic parameters (position of right camera wrt left camera):

Rotation vector:   om = [ -0.03438  0.07002  0.00152 ] ± [ 0.02265  0.02508  0.00229 ]
Translation vector: T = [ -89.21689  1.73769  6.74127 ] ± [ 1.40010  1.14134  9.57841 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).

```

Figure A.6 Stereo calibration results for cameras 11248 and 11262.

```

Stereo calibration parameters after optimization:

Intrinsic parameters of left camera:

Focal Length:      fc_left = [ 2293.19486  2281.85074 ] ± [ 45.22392  45.49123 ]
Principal point:   cc_left = [ 682.61997  450.25812 ] ± [ 63.40482  62.93807 ]
Skew:              alpha_c_left = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:        kc_left = [ -0.21241  -0.72796  -0.00759  0.00314  0.00000 ] ± [ 0.12979  1.41426  0.00404  0.00480  0.00000 ]

Intrinsic parameters of right camera:

Focal Length:      fc_right = [ 2282.64091  2277.20358 ] ± [ 42.17813  42.30620 ]
Principal point:   cc_right = [ 531.23332  501.75838 ] ± [ 70.36753  60.21138 ]
Skew:              alpha_c_right = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:        kc_right = [ -0.36796  1.29679  -0.00000  0.00020  0.00000 ] ± [ 0.10889  1.03252  0.00373  0.00557  0.00000 ]

Extrinsic parameters (position of right camera wrt left camera):

Rotation vector:   om = [ 0.04153  0.22342  0.00772 ] ± [ 0.02781  0.03406  0.00408 ]
Translation vector: T = [ -89.77171  2.66149  3.51547 ] ± [ 2.09996  1.30305  11.62733 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).

>> |

```

Figure A.7 Stereo calibration results for cameras 11218 and 11262.

```

Stereo calibration parameters after optimization:

Intrinsic parameters of left camera:

Focal Length:      fc_left = [ 2293.19486  2281.85074 ] ± [ 45.22392  45.49123 ]
Principal point:   cc_left = [ 682.61997  450.25812 ] ± [ 63.40482  62.93807 ]
Skew:              alpha_c_left = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:        kc_left = [ -0.21241  -0.72796  -0.00759  0.00314  0.00000 ] ± [ 0.12979  1.41426  0.00404  0.00480  0.00000 ]

Intrinsic parameters of right camera:

Focal Length:      fc_right = [ 2282.64091  2277.20358 ] ± [ 42.17813  42.30620 ]
Principal point:   cc_right = [ 531.23332  501.75838 ] ± [ 70.36753  60.21138 ]
Skew:              alpha_c_right = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:        kc_right = [ -0.36796  1.29679  -0.00000  0.00020  0.00000 ] ± [ 0.10889  1.03252  0.00373  0.00557  0.00000 ]

Extrinsic parameters (position of right camera wrt left camera):

Rotation vector:   om = [ 0.04153  0.22342  0.00772 ] ± [ 0.02781  0.03406  0.00408 ]
Translation vector: T = [ -89.77171  2.66149  3.51547 ] ± [ 2.09996  1.30305  11.62733 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).

>> |

```

Figure A.8 Stereo calibration results for cameras 11262 and 08719.

```

Stereo calibration parameters after optimization:

Intrinsic parameters of left camera:
Focal Length:      fc_left = [ 2285.20284  2292.18114 ] ± [ 34.94001  35.70293 ]
Principal point:   cc_left = [ 654.50939   359.65834 ] ± [ 33.24589   44.02130 ]
Skew:             alpha_c_left = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:       kc_left = [ -0.40292  1.85365  -0.00711  -0.00071  0.00000 ] ± [ 0.08303  0.74659  0.00425  0.00275  0.00000 ]

Intrinsic parameters of right camera:
Focal Length:      fc_right = [ 2268.55809  2267.27285 ] ± [ 33.70139  34.16904 ]
Principal point:   cc_right = [ 611.49811  583.98086 ] ± [ 51.98778  48.63667 ]
Skew:             alpha_c_right = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:       kc_right = [ -0.41605  1.45669  0.00148  0.00213  0.00000 ] ± [ 0.11848  1.42225  0.00358  0.00318  0.00000 ]

Extrinsic parameters (position of right camera wrt left camera):
Rotation vector:   om = [ 0.09167  0.12596  0.00388 ] ± [ 0.02098  0.02364  0.00256 ]
Translation vector: T = [ -92.53235  -3.55631  23.84397 ] ± [ 1.55614  1.17868  9.41744 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).
>>

```

Figure A.9 Stereo calibration results for cameras 08719 and 11247.

```

Stereo calibration parameters:

Intrinsic parameters of left camera:
Focal Length:      fc_left = [ 2342.76754  2337.57415 ] ± [ 48.56937  49.08763 ]
Principal point:   cc_left = [ 678.23712  513.92506 ] ± [ 75.23539  69.02691 ]
Skew:             alpha_c_left = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:       kc_left = [ -0.48939  1.91228  -0.00101  -0.00721  0.00000 ] ± [ 0.16503  2.04804  0.00487  0.00432  0.00000 ]

Intrinsic parameters of right camera:
Focal Length:      fc_right = [ 2270.54995  2276.86840 ] ± [ 44.48345  44.86809 ]
Principal point:   cc_right = [ 553.85275  374.64423 ] ± [ 75.55653  72.68366 ]
Skew:             alpha_c_right = [ 0.00000 ] ± [ 0.00000 ] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:       kc_right = [ -0.16525  -0.65453  -0.00015  0.00103  0.00000 ] ± [ 0.14664  1.40131  0.00656  0.00532  0.00000 ]

Extrinsic parameters (position of right camera wrt left camera):
Rotation vector:   om = [ -0.04794  0.52459  0.00116 ] ± [ 0.03390  0.04299  0.01352 ]
Translation vector: T = [ -362.99705  6.01253  81.30696 ] ± [ 7.32774  3.16600  19.58333 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).

```

Figure A.10 Stereo calibration results for cameras 11248 and 11247.

APPENDIX B

MATLAB CODES USED FOR THESIS

This section consists of all MATLAB codes developed for this thesis.

```
1) GRAB_IMAGE Grabs image from Pixelink A642 camera

% IMAGE = GRAB_IMAGE (HARDWARE_SERIAL_NO)

% The function requires the user to pass the hardware serial number of
% the desired camera. It opens the camera, set the parameters required
% for acquiring a good picture and snaps the image. The image is
%outputted in grayscale and is in .bmp format

function cam_image = grab_image (HWSNno)

%CONVERTS THE HARDWARE SERIAL NO TO INTERNAL SERIAL NO
switch (HWSNno)
    case {11248}
        SNno = 70216;
    case {11247}
        SNno = 70215;
    case {08719}
        SNno = 34585;
    case {11262}
        SNno = 70242;
    case {11218}
        SNno = 70168;
end
%SETS THE IMAGE PARAMETERS AND GRABS THE IMAGE
h=plopen (SNno);
plset (h,'GrabOutputType','IMAGE');
plset (h,'GrabColorConversion','BAYER_3BY3_COLOR');
plset (h,'SubWindow','PCS2112_NO_DECIMATION', 0, 0, 1024, 1280);
cam_image = plgrab (h);
cam_image = rgb2gray (cam_image);
cd ('C: \Documents and Settings\Chaitali Mulay\My
Documents\stereovision')
switch (HWSNno)
    case {11248}
        imwrite (cam_image, 'camnum0101.bmp');
    case {11262}
        imwrite (cam_image, 'camnum0201.bmp')
    case {11218}
        imwrite (cam_image, 'camnum0301.bmp')
    case {08719}
        imwrite (cam_image, 'camnum0401.bmp')
    case {11247}
        imwrite (cam_image, 'camnum0501.bmp')
end
```

2) **CALIB_IMAGES** To acquire multiple images for camera calibration or for stereovision algorithm

```

clc
clear all
disp ('Initializing Calibration Image grab')
num_cam = input ('Enter the number cameras being used for calibration
(1-5) or enter 0 to exit :');

switch (num_cam)

    case {0}
        disp ('Program terminated');
        pause (1)

    case {1, 2, 3, 4, 5}
        num_img = input ('Enter the number of images to be grabbed from
each camera :');
        disp ('Enter the serial numbers of the cameras being used
(08719, 11218, 11247, 11248, 11262)');
        for count1 = 1:num_cam
            ser_num (count1) = input ('-'); %#ok<AGROW>
        end
        for count2 = 1:num_img
            for count1 = 1:num_cam
                image = grab_image (ser_num (count1));
                cd ('C: \Documents and Settings\Chaitali Mulay\My
Documents\stereovision')
                imwrite (image, strcat ('camnum0', num2str
(count1), '0', num2str (count2), '.bmp'));
            end
            hit_num = input ('Hit Enter to continue');
        end

    otherwise
        disp ('Wrong entry')
end

disp ('Thank for using calib_images. Click on Calibration Toolbox to
calibrate the cameras');

```

```

3) CORR_IMAGE Determines correspondence of a single pixel.

    [XIN, YIN, YOUT, CORR_VAL, ROW_YVAL] = CORR_IMAGE(LTIMGNAME,
RTIMGNAME)

% The function accepts the images matrices corresponding % to
rectified left and right image and allows the user to select the point
whose correspondence is to be located.

%
%
% LTIMGNAME = MATRIX FOR RECTIFIED LEFT IMAGE
% RTIMGNAME = MATRIX FOR RECTIFIED RIGHT IMAGE
%
% The function outputs the coordinates of the test pixel as well as
those
% of the corresponding pixel. The correlation values for every value
of y
% can also be acquired.
%
%
% XIN = X COORDINATE OF TEST PIXEL
% YIN = Y COORDINATE OF TEST PIXEL
% YOUT = Y COORDINATE OF CORRESPONDENCE MATCHED PIXEL
% CORR_VAL = THE CORRELATION VALUES FOR THE ROW OF Y VALUES
% ROW_YVAL = THE Y VALUES FOR THE ROW
%
% (Note: The function assumes that the images are rectified and,
hence,
% xin = xout.

function [xin, yin, yout, corr_block2, yout_row] = corr_image (lt_img,
rt_img)
clear corr_lr;
clear xin;
clear yin;
clear yout;
disp ('SELECT THE MODE OF POINT SELECTION :')
temp = input ('1. MANUAL INPUT 2. SELECT POINT USING MOUSE :');
switch temp
    case 1
        xin = input ('ENTER THE X CO-ORDINATE OF THE POINT :');
        yin = input ('ENTER THE Y CO-ORDINATE OF THE POINT :');
    case 2
        imshow(lt_img)
        [yin,xin] = ginput(1);
        xin = round(xin);
        yin = round(yin);
        close(1)
end
clc
tic

[lenx1,leny1] = size(lt_img); %#ok<NASGU>
[lenx2,leny2] = size(rt_img); %#ok<NASGU>

xout = xin; % SINCE RECTIFIED IMAGES ARE BEING USED IT IS ASSUMED THAT
THE Y CO-ORDINATE IS THE SAME
winx = input('ENTER THE WINDOW SIZE IN X (>) DIRECTION:');

```

```

winy = input('ENTER THE WINDOW SIZE IN Y (^) DIRECTION:');

N = round(0.5*(winx-1));
P = round(0.5*(winy-1));

%FORMING THE REFERENCE INTENSITY BLOCK
for i=-N:N
    for j=-P:P
        ref_block(i+N+1,j+P+1) = lt_img(xin+i,yin+j); %#ok<AGROW>
    end
end
ref_hgram = imhist(ref_block);

%COMPARING BLOCKS ALONG COLUMN YIN
for k=N+1:leny2-N
    for i=-N:N
        for j=-P:P
            comp_block(i+N+1,j+P+1) = rt_img(xout+i,k+j); %#ok<AGROW>
        end
    end
    histeq_block = histeq(comp_block,ref_hgram); %#ok<AGROW,NASGU>
    corr_block1(k) = corr2(ref_block,histeq_block); %#ok<NASGU,AGROW>
    corr_block2(k) = corr2(ref_block,comp_block); %#ok<AGROW,NASGU>
end
max_corr1 = max(corr_block1); %#ok<NASGU>
max_corr2 = max(corr_block1);

yout_row=1:leny2-N;

[xout,yout] = find(corr_block1 == max_corr1);

```


4) **CORR_CODE** This code is used to read the left and the right images, invoke the function `corr_image` for correspondence determination and display the results.

```

clc
clear all
temp = input('WARNING: ALL WINDOWS WILL BE CLOSED ... PLEASE SAVE ALL
YOUR WORK AND HIT 1 TO CONTINUE:');
switch temp
    case {1}
        close all
    otherwise
        disp('ERROR ... PLEASE CHOOSE AGAIN !!');
        corr_code;
end
str1=input('LEFT IMAGE NAME (PLS INCLUDE THE FILE ETXN):');
str2=input('RIGHT IMAGE NAME (PLS INCLUDE THE FILE EXTN):');
str3=input('ENTER THE PATH OF DIRECTORY:');
cd(str3)

left_image = imread(str1);
right_image = imread(str2);

clear str1;
clear str2;
clear str3;
clear temp;

[xin,yin,yout,corr_lr,yout_row] = corr_image(left_image,right_image);
str4 = strcat('The pixel match for (',num2str(xin),',',num2str(yin),')
was found at (', num2str(xin),',',num2str(yout),')');
disp(str4)

figure(1)
subplot(1,2,1);imshow(left_image),title('LEFT IMAGE')
subplot(1,2,2);imshow(right_image), title('RIGHT IMAGE')

str5 = strcat('PLOT OF CORRELATION VALUES FOUND ACROSS ROW ',
num2str(xin));
figure(2)
title(str5)
ylabel('CORRELATION VALUES FOR EVERY Y ALONG THE ROW');
xlabel('Y VALUES')
plot(yout_row,corr_lr)

```

```

5) %PLOT_ROW: DETERMINES ROW-WISE CORRESPONDENCE FOR A ROW
function [yout_row, corr_block1, corr_block2] = plot_row(lt_img,
rt_img,yin_row)

[lenx1,leny1] = size(lt_img); %#ok<NASGU>
[lenx2,leny2] = size(rt_img); %#ok<NASGU>

winx = 21;
winy = 21;

xin = input('PLEASE ENTER THE ROW TO BE PLOTTED: ');
N = round(0.5*(winx-1));
P = round(0.5*(winy-1));
xout = xin;
tic

for yin = yin_row(1,1):yin_row(1,length(yin_row))
    %FORMING THE REFERENCE INTENSITY BLOCK
    for i=-N:N
        for j=-P:P
            ref_block(i+N+1,j+P+1) = lt_img(xin+i,yin+j);    %#ok<AGROW>
        end
    end
    ref_hgram = imhist(ref_block);
    %COMPARING BLOCKS ALONG COLUMN YIN
    for k=yin-125:yin+125
        for i=-N:N
            for j=-P:P
                comp_block(i+N+1,j+P+1) = rt_img(xout+i,k+j);
            end
        end
        histeq_block = histeq(comp_block,ref_hgram); %#ok<AGROW,NASGU>
        corr_block1(k) = corr2(ref_block,histeq_block); %#ok<AGROW>

%THIS CORRELATION MEASURE USES HISTOGRAM EQUALIZED BLOCK. USE THIS FOR
DECISION MAKING.
        corr_block2(k) = corr2(ref_block,comp_block); %#ok<AGROW,NASGU>
    end
    max_corr1 = max(corr_block1);
    [temp,yout] = find(corr_block1 == max_corr1);
    yout_row(yin-yin_row(1,1)+1) = yout; %#ok<AGROW>
end

temp = input('DO YOU WANT TO SAVE THE FILE?(1.YES,2.NO):');
switch temp
    case 1
        cd('C:\Program Files\MATLAB\R2007a\work')
        str = input('SPECIFY FILE NAME WITH EXTN(E.G: FILENAME.MAT)');
        save str yout_row
    case 2
        disp('THE FILE HAS NOT BEEN SAVED');
end

```

```
6) function [x,y,z] = depth_est(xrow, yleft, yright)
% DEPTH_EST Estimates the depth for a given image row
% [X,Y,Z] = DEPTH_EST(XROW, YLEFT, YRIGHT)
% XROW is the row number, YLEFT is a vector containing the y
coordinates of
% the given row (xrow) and YRIGHT is a vector containing the conjugates
% found using PLOT_ROW.
% (X,Y,Z) are the world coordinates for every point in the row input

b = 89/0.0067;      % BASE DISTANCE BETWEEN ADJACENT CAMERAS ( APPROX 89
MM)
f = 16/0.0067;     % FOCAL LENGTH OF PIXELINK A642 CAMERA (16 MM)

leny = length(yleft);
x = xrow*ones(1,leny);

disp = yleft - yright;
y = (b*yleft)./disp;
y = abs(y*0.0067);

z = (b*f)./disp;
z = abs(z*0.00067); % CONVERTING PIXELS TO CENTIMETERS, 0.0067
                    % MICROMETERS IS THE PIXEL PITCH
```

REFERENCES

- [1] CVOnline – Compendium of Computer Vision, School of Informatics, University of Edinburgh, “Applications”, November 2007,
<http://homepages.inf.ed.ac.uk/rbf/CVonline/applic.htm>

- [2] L.Vaina, “‘What’ and ‘Where’ in the Human Visual System: Two Hierarchies of Visual Modules.” in *Synthese* **83**, 1990, pp 49-91.

- [3] R. Klette, K Schlüns and A. Koschan, *Computer Vision, Three-Dimensional Data from Images*. Singapore: Springer, 1998

- [4] O. Faugeras, *Three-Dimensional Computer Vision*, Cambridge: MIT Press, 1996.

- [5] J. Heikkilä and O. Silvén, “A Four-step Camera Calibration Procedure with Implicit Image Correction.” in *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition*, 1997, pp 1106-1112.

- [6] A. Fusiello, E. Trucco and A. Verri, “A compact algorithm for rectification of stereo pairs.”, in *Machine Vision and Applications*, 2000, pp 16-22.

- [7] Intel Corp., “Camera calibration Toolbox for MATLAB”, November 2007.
http://www.vision.caltech.edu/bouguetj/calib_doc/htmls/parameters.html

- [8] P. Witoonchart, and R. Foulds, “Three-Dimensional Surface and Volume Measurements Using a Camera Array.” in *IEEE 28th Annual Northeast Bioengineering Conference* (2002): pp 145-146.

- [9] N. Shriram, “Implementation of Stereo Correspondence Algorithms For Multi – Baseline Vision System.”, M.S. Thesis, New Jersey Institute of Technology, Newark, NJ, US, 2007.