

Teat Detection for an Automated Milking System

by

Aidan Hunt Duffy (B.Eng)

This thesis is being presented in fulfillment of the requirements for the
qualification of:

Masters of Engineering

Dublin City University

Supervisor: Dr. Harry Esmonde (PhD)

2nd Supervisor: Dr. Brian Corcoran (PhD)

School of Mechanical and Manufacturing Engineering

Teagasc Advisor: Dr. Eddie O'Callaghan (PhD)

Dairy Production Research Centre, Teagasc, Fermoy, Co. Cork

July 2006

Volume 1 of 1

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of a Masters in Engineering is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed: *Aiden Hurley* (Candidate)

ID No.: 99398664

Date: 17th September 2006

Abstract	1
Chapter 1 - Introduction	2
1.1 – Automation of a Rotary Carousel Milking Parlour	2
1.1.1 – Necessity of Automation in Irish Dairy Farming.....	2
1.1.2 - Existing AMS.....	3
1.1.3 - Existing AMS unsuitable for Irish Dairy Farming	4
1.1.4 – Benefits of Automating a Rotary Based Milking Parlour	5
1.2 – Teat Identification in Autonomous Teat Cup Attachment	6
1.2.1 – Autonomous Cup Attachment.....	7
1.2.2 – Teat Identification and Location system	10
Chapter 2 – Theory.....	17
2.1 - Three Dimensional Computer Vision	17
2.1.1 – Single View Geometry	18
2.1.2 - Two View Geometry.....	24
2.1.3 - Camera Calibration	30
2.2 – Detection of Lines in Images	35
2.2.1 – Canny Edge Detector.....	35
2.2.2 – Hough Line Detector	42
Chapter 3 – IceRobotics System	46
3.1 - System Description	46
3.1.1 - Stereo Camera Rig	47
3.1.2 - IceRobotics Server.....	49
3.2 - System Assessment	62
3.2.1 - Identification of Teats	62
3.2.2 - Teat Identification in Practise	64
3.2.3 - Accuracy of System in Locating Teats	82

Chapter 4 - Teat Angulations	100
4.1 - Outline.....	100
4.2 - Algorithm.....	100
4.2.1 - Convert 3D teat into 2D image coordinates in the stereo pair of images	102
4.2.2 - Convert the stereo pair of images into greyscale	108
4.2.3 - Crop images to Region of Interest	109
4.2.4 - Isolate the edges in the cropped images.....	111
4.2.5 - Find Straight lines in Edge Image.....	112
4.2.6 - Selecting Lines corresponding to sides of Teat	114
4.2.7 - Translate Side-lines.....	123
4.2.8 - Centreline of Teat	124
4.2.9 - Arbitrary Points on Centreline	127
4.2.10 - Corresponding Points in Right Image.....	130
4.2.11 - Triangulating 3D Vector	134
4.2.12 - Transform to World Reference Frame.....	135
4.2.13 - World to Teat Rig Reference Frame	136
4.2.14 - Visualisation and Angle to Plane.....	137
 Chapter 5 – Angle Accuracy Test Results.....	 142
5.1 - Test setup	142
5.2 - Reference Plane Transformations	144
5.3 - Teat Angle Test Results	145
 Chapter 6 – Discussion.....	 157
6.1 - IceRobotics Teat Tracker Review.....	157
6.1.1 – Robustness of Teat Identification Process.....	157
6.1.2 – Accuracy of Teat Location Estimates	159
6.2 - Angle Extraction Algorithm Review	161
6.2.1 - Accuracy of Angle Measurements.....	161
6.2.2 - Performance in Detection of the Teat Sides.....	162
6.2.3 - Further Development of Algorithm	169

Chapter 7 – Conclusions	171
7.1 - Teat Identification and Location	171
7.2 – Angulations of Teats	172
7.3 – Future Work and Recommendations	172

Appendix A – Additional Mathematics

Appendix B – Apparatus Setup

Appendix C – Software Code

References

Abstract

Application time when placing all four cups to the udder of a cow is the primary time constraint in high capacity group milking. A human labourer can manually apply four cups per animal as it passes on a rotary carousel in less than ten seconds. Existing automated milking machines typically have an average attachment time in excess of one minute. These systems apply the cups to each udder quadrant individually. To speed up the process it is proposed to attach all four cups simultaneously. To achieve this, the 3D position and orientation of each teat must be known in approximate real time. This thesis documents the analysis of a stereo-vision system for teat location and presents further developments of the system for detection of teat orientation. Test results demonstrate the suitability of stereovision for teat location but indicate that further refinement of the system is required to produce increased accuracy and precision. The additional functionality developed for the system to determine teat orientation has also been tested. Results show that while accurate determination of teat orientation is possible issues still exist with reliability and robustness.

Chapter 1 - Introduction

1.1 – Automation of a Rotary Carousel Milking Parlour

This project is a component of a larger project with the goal of automating the milking process in an automated rotary milking parlour for the benefit of Irish dairy farming. A brief account of Irish dairy farming is presented to highlight the need for automation and the unsuitability of automatic milking systems (AMS) currently available.

1.1.1 – Necessity of Automation in Irish Dairy Farming

Reductions in milk price have created the need for dairy farmers to increase levels of milk production in order to preserve farm income. Increased production requires a larger herd size; more cows means more labour. Studies have shown that the milking process is already responsible for over one third of all dairy labour input [1]. The milking process takes up a large proportion of the labour input and is therefore an area that should be focussed on if significant labour-cost savings are to be made. In recent times smaller farms have ceased production of milk and the quotas have been taken up by the bigger dairy farms. This creates a demand for additional labour. There is a constant need to expand production while maintaining a minimal labour force in order to remain competitive. This means that labour input cannot increase in line with milk production. Experts agree that the most obvious way to fill the void between the workload and the available labour on a dairy farm is to automate the milking process [2]. As well as the obvious cost benefits associated with reducing required labour there are other less quantifiable (but considerably important) factors. In western countries such as Ireland it is becoming progressively more difficult for farmers to find willing labourers. The booming construction industry offers wages with which dairy farms cannot possibly compete and thus farmers have a hard time hiring reliable labourers prepared to work long and antisocial hours that dairy farming demands. Automating the milking process creates more free time. While the milking is being carried out other jobs can be tackled. The time created by an AMS gives the farmer the option of getting more work done in other areas and therefore increasing profits, but it also gives the option of more free time for non-

farming activities without reducing profits. An AMS presents the opportunity for a different lifestyle for the farmer. Automatic milking creates the opportunity for systematic cleaning of the animals. It is necessary to adequately clean the udder of the animals to prevent contamination of the milk [3]

1.1.2 - Existing AMS

In 2004 approximately 400 European farms were milking using an AMS [4] and this figure has increased since. They are increasingly popular in North East Europe where the cows are permanently housed in a shed based system. The Voluntary Milking System (VMS) by DeLaval [5] (Fig 1.1) is the current market leader with sales of over one thousand units.



Fig 1.1 – The DeLaval Voluntary Milking System [6]

An in depth study of the DeLaval VMS and of the other existing, less popular, AMS has been conducted and documented [7]. The other systems include the PUNCH Technix/Robotic Milking Solutions Titan AMS (formally Gascoigne Melotte Zenith) [8], the Lely Astronaut [9], the SAC (formally Galaxy) [10] and the Fullwood Merlin [11]. All of the existing systems are designed for milking a permanently housed herd of typically 60 cows. The milking systems operate 24 hours a day, allowing the cows to be milked on demand. The cows themselves choose when they want to be milked and make

their way to the AMS. This process is known as 'voluntary milking'. The cows are enticed to the AMS by providing concentrate feed in the milking stall. They are usually milked 3 times per day for optimum yields. If a cow returns to the AMS after too short a period of time since its previous visit it will be refused. The incorporation of an AMS to a dairy farm is costly, the system in itself costing in the region of €150,000 per two stall unit. Further costs are incurred through barn modifications, the culling of cows incompatible with the AMS, and the regular maintenance of the system. The introduction of an AMS to a farm generally results in a reduced production of milk (due to the restricted herd sizes), and because of this the decision to move to automated-milking is generally for improved lifestyle purposes rather than for financial benefits.

1.1.3 - Existing AMS unsuitable for Irish Dairy Farming

Nearly all the dairy farming in Ireland is pasture based. The existing AMS are unsuitable for use with grazing because difficulties arise in achieving frequent milking. The readily available supply of food (from grazing) means that the lure of the concentrate in the milking stall is not so strong. In order to achieve visits regularly enough to produce profitable milk yields it is necessary to install more than one AMS unit. More units spread out over the grazing area means the cows are not required to walk as far to be milked, making the process more appealing. However, this significantly increases the capital outlay involved for a farm moving to automatic milking. As well as the cost of the additional units, factors such as extra milk lines, power lines and cooling must be considered. Maintenance and monitoring of cows becomes more difficult due to the distance between the units and is more time consuming for the farmer. Also, the dependence on concentrate food offsets the savings incurred with grazing. Another problem with introducing more AMS units is that the likelihood of idle time (periods when the AMS is not in use) is increased. Allowing the cows the freedom of a natural grazing habitat will inevitably cause the cows to settle into eating and milking habits influenced by the environment. This will cause peak milking periods when the AMS units will be in more demand. Similarly there will be periods when the machine is not in use at all. The huge cost of installing an AMS is often justified by the fact that it is in constant

use 24 hours a day, 365 days a year, but with grazing this is not the case. The machines are not utilised to their maximum capacity and are a source of inefficiency on the dairy farm.

1.1.4 – Benefits of Automating a Rotary Based Milking Parlour

A modern rotary carousel, such as the one pictured in Fig 1.2, is suitable for milking large herd sizes. With a 60 stall rotary it is possible to milk up to 300 cows in an hour [2]. Together, all the cows are herded to the rotary when it is time for them to be milked. Tasks such as cluster removal, teat spraying, milk monitoring, milk diversion, cow identification and health monitoring have already been successfully automated. Manually attaching the milking cups to each animal is the most time consuming element of the process for the farmer during milking.



Fig 1.2 – A Rotary Carousel [12]

The rotary topology is suited to the conditions of Irish dairy milking. Automating the attachment of the milking cups would provide a time efficient milking solution. Employing such a system in Irish dairy farming will have numerous advantages over the traditional automated systems currently available, and over a non-automated rotary parlour:

- **Greater Capacity:** The throughput of the system is not restricted by the milking time of the animal as with conventional systems. Bigger capacity can be achieved by

increasing the number of stalls in the carousel and by reducing the teat cup attachment time

- **More Cost Effective:** A single robot can be used to service more cows. The utilisation of assets is better than with the conventional automated milking systems.
- **Reduced Labour:** The farmer is only required to herd the animals to the AMS Rotary. Presence is only required when there is a problem. This frees up a significant quantity of time for a farmer who would usually have to be present for the entire milking process using a conventional rotary system (manual attachment of teat cups).
- **No Down Time:** In the event of the AMS malfunctioning and requiring maintenance the herd will not go un-milked. Should the system be non-operational the rotary system allows the farmer to milk the herd by manually attaching the teat cups to the cows.
- **Conventional rotary parlours** are already established in the Irish dairy industry. These have proven successful and more farmers are willing to venture with a recognisable system.
- The approach to dairy farming does not change as drastically as it does when a standard AMS is introduced. The food sources that the farmers are currently using can continue to be used. Grazing can continue.

1.2 – Teat Identification in Autonomous Teat Cup Attachment

The goal of this project is to develop a system to provide the information necessary to autonomously attach the four milking cups to the teats of a cow. This section outlines the attachment process and defines the requirements of the teat identification system. Some of the methods of identification used in existing automated milking systems are briefly described. The reasons for their unsuitability for this project are given. The decision to use a Stereo-Vision system is explained, as is the acquirement of a purpose built vision system and the need to augment its functionality.

1.2.1 – Autonomous Cup Attachment

The teat cup application time constrains the utilisation of a rotary carousel. A typical operator can manually apply the cluster of four teat cups to a cow as it passes in under ten seconds. A traditional AMS can take up to two minutes to complete the teat cup application stage and usually takes around a minute. There is no onus on these systems to quickly apply the cups since there isn't a dramatic effect on throughput (The attachment time is a small fraction of the milking time, for which the AMS is occupied, and, the milking is spread out over a 24 hour period). The automation of teat cup application is broken into two stages: 1) Identifying and locating the teats and 2) Applying the cups to the teats.

1.2.1.a - Teat Cup Application

In all of the existing automated systems each of the four milking cups are individually attached to the teats. The teat identification system provides the location of a single teat for attachment purposes. This process is completed four times per cow and incurs a large time overhead. Fig 1.3 contains a prototype design for a manipulator arm end-effector intended to simultaneously attach all four milking cups to an animal.

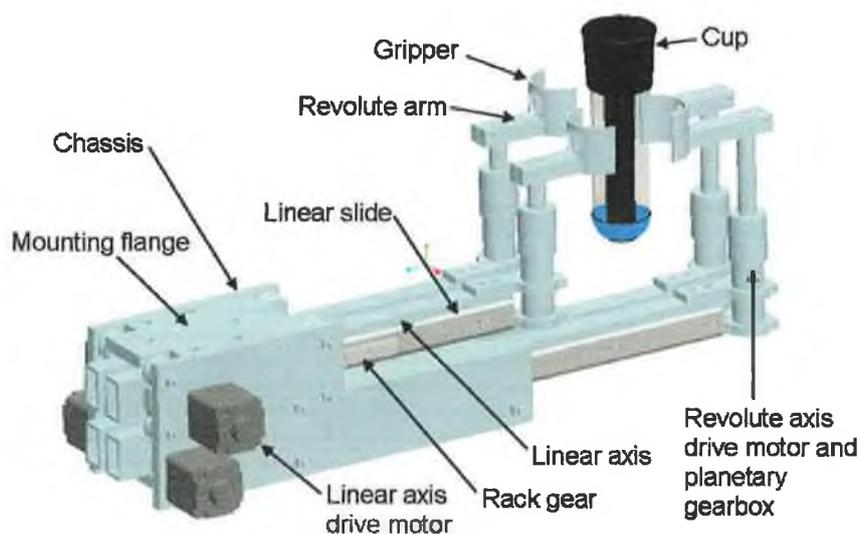


Fig 1.3 – End effector to simultaneously attach four milking cups to a cow [13]

The design seen in Fig 3.1 has four linear actuators. Attached to each actuator by a rotational actuator is a suction plate for holding a milking cup. The design allows the end effector to manoeuvre the cups independently in a horizontal plane beneath the udder of a cow. Once all the cups are positioned beneath the teats the four cups are moved vertically upwards to complete the attachment. The upward actuation is provided by the manipulator arm to which the end effector is attached. There is no independent actuation of the four teat cups along the vertical axis. The angle at which a teat cup is held is fixed. The central axes of all four cups remain parallel to the vertical axis whilst under the cow during the attachment process. Independent movement of the cups in the horizontal plane allows the individual adjustment of the cups during the vertical actuation of the end effector for ease of attachment to a teat orientated at a large angle to the vertical. A cup can be adjusted horizontally so that centre of the cup opening traverses the central axis of the teat as the cup moves up.

1.2.1.b – Identification and Location of Teats

To enable the four teat cups to be attached simultaneously the teat identification and location system must be able to provide the 3D position and orientation of each teat instantaneously in approximate real time. It is a design requirement that the 3D position of teats be known to within an accuracy of $\pm 5\text{mm}$ and their orientation angles known to within $\pm 5^\circ$.

1.2.1.c – Standard Udder Dimensions

The data presented in Table 1.1 was collected from a herd of 54 Friesian cows from Holland [14]. This data defines the working volume in which teats can be expected and the expected spacing and sizes (width and length) of the teats.

Trait [cm]	\bar{x}	sd	Minimum	Maximum
Horizontal length	44.1	3.9	38.0	54.0
Length of attachment	47.8	3.8	40.0	57.0
Foreudder width	29.3	3.0	23.0	35.0
Rear udder width	21.0	1.7	18.0	25.0
Depth of foreudder	24.8	1.7	22.0	28.0
Rear teats distance to the floor	59.2	2.9	52.0	66.0
Front teats distance to the floor	58.8	2.7	53.0	66.0
Length of front teats	4.9	0.6	3.5	6.0
Length of rear teats	4.0	0.5	3.0	5.0
Diameter of front teats	2.7	0.2	2.2	3.2
Diameter of rear teats	2.7	0.2	2.2	3.2
Distance between front teats	13.0	2.3	8.0	18.0
Distance between rear teats	5.2	1.5	3.5	10.0
Distance between front and rear teats	12.5	1.9	7.5	18.0

Table 1.1 – Average dimensions of Friesian Cows from Holland

The diagram in Fig 1.4 has been derived from the data in Table 1.1. The region in the horizontal plane in which the teats can be expected is illustrated. The grey region corresponds to the upper dimension limits, the orange region corresponds to the lower dimensions limits and the green region represents the mean dimensions given in Table 1.1.

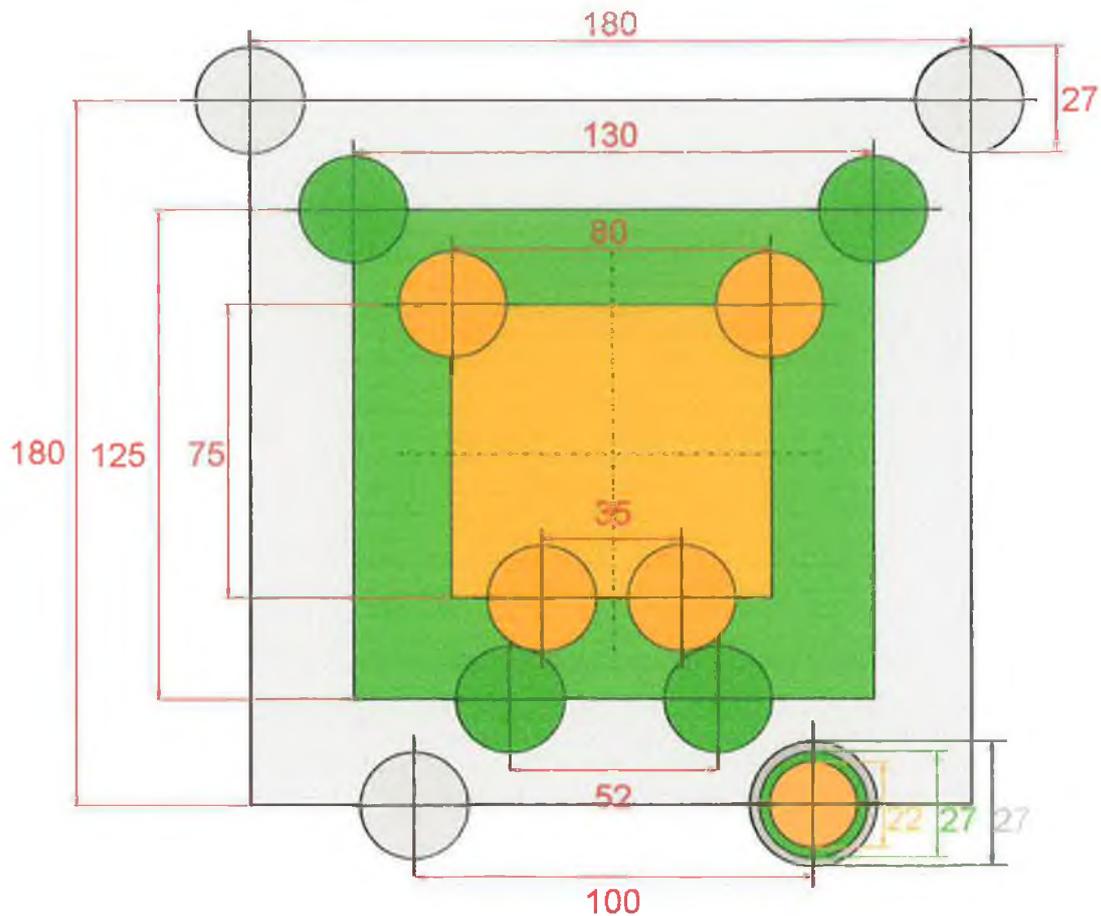


Fig 1.4 – Minimum, Maximum and Mean udder dimension from a sample of 54 cows

1.2.2 – Teat Identification and Location system

1.2.2.a - Technologies used in existing systems

Two internal reports [7] [15] document in detail the technologies that are employed in the AMS that are currently available. These reports rely upon firsthand observation of the systems in operation, the specifications provided by manufacturers and from the review of patents pertaining to the systems. The primary sensing solution employed in existing AMS is a laser based vision system. A single camera is used to detect the location of a laser stripe that is incident on a teat. The location of the stripe in the camera image, coupled with the relative distances between the laser and the camera and the angles of incidence is used to triangulate the position of the teat.

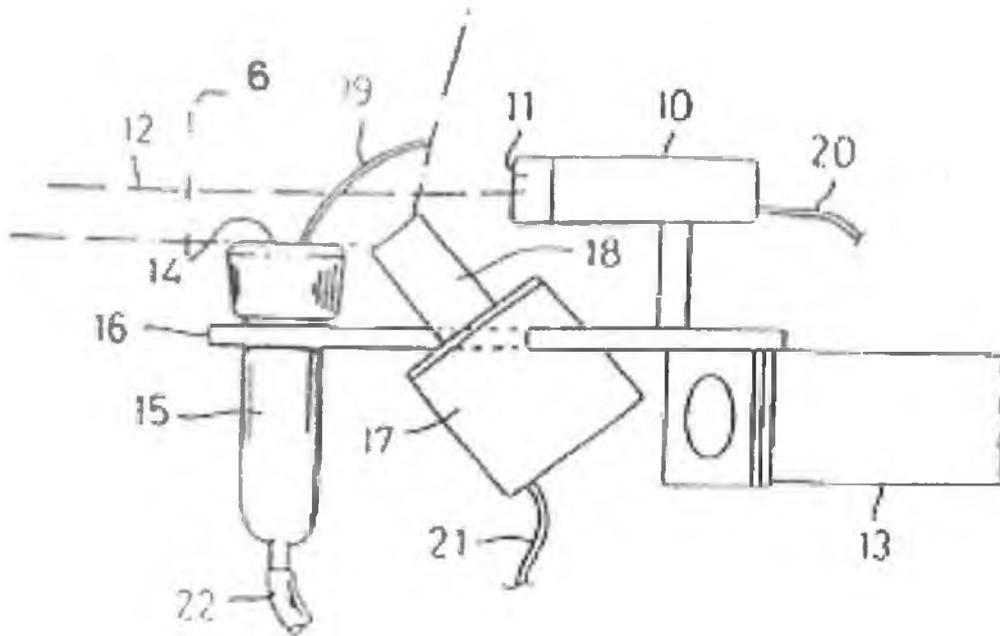


Fig 1.5 – Image from patent WO98/47348: Single Camera and Laser stripe generator

The drawings in Fig 1.5 and Fig 1.6 are included in the international patent WO90/47348 [16]. In Fig 1.5 there is a camera (no. 17) and a laser horizontal stripe generator (no. 10) attached to the end-effector of a manipulator arm (no. 13). The end-effector has the means of gripping the milking cup (no. 15) by means of a carrier (no. 16). The end effector is manoeuvred beneath the cow in order to scan the laser across the udder. When the horizontal laser stripe is incident on a teat a line segment will be seen in the image of the camera. Scanning the end-effector in a vertical plane will result in a series of line segments in the image corresponding to the teat as in Fig 1.6. The ends of the segments define the outer edges of the teats. Once the teat is identified its position is determined using laser triangulation techniques.

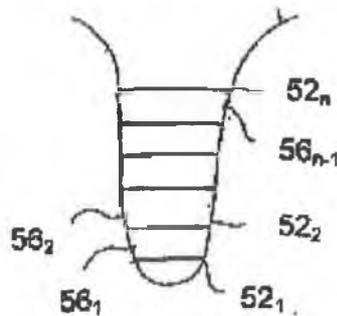


Fig 1.7 - Image from patent WO98/47348: Series of illumination stripes on a single teat

Such a system can provide reliable and accurate information regarding the position of a teat but it can only detect a single teat at a time. Once a teat has been successfully located and the milking cup is applied the system moves on to the next teat. This procedure is repeated until all four milking cups are attached. This method is not a suitable means of teat detection as the positions of all four teats are not known simultaneously. Also, there is a time overhead associated with the scanning motion.

1.2.2.b - Stereo Vision

Stereo vision employs the same techniques used by humans to visually determine the depth. Two cameras are used to acquire a pair of images of the same scene from slightly different viewpoints. An object in the scene will appear in two different locations in the images produced by the two cameras. This difference, known as the disparity, is used to determine how far away an object is. The closer the object is to the cameras the greater the disparity will be. If an object can be seen in the image of both cameras then its 3D position can be triangulated. In order to recover accurate measurements of the scene captured in the image camera calibration techniques must be employed to determine parameter values which quantify the inherent lens distortion present in the image. Early developments in camera calibration came in the field of stereoscopic mapping [17]. The dramatic increase in aerial photography for reconnaissance and mapping during the Second World War brought international recognition that standardisation of techniques for camera calibration were necessary, but it was not until 1966 that the modern camera model form first appeared in published work [18]. Brown demonstrated that distortions

are entirely attributable to decentering of the lens. Advances in digital imaging (CCD cameras and computer processing) paved the way for low cost 3D machine vision. Tsai's paper (1987) [19] on the calibration of 'off-the-shelf' TV cameras and lens, accurately determining both the internal and external parameters of the cameras, was at the epicentre of this 3D imaging revolution. Iterative non linear optimisation techniques significantly improved the distortion parameter estimates [20]. It was shown that 3D reconstruction of scenes can be achieved using an un-calibrated stereo rig and the Fundamental matrix and Epipolar geometry can be established from a single stereo pair [21]. This relies only on the relative positions and orientations of the cameras and does not require the internal parameters to be initially determined, greatly simplifying the task of calibration for applications such as image rectification [22] and stereo matching [23]. Reduced processing overhead of optimises stereo algorithms coupled with increased processing power had enabled real time 3D reconstruction of a scene using stereo vision [24].

For this project, stereo vision based systems have the following advantages over the laser scanning systems:

- The cameras do not need to be moved in order to determine the position of the target object
- Multiple targets can be resolved simultaneously Stereovision allows the simultaneous triangulation of all objects in the field of view of both cameras at any one time
- Specialised illumination (such as a laser stripe) is not required, a simple lamp providing uniform lighting is sufficient
- The illumination source is stationary; it does not have to move in order to highlight areas of interest.
- Feature based recognition allows for intelligent identification of the teats of the animal, giving scope for an adaptive system that learns teats patterns of problematic animals and adapts accordingly.
- Software based location allows easy implementation of different approaches and different algorithms without requiring mechanical modifications.

- Software analysis enables information regarding the orientation angle, the size and the shape of the teats to be extracted from the images. Problems, such as low udders, touching teats or missing teats can be quickly identified.

For the purposes of this project the most significant benefit of a stereo vision based teat identification system is the reduced time overhead. All four teats can be identified from a single pair of images and their locations calculated. A stream of images (video) allows the teat coordinates to be known in approximate real time, thus allowing the simultaneous attachment of all four milking cups.

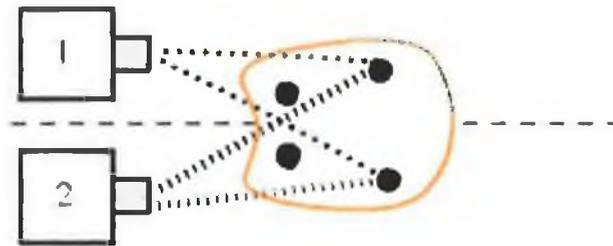


Fig 1.8 – Stereo Camera Rig, view of udder from rear, unobstructed view of all teats for both cameras

Fig 1.8 illustrates the setup of a stereo camera rig. Two cameras are positioned behind the cow, viewing the udder. In this scenario all four teats are visible to both of the cameras; this allows the positions of all the teats to be triangulated. Fig 1.9 illustrates the problem of occlusion: the objects must be in view of both cameras for their positions to be determined, but in this case the teats closer to the camera are obstructing the view of the far teats.

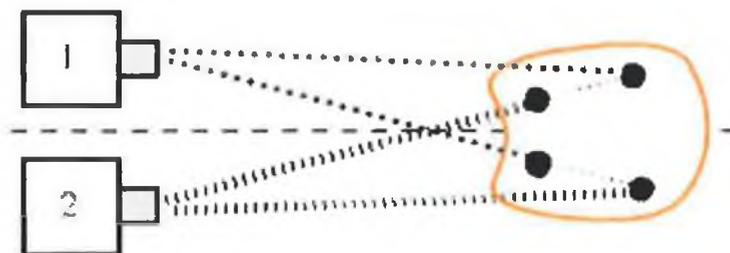


Fig 1.9 – Stereo Camera Rig, view of udder from rear, far teats are occluded

In Fig 1.9 the far teats are occluded in both views, but it is only necessary for one view to be occluded to inhibit teat identification. Occlusion could be caused by the presence of

the end effector applying the milking cups and the milking cups themselves. Since the manipulator will approach the udder through the back legs of the cow the location of the camera rig has been chosen as in Fig 1.10. This position will help prevent the view of the udder being obscured by the end effector. The viewing angle of the cameras is chosen so that the cameras are looking upwards at the udder. This prevents the front teats obscuring the view of the rear teats.

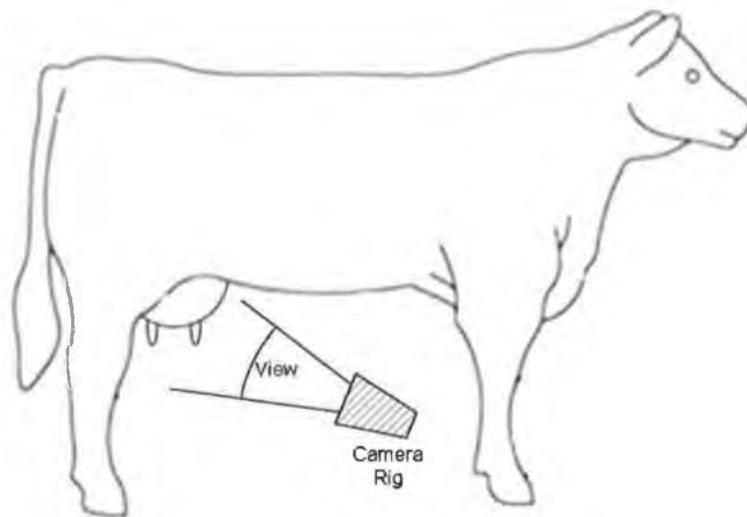


Fig 1.10 – Position of stereo camera rig under cow and its viewing angle

1.2.2.c - IceRobotics Teat Tracker

Discussed in detail in Chapter 3, the IceRobotics Teat Tracker is a stereo vision system designed for determining the 3D positions of cow teats. It has been acquired with the intention of incorporating it into the Rotary AMS. The system has been analysed to determine whether it is suitable for providing simultaneous teat coordinates to the cup application robot. Its performance is assessed based on its ability to correctly identify teats in the scene and also on the accuracy of position estimates that it provides for each teat.

1.2.2.d – Teat Angle Estimation

The IceRobotics system does not provide information regarding the angle at which a teat is orientated. As discussed in Section 1.2.1.b it is necessary to detect when a teat is orientated at a large angle to the vertical plane so that the end-effector can adjust the cup during the application to the teat. This functionality has been developed and the algorithm to implement it is described in detail in Chapter 4. Stereo vision reconstruction methods (Chapter 2.1.2.c) are used to identify two 3D coordinates along the central axis of the teat. These two points describe a vector which defines the orientation angle of the teat. The art of recovering three dimensional geometry from images is long established. T.A. Clarke and J.G. Fryer published a paper which documents its history and it's development [17]. Modern camera calibration techniques have developed in the past twenty years since Tsai's radical publication on the subject in 1987 [19]. There have been numerous papers and reference texts published since, most notably for this project the work of Hartley and Zisserman [25] and Olivier Faugeras [26].

Chapter 2 – Theory

2.1 - Three Dimensional Computer Vision

This section provides background theory to the techniques used throughout this work in determining the three dimensional (3D) location of an object in space based upon the position the object appears in a pair of two dimensional (2D) digital images, a process known as reconstruction. The pairs of images containing the object are captured at the same instant from two different viewpoints. This is known as a stereo pair. Reconstruction of the scene is achieved using triangulation as discussed in Section 2.1.2.c. The basis of image formation and the projection of points through a camera centre, required for reconstruction, are presented in Section 2.1.1 which contains the camera model and the transformations governing the transformations through the camera centre. The projective camera matrix, which transforms a point in 3D World space to a 2D image coordinate, is defined in this section. A method for evaluating the numerical components of this matrix is described in Section 2.1.3.a - single camera calibration. Reconstruction from two camera views requires information about the positions and the orientations of the cameras relative to each other. Section 2.1.2 describes two-view geometry, in which the information regarding the transformation from one camera to another is contained within the projective camera matrices. The relationship between a point in the image of one camera and the corresponding point in the image of the other camera (the two points are the images of the same World point) is detailed. Epipolar geometry defines a line in the image to which a point corresponding to a point in the other image of the stereo pair must belong. This principle is used in limiting the search area when finding point correspondences. The fundamental matrix, which represents the epipolar geometry algebraically, is defined in Section 2.1.2.b and methods for its numeric evaluation are presented in section 2.1.3.b. Throughout this work the 3D reconstructions are carried out using the Caltech Camera Calibration Toolbox for Matlab [27], a freely available software tool with reconstruction functions. A brief insight to the theory behind its operation is given in the following sections.

2.1.1 – Single View Geometry

2.1.1.a - Pinhole Camera Model

A camera maps points in the 3D world to a 2D image; 3D points are projected onto an image plane by the camera lens. The image plane is made up of an array of light sensors. A basic model of this projection is seen in Fig 2.1.

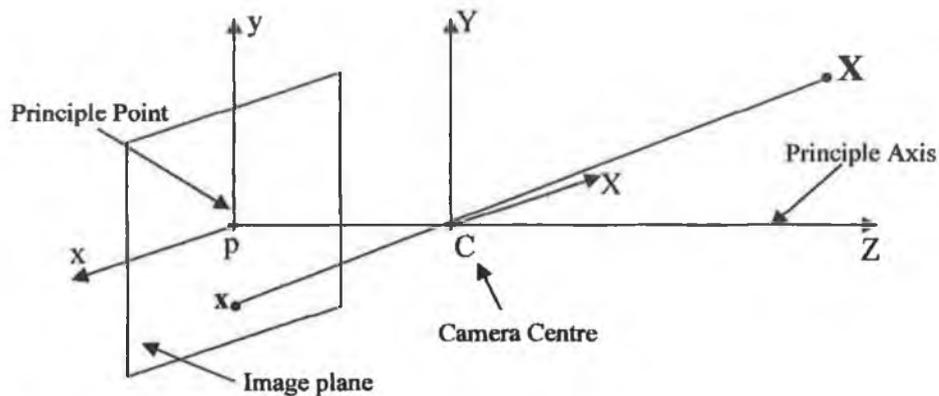


Fig 2.1 – Pinhole camera model

In diagram in Fig 2.1 the origin of the world coordinate frame has been placed on point C , the camera centre. The line perpendicular to the image plane through the camera centre is known as the principle axis. The point where this line intersects the image plane is known as the principle point. The distance between the principle point and the camera centre is the focal length of the camera, and is denoted as f in Fig 2.2. A point $X = (X, Y, Z)$ is projected through the camera centre (the 'pinhole') and onto the image plane at $x = (x, y)$. Fig 2.1 is a view of this projection in the YZ plane. The projected image is inverted because there is a single lens, the pinhole.

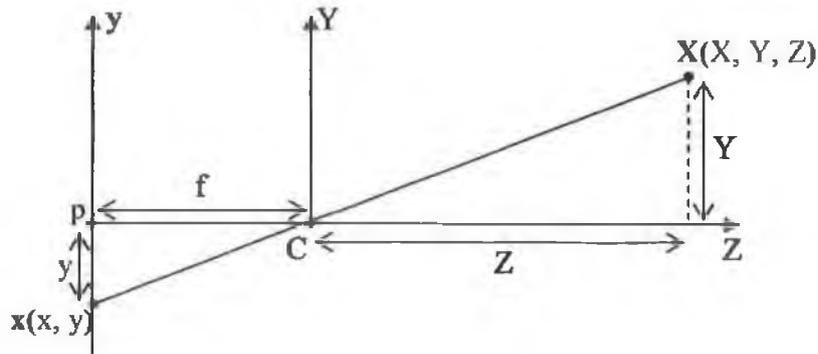


Fig 2.2 – Section View: Pinhole projection in the YZ plane

In Fig 2.2 the triangles $\triangle CYZ$ and $\triangle Cyp$ are similar triangles. Therefore the ratios of the side lengths are the same for both triangles:

$$\frac{y}{f} = \frac{Y}{Z} \quad (2.1)$$

$$\Rightarrow y = \frac{fY}{Z} \quad (2.2)$$

By observing the projection from the XZ plane it can be similarly shown that:

$$x = \frac{fX}{Z} \quad (2.3)$$

2.1.1.b - Projective Camera Matrix

The previous defines the transformation from the 3D world coordinate to the 2D image coordinate as:

$$(\mathbf{X}, \mathbf{Y}, \mathbf{Z})^T \mapsto \left(\frac{fX}{Z}, \frac{fY}{Z} \right)^T \quad (2.4)$$

Representing the world and image coordinates by homogeneous vectors allows the projection to be expressed as a linear mapping between their homogeneous coordinates [28]:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX \\ fY \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (2.5)$$

\mathbf{x} \mathbf{P} \mathbf{X}

This can be represented by:

$$\mathbf{x} = \mathbf{P}\mathbf{X} \quad (2.6)$$

where \mathbf{P} is the 3x4 homogeneous camera projection matrix.

The previous models and expressions imply that the origin of the image coordinate frame is at the principal point. This is not necessarily the case, in practise it is the upper left corner of the image that is taken as the origin; this would be the lower right corner of the image plane in Fig 2.1 (the image is inverted). A projection matrix can be altered to accommodate a general offset.

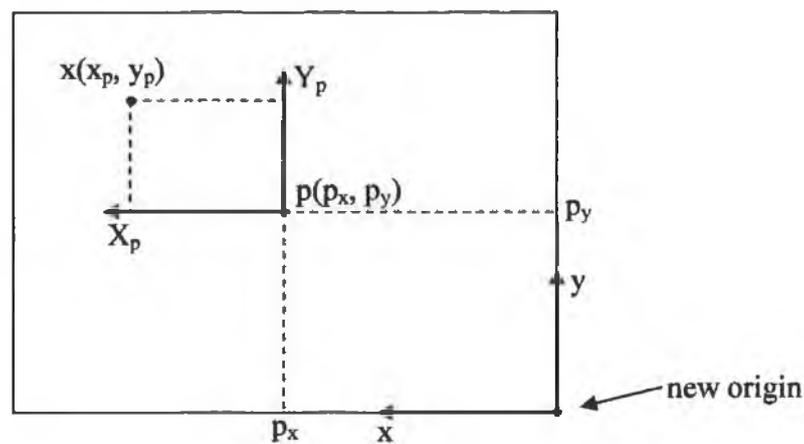


Fig 2.3 – Image plane with origin at lower right corner

Fig 2.3 illustrates an offset of the origin of the image coordinate system. The image point $x(x_p, y_p)$ with respect to the principal point is described as $(x_p + p_x, y_p + p_y)$ with respect to the new origin. This gives the new general transformation:

$$(X, Y, Z)^T \mapsto \left(\frac{fX}{Z} + p_x, \frac{fY}{Z} + p_y \right)^T \quad (2.7)$$

Where (p_x, p_y) is the coordinate of the principal point with respect to the offset image plane origin. This is expressed in homogeneous coordinates as:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} f & p_x & 0 \\ f & p_y & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (2.8)$$

2.1.1.c - Intrinsic Parameters

K , the camera calibration matrix is written as:

$$K = \begin{bmatrix} f & p_x \\ f & p_y \\ 0 & 0 & 1 \end{bmatrix} \quad (2.9)$$

From Equation 2.8 it follows that:

$$\mathbf{x} = K \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = K[I | 0] \mathbf{X}_{cam} \quad (2.10)$$

where \mathbf{X}_{cam} is a coordinate in the camera coordinate frame. The camera coordinate frame is defined such that its origin is placed on the camera centre and the Z-axis is collinear with the principle axis of the camera. The camera calibration matrix maps a point in the

camera coordinate frame to point in the image plane. The previous definition of K implies that there is equal scaling in both axial directions. This is not necessarily the case; for example, a CCD camera may have non square pixels. This updates the K to a more general form:

$$K = \begin{bmatrix} \alpha_x & & p_x \\ & \alpha_y & p_y \\ & & 1 \end{bmatrix} \quad (2.11)$$

where $\alpha_x = fm_x$ and $\alpha_y = fm_y$

where $m_x : m_y$ is the aspect ration of the image plane.

A final, more general solution for K is:

$$K = \begin{bmatrix} \alpha_x & s & p_x \\ & \alpha_y & p_y \\ & & 1 \end{bmatrix} \quad (2.12)$$

where s is referred to as the skew parameter. This takes into account the possibility that the X and Y axes of the image plane are not perpendicular. Usually this is not the case, so the value of s will be zero.

2.1.1.d - Extrinsic Parameters

The camera calibration matrix, K , contains the intrinsic parameters of the camera. It maps points from the camera coordinate frame to the image plane. Points in the World coordinate frame are transformed to the camera coordinate frame by a series of rotations and translations. The matrix containing this transformation is called the extrinsic matrix.

The following expression transforms a homogeneous point \mathbf{X} in world coordinates to the homogeneous point \mathbf{X}_{cam} in the camera coordinate frame by pre-multiplying \mathbf{X} by the extrinsic matrix:

$$\mathbf{X}_{\text{cam}} = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ 0 & 1 \end{bmatrix} \mathbf{X} \quad (2.13)$$

Where \mathbf{R} is a 3x3 rotation matrix representing the orientation of the world coordinate frame with respect to the camera coordinate frame. $\tilde{\mathbf{C}}$ is the origin of the camera coordinate frame in inhomogeneous world coordinates.

The matrix \mathbf{P} (Equation. 2.6) defined such that $\mathbf{x} = \mathbf{P}\mathbf{X}$, is now further defined as:

$$\mathbf{P} = \mathbf{K}[\mathbf{R} \mid \mathbf{t}] \quad (2.14)$$

where:

$$\mathbf{t} = -\mathbf{R}\tilde{\mathbf{C}} \quad (2.15)$$

2.1.2 - Two View Geometry

2.1.2.a - Epipolar Geometry

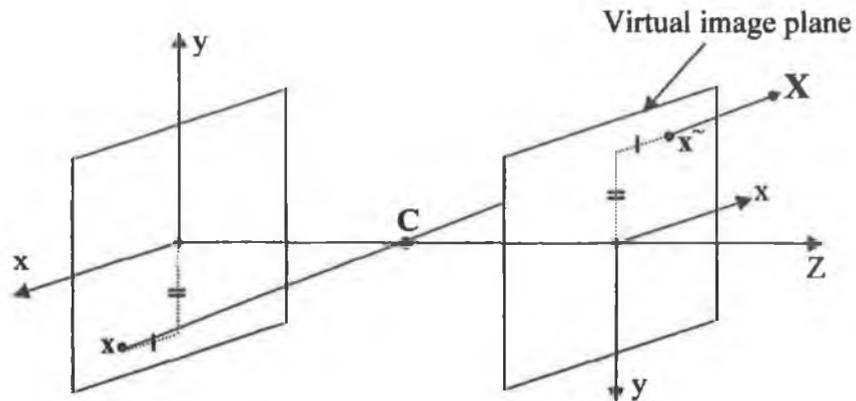


Fig 2.4 - Fig 2.1 revisited: Intersection with imaginary image in front of camera centre

Fig 2.4 illustrates that a virtual image plane can be placed in front of the camera centre and the ray projected from the point X will intersect at the point \tilde{x} with equivalent component lengths as the point x in the real image plane. For further analysis and ease of understanding the plane containing \tilde{x} will be used as the image plane.

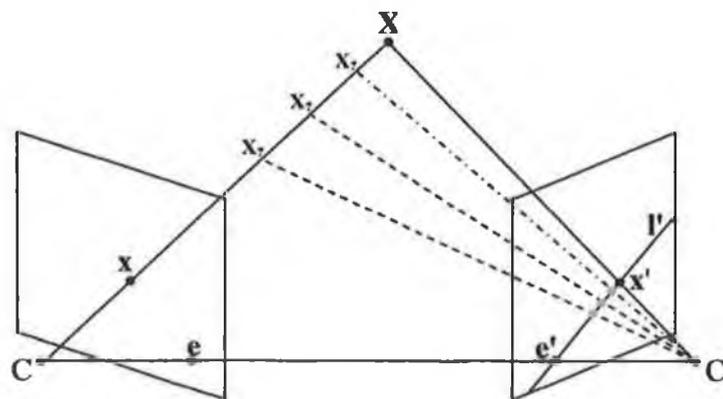


Fig 2.5 - Two Camera views, (image planes in front of camera centres)

Fig 2.5 illustrates two separate cameras viewing a single point X located in 3D space. The world point X is projected onto the left image plane through the left camera centre C to the image coordinate x . It is projected onto the right image plane through the right camera

centre C' as \mathbf{x}' . The image of the right camera centre in the left image plane is the left epipole, \mathbf{e} . The right epipole, \mathbf{e}' , is the image of the left camera centre in the right image plane. The line \mathbf{l}' passes through both \mathbf{e}' and \mathbf{x}' and is known as the epipolar line. Fig 2.5 illustrates the principles of epipolar geometry; if the point \mathbf{X} is unknown, but \mathbf{x} is known, then it is known that \mathbf{X} must lie upon a ray projecting through the camera centre C , i.e. the $\mathbf{X}_?$ variables are all possible locations of \mathbf{X} . All possible values of \mathbf{X} project onto the right image plane, through C' , along the epipolar line \mathbf{l}' . This means that for any point in one image plane there is a single line in the other image plane to which a corresponding point can belong. The epipolar line \mathbf{l}' is the cross product of the epipole \mathbf{e}' and the image point \mathbf{x}' , i.e.

$$\mathbf{l}' = \mathbf{e}' \times \mathbf{x}' \quad (\text{See Verification 2.1}) \quad (2.16)$$

This may be written as:

$$\mathbf{l}' = [\mathbf{e}']_x \mathbf{x}' \quad (2.17)$$

where:

$$[\mathbf{e}']_x = \begin{bmatrix} 0 & -e'_z & e'_y \\ e'_z & 0 & -e'_x \\ -e'_y & e'_x & 0 \end{bmatrix} \quad (2.18)$$

Verificaion 2.1: The cross product of two homogenous 2D vectors results in a line of which both points are an element

Γ' is of form: $ax + by + c = 0$ which is equal to $(a, b, c)^T$
 $\mathbf{e}'(e'_x, e'_y)$ and $\mathbf{x}'(x'_x, x'_y)$ are on the line Γ'

$$\mathbf{e}' \times \mathbf{x}' = \begin{vmatrix} i & j & k \\ e'_x & e'_y & 1 \\ x'_x & x'_y & 1 \end{vmatrix} = i(e'_y - x'_y) - j(e'_x - x'_x) + k(e'_x x'_y - e'_y x'_x)$$

Let the determinant equal zero, then:

$$i(e'_y - x'_y) - j(e'_x - x'_x) + k(e'_x x'_y - e'_y x'_x) = 0$$

Which is of the form: $ax + by + c = 0$

Where $-\frac{a}{b} = \frac{(q_y - p_y)}{(q_x - p_x)} =$ slope of Γ'

Substituting in the values of \mathbf{e}' and \mathbf{x}' verifies that they are on the line:

$$e'_x(e'_y - x'_y) - e'_y(e'_x - x'_x) + 1(e'_x x'_y - e'_y x'_x) = 0 \quad \text{True!}$$

$$x'_x(e'_y - x'_y) - x'_y(e'_x - x'_x) + 1(e'_x x'_y - e'_y x'_x) = 0 \quad \text{True!}$$

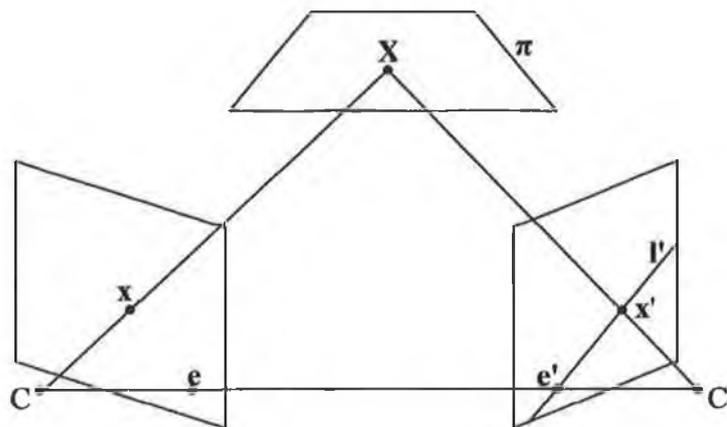


Fig 2.6 – Transfer via a plane π [29]

In Fig 2.6 π is any plane in space that does not pass through either of the camera centres. The point \mathbf{x} , in the 2D left image plane, projects to \mathbf{X} on the 2D plane π . This point \mathbf{X} projects to \mathbf{x}' in the 2D right image plane. There is therefore a direct linear transformation, or a 2D homography, between the points \mathbf{x} and \mathbf{x}' . This homography is denoted as \mathbf{H}_π such that:

$$\mathbf{x}' = \mathbf{H}_\pi \mathbf{x} \quad (2.19)$$

It was shown previously that:

$$\mathbf{l}' = [\mathbf{e}']_x \mathbf{x}' \quad (2.20)$$

This can now be written as:

$$\mathbf{l}' = [\mathbf{e}']_x \mathbf{H}_\pi \mathbf{x} \quad (2.21)$$

\mathbf{F} , the fundamental matrix is defined as:

$$\mathbf{F} = [\mathbf{e}']_x \mathbf{H}_\pi \quad (2.22)$$

Therefore:

$$\mathbf{l}' = \mathbf{F}\mathbf{x} \quad (2.23)$$

The point \mathbf{x}' is on the line \mathbf{l}' , therefore:

$$\mathbf{x}'^T \mathbf{l}' = 0 \quad (\text{See Verification 2.2}) \quad (2.24)$$

Substituting in the previous definition of \mathbf{l}' (Equation. 2.23) gives the following correspondence condition between \mathbf{x} and \mathbf{x}' :

$$\mathbf{x}'^T \mathbf{F}\mathbf{x} = 0 \quad (2.25)$$

Verificaion 2.2: $\mathbf{x}'^T \mathbf{l}' = 0$ if \mathbf{x}' is an element of the line \mathbf{l}'

\mathbf{l}' is of the form: $ax + by + c = 0$ which is equal to $\begin{bmatrix} a \\ b \\ c \end{bmatrix}$

\mathbf{x}' is of the form $\begin{bmatrix} x'_x \\ x'_y \\ 1 \end{bmatrix}$ therefore: $\mathbf{x}'^T \mathbf{l}' = \begin{bmatrix} x'_x \\ x'_y \\ 1 \end{bmatrix}^T \begin{bmatrix} a \\ b \\ c \end{bmatrix} = ax'_x + bx'_y + c$

If \mathbf{x}' is an element of the line \mathbf{l}' then the previous statement must equate to zero.

2.1.2.c – Reconstruction

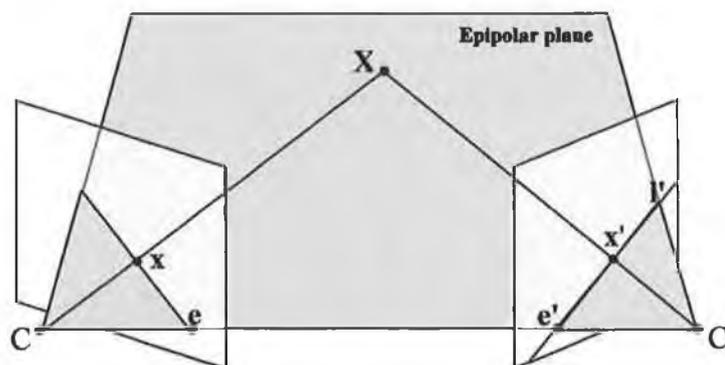


Fig 2.7 – x, x' and X are all in the same plane, the epipolar plane [30]

If the camera matrices (P and P') and the corresponding points (\mathbf{x} and \mathbf{x}') are known, the point \mathbf{X} can be triangulated. In Fig 2.7 it is illustrated that the points \mathbf{x} , \mathbf{x}' and \mathbf{X} are all in the same plane, the epipolar plane. The back projection rays of \mathbf{x} and \mathbf{x}' though their respective camera centres will therefore have a point of intersection (except if the 3D point lies on the baseline - the line joining the two camera centres). The rays intersect at \mathbf{X} ; evaluating \mathbf{X} in this manner is known as stereo triangulation.

According to Equation 2.6 $\mathbf{x} = \mathbf{P}\mathbf{X}$ and $\mathbf{x}' = \mathbf{P}'\mathbf{X}$ where $\mathbf{x} = [x_x \ x_y \ 1]^T$, $\mathbf{x}' = [x'_x \ x'_y \ 1]^T$ and $\mathbf{X} = [X_x \ X_y \ X_z \ 1]^T$. These two equations can be combined into the form of:

$$\mathbf{A}\mathbf{X} = 0 \quad (2.26)$$

The cross product of an image point and a transformed World point is zero, as seen in Equation 2.27.

$$\mathbf{x} \times (\mathbf{P}\mathbf{X}) = 0 \quad (2.27)$$

The cross product expands to three equations:

$$x_x (\mathbf{p}_{row}^3 \mathbf{X}) - (\mathbf{p}_{row}^1 \mathbf{X}) = 0 \quad (2.28)$$

$$x_y (\mathbf{p}_{row}^3 \mathbf{X}) - (\mathbf{p}_{row}^2 \mathbf{X}) = 0 \quad (2.29)$$

$$x_x (\mathbf{p}_{row}^2 \mathbf{X}) - x_y (\mathbf{p}_{row}^1 \mathbf{X}) = 0 \quad (2.30)$$

Similarly

$$\mathbf{x}' \times (\mathbf{P}'\mathbf{X}) = 0 \quad (2.31)$$

yields:

$$x'_x (\mathbf{p}_{row}^{i3} \mathbf{X}) - (\mathbf{p}_{row}^{i1} \mathbf{X}) = 0 \quad (2.32)$$

$$x'_y (\mathbf{p}_{row}^{i3} \mathbf{X}) - (\mathbf{p}_{row}^{i2} \mathbf{X}) = 0 \quad (2.33)$$

$$x'_x (\mathbf{p}_{row}^{i2} \mathbf{X}) - x'_y (\mathbf{p}_{row}^{i1} \mathbf{X}) = 0 \quad (2.34)$$

where \mathbf{p}_{row}^i is the i^{th} row of the P matrix and $\mathbf{p}_{row}^{i'}$ is the i^{th} row of the P' matrix. The A matrix is formed with the four linear independent equations: Equations 2.28, 2.29, 2.32 and 2.33 such that:

$$\mathbf{A}\mathbf{X} = \begin{bmatrix} x_x \mathbf{p}_{row}^3 - \mathbf{p}_{row}^1 \\ x'_y \mathbf{p}_{row}^{i3} - \mathbf{p}_{row}^{i2} \\ x'_x \mathbf{p}_{row}^{i3} - \mathbf{p}_{row}^{i1} \\ x'_y \mathbf{p}_{row}^{i3} - \mathbf{p}_{row}^{i2} \end{bmatrix} \begin{bmatrix} X_x \\ X_y \\ X_z \\ 1 \end{bmatrix} = 0 \quad (2.35)$$

Equation 2.35 contains four equations and four unknowns. A non-zero solution for \mathbf{X} is determined as the unit singular vector corresponding to the smallest singular value of \mathbf{A} (equivalently, the solution is the unit eigenvector of $\mathbf{A}^T\mathbf{A}$ with least eigenvalue). A point that lies on the baseline cannot be triangulated. As seen in Fig 2.6 the image points corresponding to the base line are the epipoles \mathbf{e} and \mathbf{e}' for the left and right cameras respectively. The back projected rays are collinear and therefore have no single point of intersection.

2.1.3 - Camera Calibration

Camera calibration is the process of numerically estimating the parameter values of the camera matrices

2.1.3.a - Single Camera Calibration

In the case of a single camera the camera projection matrix, \mathbf{P} , maps a point \mathbf{X} in world coordinates to a point \mathbf{x} in the image.

$$\mathbf{x}_i = \mathbf{P}\mathbf{X}_i \quad (2.36)$$

where \mathbf{x}_i is the set of projected image points corresponding to the set of world points \mathbf{X}_i

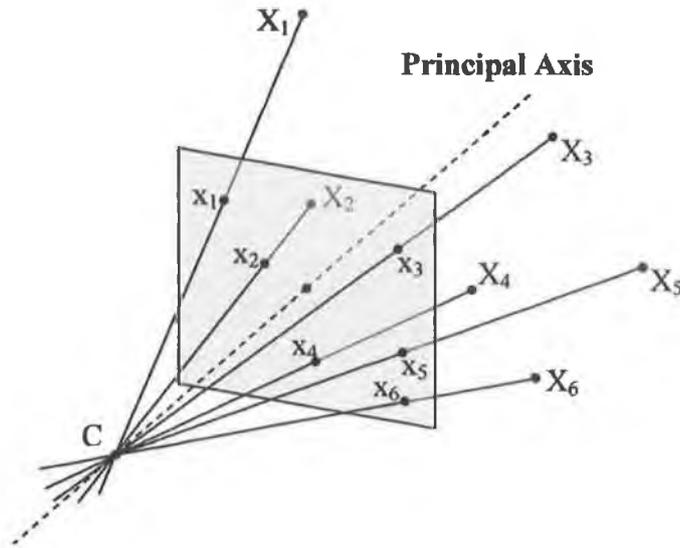


Fig 2.8 – Mapping of set of points X_i to x_i

Each correspondence between a world point X_i and x_i yields the equation:

$$\begin{array}{c}
 \begin{bmatrix} su_i \\ sv_i \\ s \end{bmatrix} = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix} \\
 \mathbf{x} \qquad \qquad \mathbf{P} \qquad \qquad \mathbf{X}
 \end{array} \tag{2.37}$$

which expands to:

$$\begin{bmatrix} su_i \\ sv_i \\ s \end{bmatrix} = \begin{bmatrix} P_{11}x_i + P_{12}y_i + P_{13}z_i + P_{14} \\ P_{21}x_i + P_{22}y_i + P_{23}z_i + P_{24} \\ P_{31}x_i + P_{32}y_i + P_{33}z_i + P_{34} \end{bmatrix} \tag{2.38}$$

which further expands to give two linearly dependent equations:

$$\begin{array}{l}
 u(P_{31}x_i + P_{32}y_i + P_{33}z_i + P_{34}) = P_{11}x_i + P_{12}y_i + P_{13}z_i + P_{14} \\
 v(P_{31}x_i + P_{32}y_i + P_{33}z_i + P_{34}) = P_{21}x_i + P_{22}y_i + P_{23}z_i + P_{24}
 \end{array} \tag{2.39}$$

Each correspondence between a world point and an image point gives two linear dependant equations in which the values of X_i (X_1, X_2, X_3) and $x(u,v)$ are known. This leaves twelve unknowns ($P_{11} \rightarrow P_{34}$). If six point correspondences (that are not co-planar) are known there are a total of 12 linear dependant equations and the P matrix can be solved. (In fact only 11 equations are required to solve P as it only has 11 degrees of freedom due to the scaling factor of homogeneous coordinates). In practice more than six point correspondences will be used, giving an over-determined system of equations. Due to errors that will be introduced in any real world measurements of the calibration scene, there will not be an exact solution for the P matrix. If a large number of point correspondences are used then an error that is present will have less effect on the minimised solution of the over determined system. The system can be solved with the constraint that the 3D geometric error is minimised. [31]. To generate a set of X_i coordinates there must be known 3D scene geometry. In practice a calibration object is typically used to provide this knowledge. A calibration object that is simple to construct, and that can provide accurate scene information is shown in Fig 2.9. It is a printed 'Chessboard' pattern on a planar surface. Printing using a standard desktop printer, one can construct a calibration grid of which the corner points are accurately and consistently spaced. Multiple images with the calibration board in different orientations (Fig 2.10) must be used to ensure that the points are not co-planar. If the points are co-planer then a unique solution for P cannot be determined. An accurate calibration pattern incorporating multi-planer geometry is difficult to construct, however, should one have such a pattern, only a single image is required for calibration.

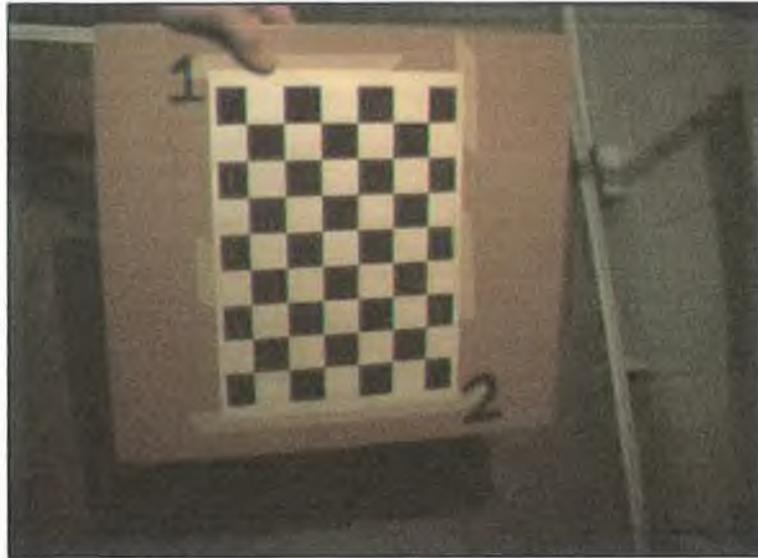


Fig 2.9 – ‘Chessboard’ calibration pattern

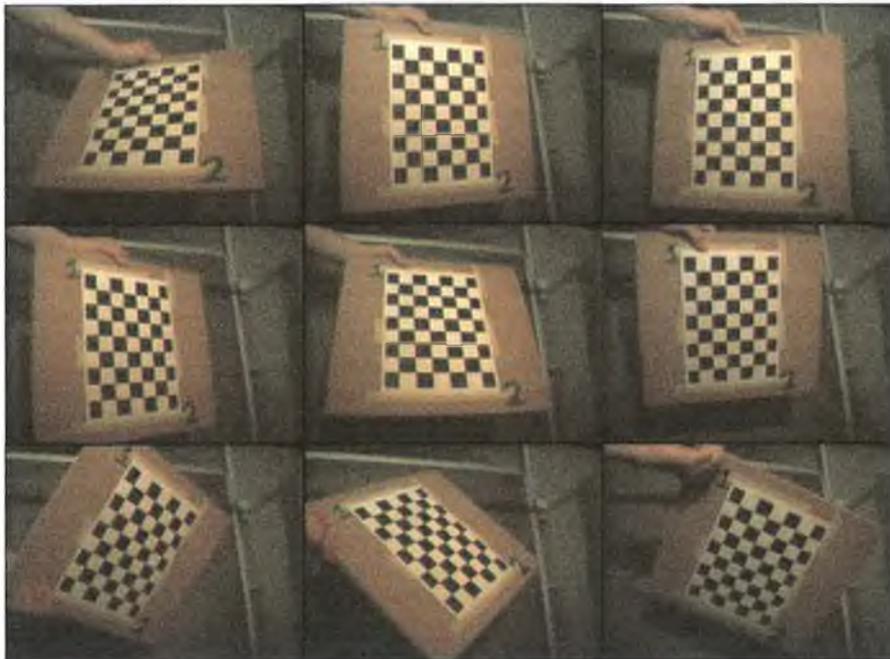


Fig 2.10 - Multiple orientations of calibration pattern: non co-planar world points

2.1.3.b - Two Cameras and the Fundamental Matrix

The fundamental matrix is previously defined (Equation. 2.25) as:

$$\mathbf{x}'^T \mathbf{F} \mathbf{x} = 0$$

where $\mathbf{x} \leftrightarrow \mathbf{x}'$ is a pair of corresponding points in two images. If enough point matches are known (seven or more) then Equation 2.25 can be used to determine the values of the Fundamental matrix. Re-writing the equation with $\mathbf{x} = (x, y, 1)^T$ and $\mathbf{x}' = (x', y', 1)^T$ gives:

$$\begin{bmatrix} x' & y' & 1 \end{bmatrix} \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0 \quad (2.40)$$

which expands as:

$$\begin{bmatrix} x' & y' & 1 \end{bmatrix} \begin{bmatrix} xf_{11} + yf_{12} + f_{13} \\ xf_{21} + yf_{22} + f_{23} \\ xf_{31} + yf_{32} + f_{33} \end{bmatrix} = 0 \quad (2.41)$$

which gives the equation:

$$x'xf_{11} + x'yf_{12} + x'f_{13} + y'xf_{21} + y'yf_{22} + y'f_{23} + xf_{31} + yf_{32} + f_{33} = 0 \quad (2.42)$$

If $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ are a set of point matches then a set of equations is described:

$$\begin{bmatrix} x'_1 & y'_1 & 1 \\ \vdots & \vdots & \vdots \\ x'_n & y'_n & 1 \end{bmatrix} \begin{bmatrix} x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = 0 \quad (2.43)$$

A
f

To solve for f a large number of point correspondences (around 20) are used to give an over determined system of equations (there are more equations than unknowns). Due to noise, the point correspondences are not exact so no exact solution exists for f . A least squares solution can be found using numerical methods, where the solution is the unit eigenvector corresponding to the smallest eigenvalue of $A^T A$ [32].

2.2 – Detection of Lines in Images

Teats are isolated in the images by establishing the group of pixels that define the outer edge of the teat. An edge detector [33] can help identify these points. An edge detector reduces the amount of data in an image, filtering out useless information, while preserving important structural information (the edges) in the image. An edge occurs at a boundary in an image, a boundary is characterised by a large difference in the intensity of one pixel to the next. Once an edge detector has identified the pixels in the images representing edges in the image, a line detector can be used to determine which edge pixels correspond to straight lines in the image.

2.2.1 – Canny Edge Detector

The Canny edge detector was designed to meet three criteria: 1) Good detection: as many real edges as possible should be detected in the image. 2) Good Localisation: the detected edge locations should be as close as possible to the actual locations of the edges in the images. 3) Minimal Response: there should only be one line identified for a single edge [34] [35]. The steps taking to achieve these aims are [36]:

- Image smoothing using a Gaussian filter to reduce noise
- The local gradient and edge direction are computed for each pixel
- Non-maximal suppression of ridges in gradient magnitude map found in previous step
- Ridge pixels are thresholded using an upper and lower threshold to define strong and weak edges

- Edge linking incorporates into the overall edge image the edges that are immediate neighbours to strong edge pixels

2.2.1.a - Image smoothing using a Gaussian filter to reduce noise

A Gaussian lowpass filter is first applied to smooth the image [37]. Applying the filter will help prevent Gaussian noise in the image being mistaken as an edge by the detector. Fig 2.11 contains a convolution mask [38] that is a discrete approximation to a Gaussian function with a standard deviation, σ , of 1.4. The centre of the mask is passed over each pixel in the image, recalculating the pixel value based on the convolution mask. The new intensity value of the active pixel (located under the centre square of the mask) is calculated as the sum of the current intensity value of that pixel and of its neighbouring pixels all multiplied by the corresponding weights contained in the mask.

0.0105	0.0227	0.0293	0.0227	0.0105
0.0227	0.0488	0.0629	0.0488	0.0227
0.0293	0.0629	0.0812	0.0629	0.0293
0.0227	0.0488	0.0629	0.0488	0.0227
0.0105	0.0227	0.0293	0.0227	0.0105

Fig 2.11 – Gaussian Lowpass filter mask, $\sigma = 1.4$

The values in the mask of Fig 2.11 are calculated according to Equation 2.45, which is derived from Equation 2.44 [39].

$$G(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}} \quad (2.44)$$

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (2.45)$$

Equation 2.45 is a circularly symmetric Gaussian function, where (x, y) represents the 2D location of a point relative to the mean. The centre square of the grid (containing the value 0.0812) has the coordinate $(x = 0, y = 0)$. The square in the upper right corner (containing the value 0.0105) has the coordinate $(x = 3, y = 3)$. The mask is filled according to Equation 2.45.

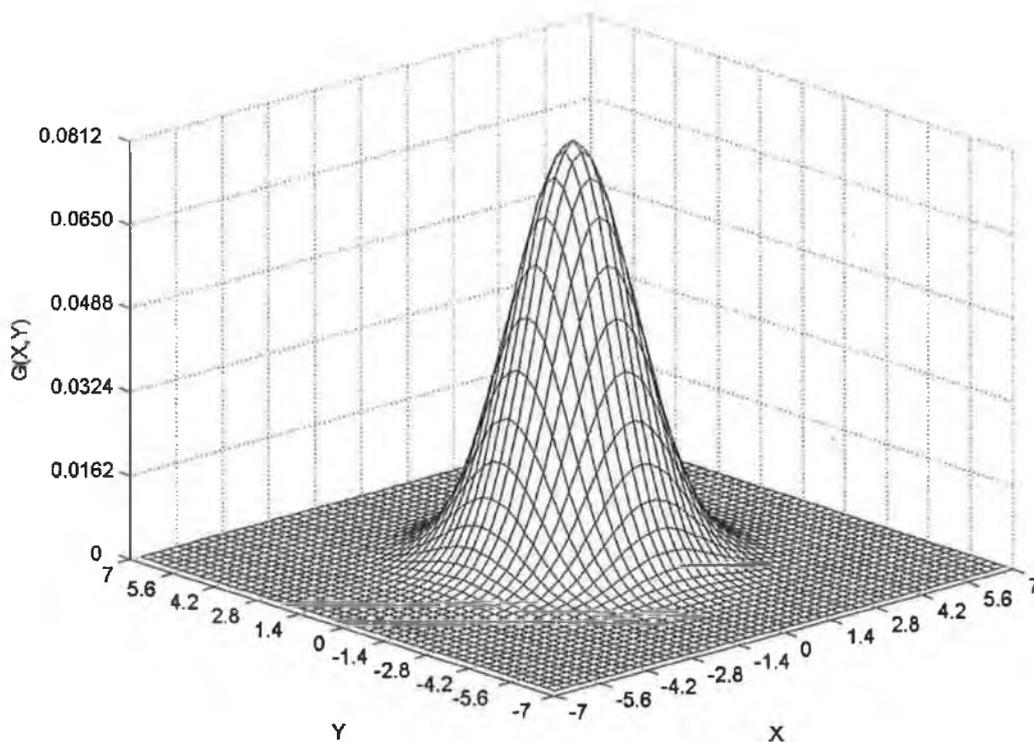


Fig 2.12 - 2D Gaussian distribution, mean = (0, 0), $\sigma = 1.4$

Fig 2.12 is a 3D visual representation of a 2D Gaussian distribution with a standard deviation of 1.4. The surface illustrates the proportions in which the new intensity of the centre pixel $(0, 0)$ is calculated from the existing pixel intensity value and that of its surrounding pixels. Applying this function to all the pixels blurs the image, reducing the effect of high frequency noise while preserving the image structure. Fig 2.13 is a sample image containing Gaussian noise, seen as speckles in what would otherwise be a uniform grey background. Fig 2.14 is the resulting image after applying a convolution mask such as the one in Fig 2.11. The smoothing of the image, and the noise reduction, is seen in the zoomed view on the right of Fig 2.14.

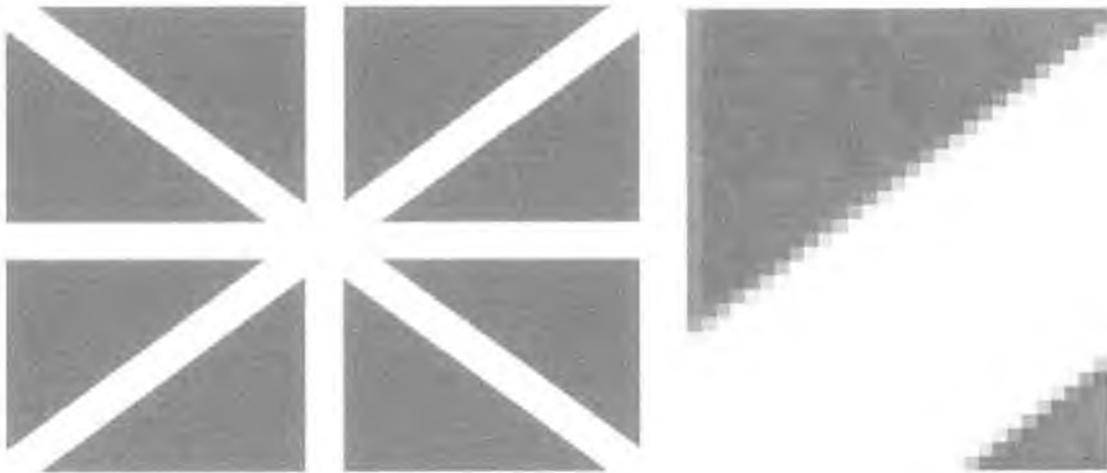


Fig 2.13 – Sample image containing Gaussian noise, zoomed view on right

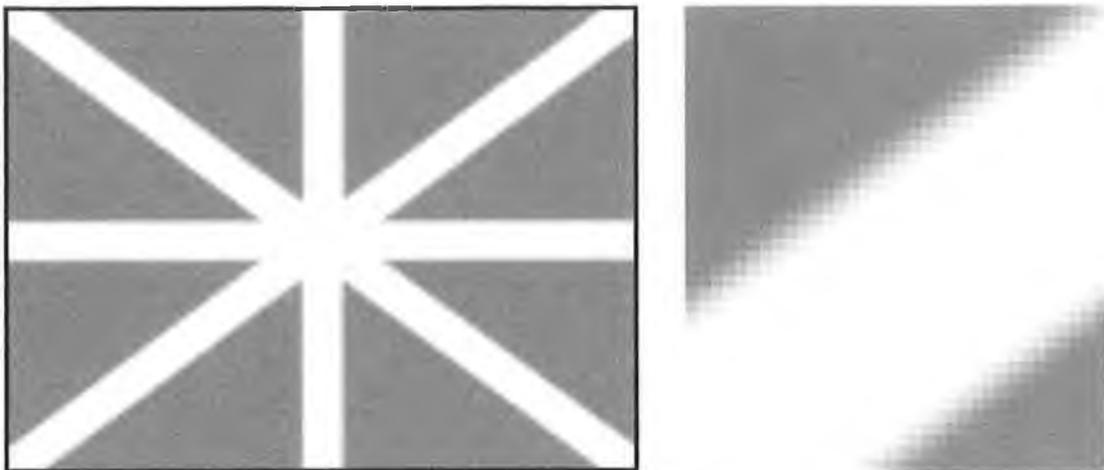


Fig 2.14 – Image after applying low pass filter, zoomed view on right

2.2.1.b - Local gradient and edge direction

The gradient of a pixel is how much its intensity value varies from that of its neighbouring pixels. Fig 2.15 is a plot of the intensity values of a horizontal row of pixels passing through an edge. The edge is seen as the change of intensity from grey to white in the pixel row.

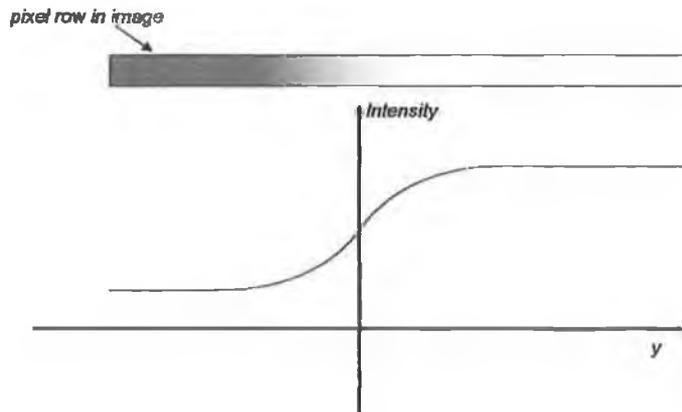


Fig 2.15 – Plot of intensity values along a row of pixels

Fig 2.16 is a plot of the derivate of the intensity with respect to the horizontal pixel location. The peak in the plot corresponds with the edge of the image.

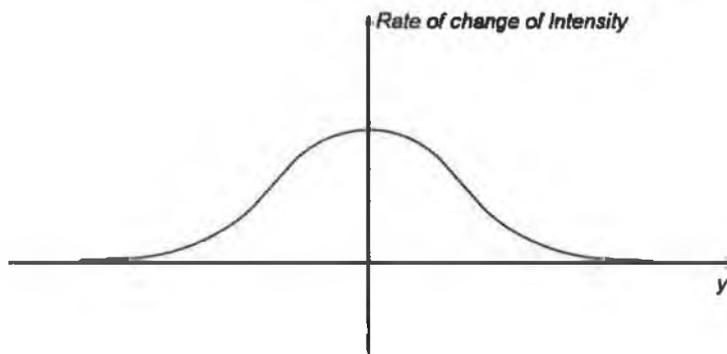


Fig 2.16 – Derivative of intensity with respect to horizontal pixel location

The derivative of intensities in an image can be found using convolution masks such as those in Fig 2.17. G_x on the left detects vertical edges; G_y on the right detects horizontal edges. Fig 2.18 contains the results of applying the two convolution masks.

-1	0	1
-2	0	2
-1	0	1

1	2	1
0	0	0
-1	-2	-1

Fig 2.17 – Derivative convolution masks, G_x on left, G_y on right

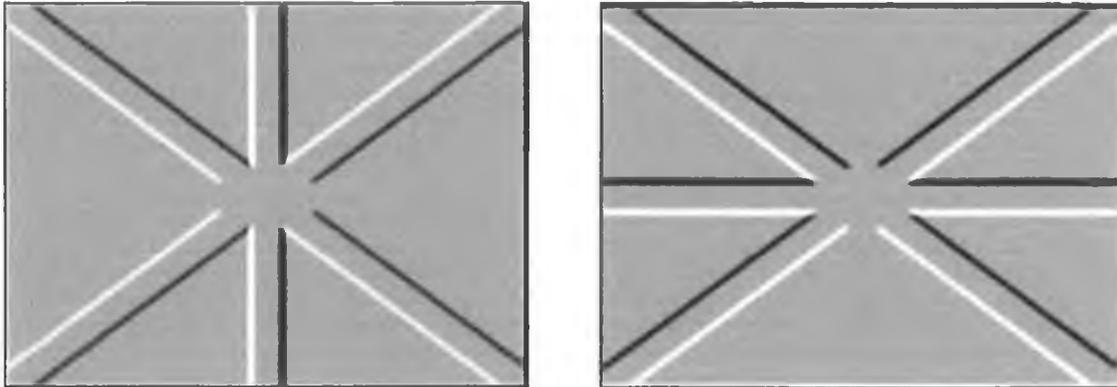


Fig 2.18 – Derivate Results: convolution of G_x on left, convolution of G_y on right

The convolution mask moves from left to right and from top to bottom in the image. It is seen in Fig 2.18 that moving from a grey region to white region results in a high (white) intensity output. Conversely, moving from a white to a grey region results in a low (black) intensity output. By combining the derivatives in both directions, and applying a threshold so that the maximum and minimum intensities the image seen in Fig 2.19 is produced.

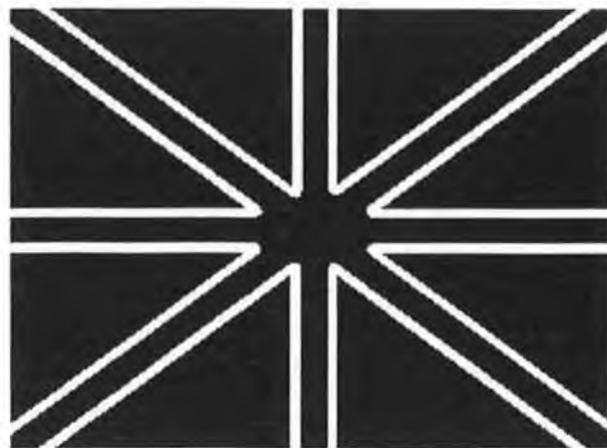


Fig 2.19 – Combination of derivatives in both horizontal and vertical directions

The local gradient is calculated for each pixel as:

$$g(x, y) = \sqrt{G_x^2 + G_y^2} \quad (2.46)$$

The edge direction is calculated at each point as:

$$\alpha(x, y) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \quad (2.47)$$

The edge direction is the angle of inclination of the edge in the image. Each pixel has eight neighbouring pixels; therefore there are only four discrete directions: horizontal ($0^\circ/180^\circ$), vertical ($90^\circ/270^\circ$), positive diagonal ($45^\circ/225^\circ$) and the negative diagonal ($135^\circ/315^\circ$). Each pixel is assigned the direction which best approximates the values obtained using Equation 2.47.

2.2.1.c - Non-maximal suppression

The edge pixels found in Fig 2.19 form thick lines. The pixels correspond to an intensity ridge made up of the start of the edge, the middle of the edge (top of ridge) and the end of the edge as the convolution mask passed through it. Non-maximal suppression is used to set to zero all pixels that are not on top of the ridge to zero. This gives a thin line in the output.

2.2.1.d - Thresholds Applied

The ridge pixels are thresholded using two thresholds, $T1$ and $T2$, with $T1 < T2$. Ridge pixels with a value greater than $T2$ are strong edge pixels. Ridge pixels with a value between $T1$ and $T2$ are weak edge pixels.

2.2.1.e - Edge Linking

The strong edges are followed using the edge directions previously found. If a weak edge pixel is connected to a strong edge pixel in the correct direction then it is considered to be a strong edge. In this way a pixel value of greater than $T2$ is required to start following an edge, but the edge is followed until a value less than $T1$ is encountered.

The output of the Canny edge detector is seen in Fig 2.20, a binary coded image in which a pixel is either an edge pixel or it is not an edge pixel.

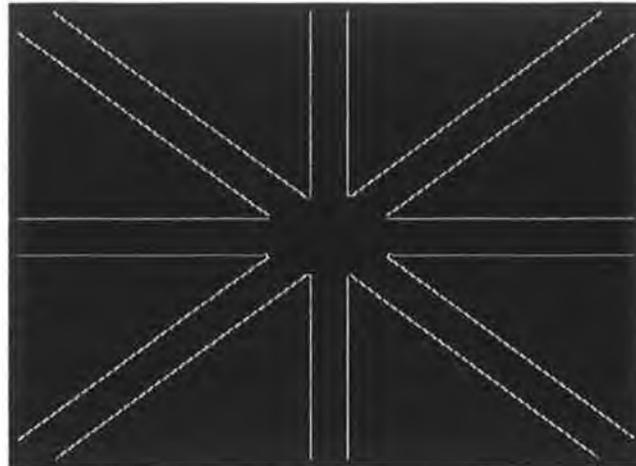


Fig 2.20 - Output of Canny Edge Detector

2.2.2 – Hough Line Detector

The Hough line detector [40] is used to find the pixels in the output of an edge detector that belong to straight lines and to determine the equations of these lines. Ideally an edge detector should produce an image highlighting only pixels that appear on edges and should represent all of the edge in the image. In practise, due to intensity discontinuities, the resulting pixels rarely characterise an edge completely. Causes of intensity discontinuities include noise and non-uniform illumination. The line detector therefore has the additional task of finding and linking line segments in an image. The Hough line detector determines the lines in the image with the largest number of pixels. One way to do this is to take any pixel and calculate which of all possible lines passing through the pixel contains the most number of other edge pixels in the image. This is repeated for all edge pixels in the image. The lines having the largest number of edge elements are the primary lines in the image. This process is computationally expensive.

The Hough Transform visualises the equation of a line in parameter space [41]. Consider a point (x_i, y_i) . There are infinitely many lines passing through this point which satisfy the slope intercept line equation (Equation 2.48) such that $y_i = mx_i + c$

$$y = mx + c \tag{2.48}$$

If the values of m and c are fixed then there is a line defined in the xy plane to which (x_i, y_i) must belong.

Equation 2.48 can be written as:

$$c = -mx + y \tag{2.49}$$

If now the values of x and y are fixed then there is a line defined in the mc plane (parameter space) to which (m_i, c_i) must belong.

On the left of Fig 2.21 there are two points in the xy plane. The corresponding lines in parameter space are shown on the right. The intersection of these two lines is the point (m', c') . These parameters correspond to the line in the xy plane that joins the two points (x_i, y_i) and (x_j, y_j) .

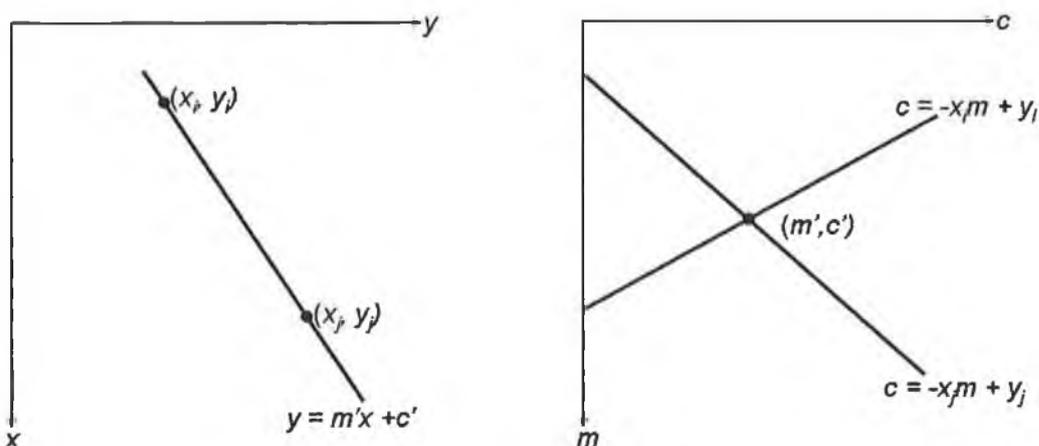


Fig 2.21 – Left: 2 points forming a line in xy plane. Right: Parameter Space (mc plane) lines corresponding to the 2 points

If the parameter space lines for all pixels in the edge image are plotted then lines in the images could be identified by a large number of parameter space lines intersecting at the same point in parameter space. It is unlikely though that this approach would succeed due to the fact that m (the slope of the line in xy plane) approaches infinity as the line approaches infinity. To avoid this complication the normal representation of a line is used (see Appendix A.2.2):

$$x \cos \theta + y \sin \theta = \rho \quad (2.50)$$

On the left of Fig 2.22 is seen that a horizontal line has $\theta = 0^\circ$ and a vertical line has $\theta = 90^\circ$. Each sinusoidal curve in the plot on the right of Fig 2.22 represents the range of different values for ρ and θ for the family of lines passing through a single point in xy space. The intersection point of the two curves (ρ', θ') represents the parameter values for the line joining the points (x_i, y_i) and (x_j, y_j) .

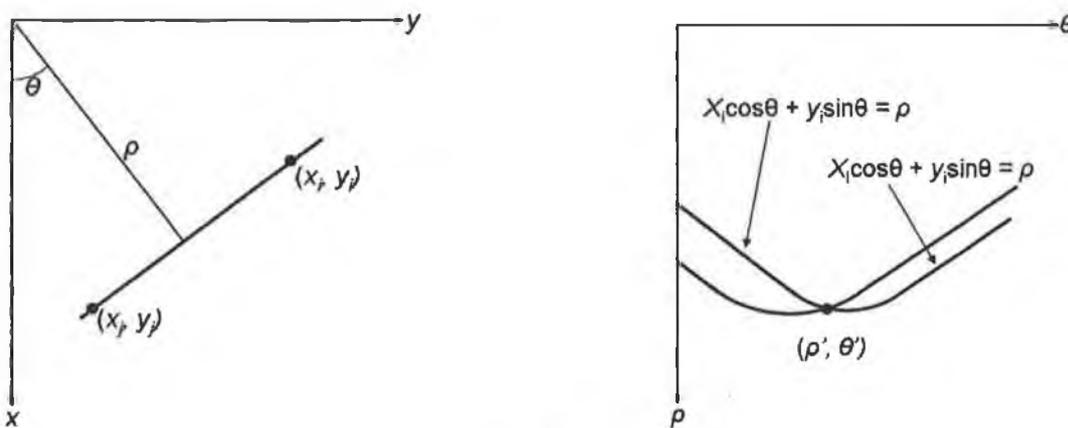


Fig 2.22 – Left: (ρ, θ) parameterization of lines in xy plane. Right: Sinusoidal curves in $\rho\theta$ plane

The Hough line detector divides the parameter space into accumulator cells as in Fig 2.23. Initially the value of each cell is set to zero. For each sinusoid, any cells that it passes through have their value incremented by one. Pixels in the edge image that form a line will have corresponding sinusoids that intersect at the same point in the parameter space. The accumulator cell with the largest value corresponds to the point in parameter

space where the most intersections occur. This point represents the parameter values of the strongest line match in the image.

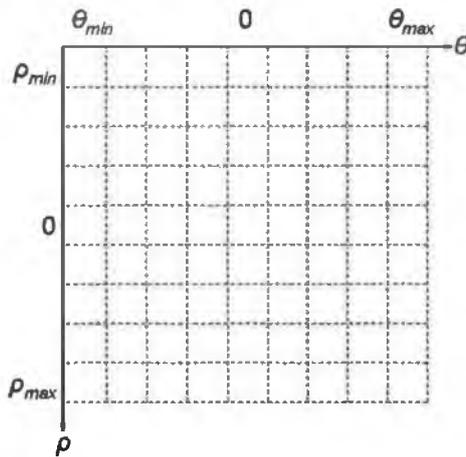


Fig 2.23 - Accumulator cells in the $\rho\theta$ plane

There are a discrete number of accumulator cells and the sinusoid must be rounded off to the nearest cell. The number of subdivisions in the $\rho\theta$ plane determines the accuracy of the line detector. Once the cells containing the highest values (the 'Peaks' of the Hough Transform) up to maximum number of peaks (the desired number of lines to be detected in the image) the pixels in the edge image that are elements of the detected line equations are marked. Groups of marked pixels along a line are marked as line segments, gaps in the pixels can be filled if desired. The endpoints of these line segments are the output of the Hough line detector.

Chapter 3 – IceRobotics System

IceRobotics is a Scottish based company “committed to developing leading-edge automation solutions to address the needs and priorities of the modern livestock farmer” [42]. The ‘IceTracker’, an IceRobotics product, is a machine vision sensor for real time tracking of cow teats. It is designed to acquire and track teats in real time, reporting the 3D position of the teat ends at a frequency of 10Hz with an accuracy of $\pm 5\text{mm}$. It is designed to perform regardless of warts, bumps, colour variations or hair on the teats.

3.1 - System Description

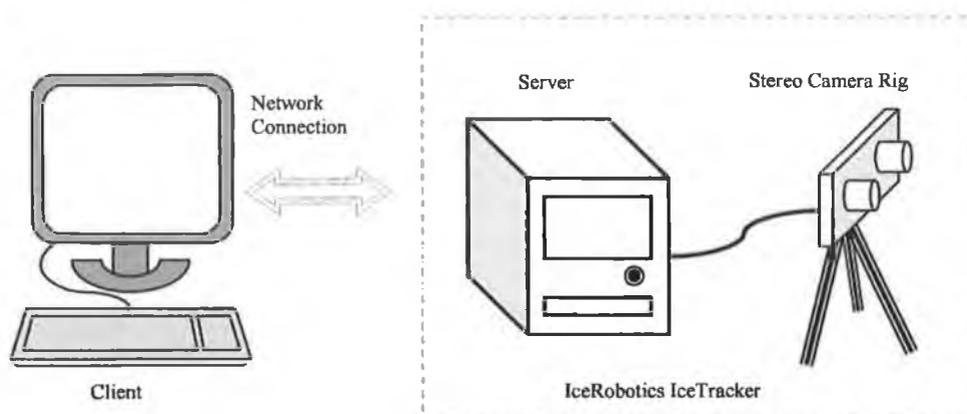


Fig 3.1 – Illustration of IceTracker system setup

The system operates using a client server interface to communicate with the user. Instructions and operating parameters are passed to the server from the client; the server returns images from the cameras as well as the position coordinates of the teats identified in the images. Fig 3.1 illustrates the topology of the system. The IceRobotics system is made up of the cameras and the server, a separate module with which an outside client communicates.

3.1.1 - Stereo Camera Rig

Two PCB mounted RGB colour CCD cameras make up the stereo camera rig. They each have lenses with focal length 3.5mm. The resolution of the cameras is 640x480 pixels. The cameras are mounted on two separate PCBs connected by a flexible ribbon cable. This allows the cameras to be moved freely relative to each other within the confines of the cable. Fig 3.2 shows the camera rig.



Fig 3.2 – Stereo Camera Rig

The PCBs have been mounted on a rig that allows the controlled orientation of the cameras relative to each other by reducing the freedom of movement to two axial rotations and planar translation. Locking screws allow the individual adjustment of the constraints. The degrees of freedom are illustrated in Fig 3.3.

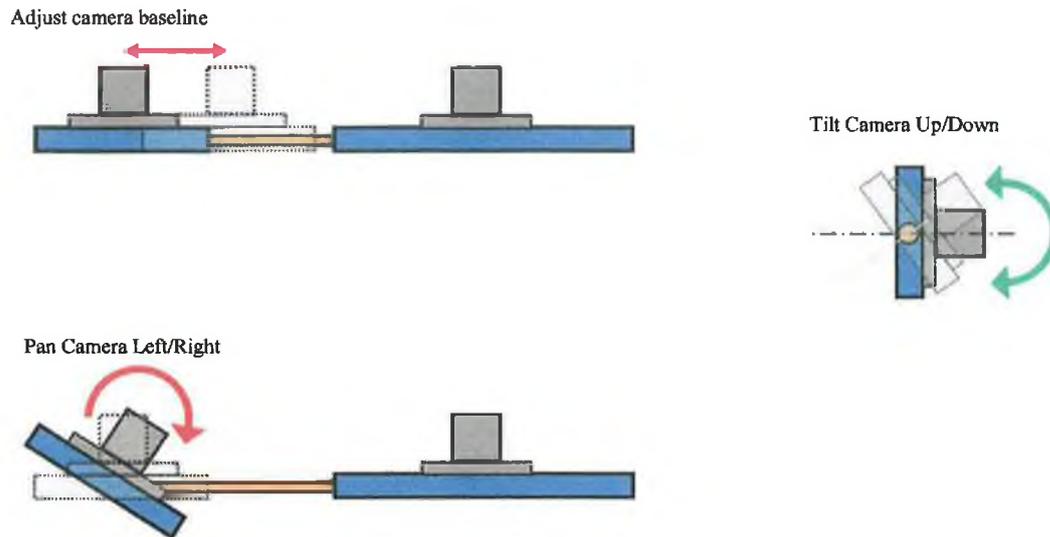


Fig 3.3 - Freedom of movement of left camera with respect to right camera

The movements of the cameras permitted by the rig allow the adjustment of the baseline and the crossover angle (these parameters are discussed in Section 3.2.3.c). Rotation about the Z axis, however, is not permitted. It is therefore not possible to adjust the cameras so that they are physically rectified (i.e. a horizontal line in the left camera image would appear as a horizontal line in the right camera image). The occurrence of this property therefore relies on the accurate construction of the rig. Unfortunately the cameras were not mounted on the rig in a rectified configuration and it is not possible to physically adjust for this. The cameras provide synchronized video stream: both cameras capture a single frame at the same instant; these frames are a synchronized stereo pair. It is essential that the images are synchronized for the extraction of 3D information from a moving target. Should the frames not be synchronised, any motion of a target in the time between the capturing of the first and second frames of a stereo pair will be interpreted as disparity in the images. This would destroy the relevance of any depth information extracted. The capture of images by both cameras is triggered by the same internal clock to ensure synchronization. The images are transmitted from the camera PCB to the server via an LVDS (low voltage differential signal) cable. This allows the server to

communicate to the cameras and control settings such as exposure times and auto exposure levels.

3.1.2 - IceRobotics Server

The server is a stand alone processing unit. It has the architecture of a PC but is designed to run without the inputs of a keyboard or mouse and without the output of a monitor. The server can be interfaced via a serial port and via a LAN (local area network) connection. A connection to the server is first established via the serial port. The server is then assigned an IP address so that it is recognised on the network and can be accessed by any computer on the same network. Operational commands can be given to the system via the serial connection which also allows test coordinates to be transmitted back to the client. The serial connection, however, cannot transmit video images from the cameras back to the client as the bandwidth required exceeds the transmission rates of the serial connection. In order to allow the reception of the camera images a LAN connection must be used. Fig 3.4 contains photographs from the front and from behind the server, it is seen that it very similar to a PC.



Fig 3.4 – View of server from front (left) and from behind (right)

3.1.2.a - Client

IceRobotics provide software for connecting to and communicating with the server using a PC running Windows XP Operating System. Sample programs are provided to allow basic operation such as configuring the server's network settings; starting and stopping the tracking of teats, displaying the video stream from the cameras, calibrating the cameras and displaying the detected 3D teat coordinates in real time. All these operations are carried out on board the server, the sample program running on the client makes function calls to the server and the appropriate action is taken. A C++ Software Development Kit (SDK) is provided; this allows separate C++ programs to be written incorporating function calls to the server. The SDK library provides functions that are not utilised in the sample program. These functions allow a user to write a program allowing the internal parameters of the server to be adjusted, giving greater control of the teat tracking process.

3.1.2.b - Sample Client Application

The 'IceTeatTraker API' is a sample program provided by IceRobotics which enables a client to connect to the server and control the teat tracking process. An executable (IceTeatTrakerClient.exe) file opens a GUI (Graphical User Interface) as pictured in Fig 3.5. This GUI allows the user to first establish a connection with the server via the serial (RS232) port. As previously mentioned, serial communications can be used to control the server and receive teat coordinates but video cannot be communicated back to the client. It is therefore advantageous to exclusively use the LAN connection for communicating with the server once it is available. The network settings of the server must be configured before the client will be able to recognise the server on the network. Configuration is first carried out over the direct serial port.

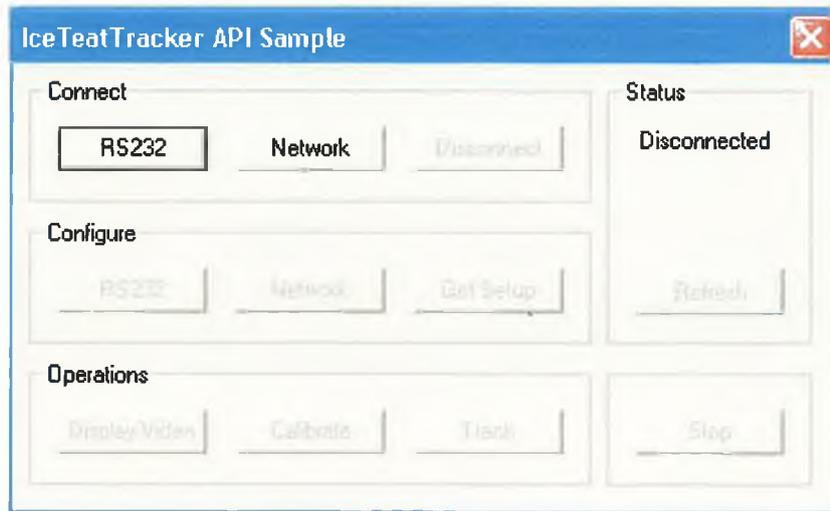


Fig 3.6 – Sample Application: IceTeatTracker API

The dialogue box shown in Fig 3.6 divides the applications tools into three categories: ‘Connect’, ‘Configure’ and ‘Operations’. ‘Connect’ and ‘Configure’ establish the communication channels and configure the network settings. Once configured there should be no further need for the ‘Configure’ tools: a LAN connection can be established using only the ‘Connect’ tools (provided the local network remains unchanged). The ‘Operations’ tools control the actions of the server once a connection is established. ‘Display Video’ generates two further windows each containing a live stream of video, one from the left camera, the other from the right camera. ‘Calibrate’ also displays the two cameras views with the calibration routine indicators overlaid. ‘Track’ tells the server to start searching for cow teats. It again displays the cameras views with markings overlaid anywhere a teat is detected. Another window is opened displaying the 3D coordinates of detected teats and representing them on a 3D plot. The ‘Stop’ button can be used to halt the current operation.

3.1.2.b.1 - Calibration

Before the Tracking operation can be executed the vision system must be calibrated using the ‘Calibrate’ operation. When the calibration procedure is completed a file containing the calibration parameters is written to the root folder of the client’s hard drive. This data

is passed to the server when the 'Tracking' operation is called. The 'Calibration' operation brings up two windows as shown in Fig 3.7.

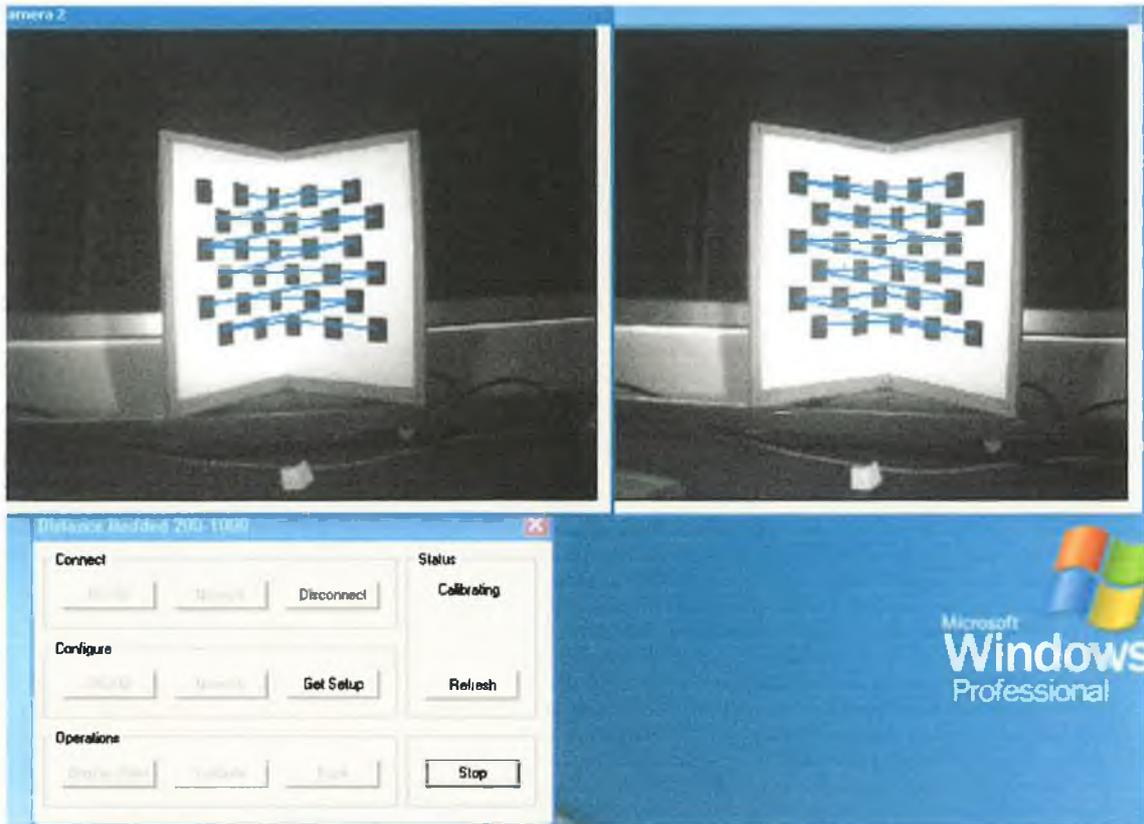


Fig 3.7 – Windows displayed during calibration operation

The calibration routine searches for the black squares of the pattern on the calibration board seen in Fig 3.7. Red dots are overlaid on the images indicating the centres of the detected squares. Blue lines are overlaid on the images, joining the square centres. When sufficiently many squares have been detected, in a single instant and in both images, then the calibration process is complete and the calibration data is written to the hard drive. The calibration board used by the IceRobotics system is shown in Fig 3.8.

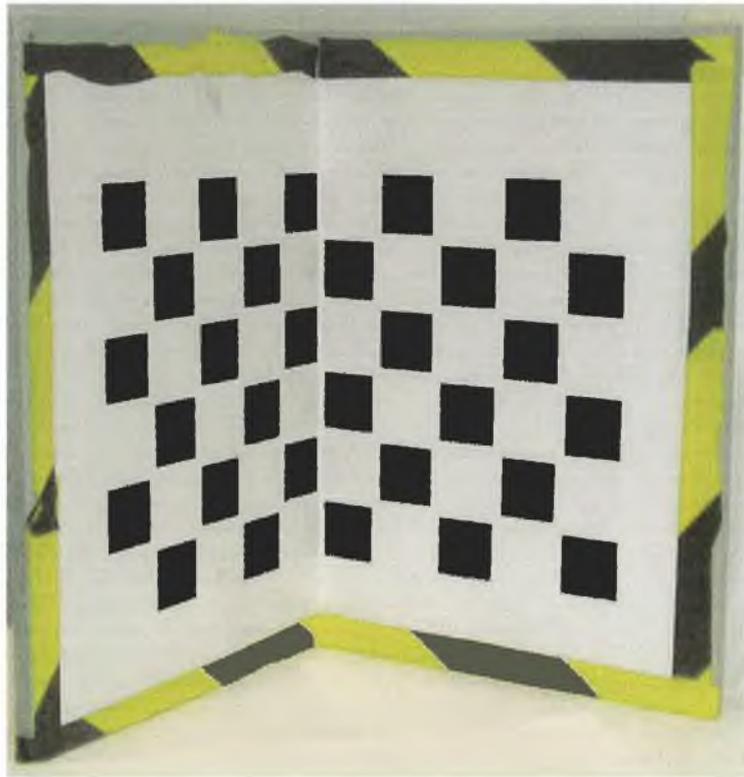


Fig 3.8 – The calibration board for IceRobotics System

3.1.2.b.2 - Tracking

The operation command 'Track' informs the system to begin the teat tracking process. The system identifies objects in the images that could possibly be cow teats due to shape, size and location. As will be discussed in the following sections the user can control the variables governing the criteria used by the system when searching for a teat. The system identifies possible teat matches in the stereo image pairs. Identified teat candidates from one image are matched with corresponding candidates from the other image. If the 2D pixel coordinates of the bottom of a teat are known for both the left and right images then the location of the teat in 3D is triangulated. The positions in the image of both the left and right camera of objects identified as teats are indicated with a red 'X' overlaid. This is seen in Fig 3.9.

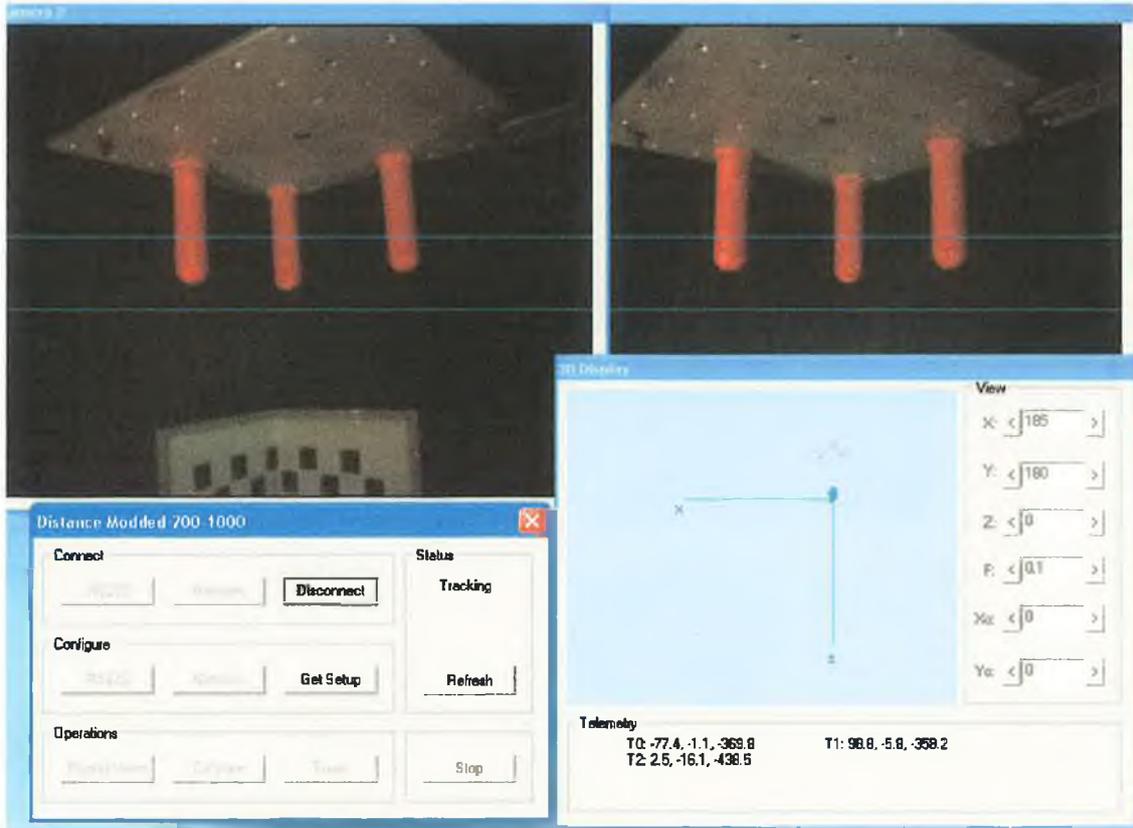


Fig 3.9 – Output display on client during tracking process

Fig 3.9 is a screenshot from the client during the tracking process: two windows display the video stream from the left and the right cameras. Red 'X's' are overlaid on objects in the image that have been identified as teats. The two horizontal lines in middle of both video windows represent the vertical search region. This is a user controlled parameter which specifies the region of the image in which the system should search for teat candidates. A third window, '3D DISPLAY', is also seen in Fig 3.9. This is a 3D plot of the identified teats in space. The view of this plot can be rotated and zoomed as required. In Fig 3.9 the view has been chosen so that the 3D plot is a plan view, i.e. the Y-axis is coming out of the screen (or page) and the XZ plane is coplanar with the screen (or page). The cameras are located at the origin pointing at the targets which are straight ahead (above the origin in the display). This is easily understood by comparing the video images to the 3D plot. The depth information of the three identified teats is clearly shown in the plan view of the 3D Plot.

Underneath the 3D plot display there is an output of the system 'Telemetry'. This is a real time output of coordinates corresponding to teats identified in the images.

3.1.2.c - Using the SDK

The software developer kit provided by IceRobotics primarily consists of useful functions grouped in classes. The classes include the `IceTeatTracker` class; the `IceUtilityImageHelper` class; the `IceUtilityViewpointTransform` class and the `IceErrorLog` class. These classes are used to develop an application capable of interfacing with the IceRobotics system and controlling it.

3.1.2.c.1 - The IceTeatTracker Class

The `IceTeatTracker` class encapsulates the main interface with the TeatTracker system. Table 3.1 contains a list of the public member functions of the `IceTeatTracker` Class; these are the functions that operate on the server and can be called by an application running on the client.

Function	Description
IceConfigureNw	Configure network connection
IceConfigureRs232	Configure serial connection
IceConfigureTracking	Configure tracking
IceConnectRs232	Connect to server through serial
IceConnectNw	Connect to server through network
IceDisconnect	Disconnect from server
IceDisplayVideo	Start displaying video
IceDownloadFirmware	Download firmware from server
IceGetNwSetup	Get current network configuration
IceGetRs232Setup	Get current serial configuration
IceGetState	Get current running mode
IcePeek	Look at system value
IcePoke	Set system value
IceStartCalibration	Start calibrating camera
IceStartTracking	Start tracking tests
IceStop	Stop current server mode
IceUploadFirmware	Upload firmware to server
OnCalibrateDone	Called when calibration complete
OnError	Called on error
OnPosition	Called to report a position
OnVideoFrame	Called to report a video frame

Table 3-1 – Member Functions of the IceTeatTracker class

As is seen from the description column of Table 3.1 the `IceTeatTracker` class contains the functions necessary for the basic operation the `TeatTracker`: connecting to the server, displaying video, calibration, tracking and configuring the tracking variables. The description given in the table is adequate for the majority of the functions. However, particular attention should be paid to the following functions: `IcePeek`, `IcePoke`, `OnCalibrateDone`, `OnError`, `OnPosition`, `OnVideoFrame`.

3.1.2.c.2 - Peek / Poke – Addressable Server Parameters

The `IcePeek` and `IcePoke` functions are used to access the internal system parameters which are not directly accessible by the other functions. Changing the values of these parameters will effect how the other member functions will perform. The `IcePeek` function

is used to inspect the current value of the parameter; the `IcePoke` function is used to set the parameter to a new value. Table 3.2 lists the addresses which can be accessed using the Peek / Poke functions and the parameters that they are related to. The Tracking variables control the performance of the TeatTracker during tracking: the maximum number of teats to identify, the region of the image in which to identify teats (vertical search window – as seen in Fig 3.9), and the minimum and maximum width of a detected teat. The resolution of the images communicated back to the client is controlled by the image height and width parameters. The tracking exposure parameter controls the shutter speed of the cameras, a longer exposure time will allow more light to fall on the CCD as each frame is grabbed but will also cause blurring in images if objects are moving. Short exposure times require the use of intense illumination. By setting the expose parameter to '-1' the IceTraker is put into auto exposure mode. The system will choose an appropriate exposure based on the 'Auto exposure target luminance while tracking' parameter. The system dynamically adjusts the exposure time to match the luminance of the target (the teats) to the parameter value. The Calibration and the Video Only addressable parameters operate in much the same way as the Tracking parameters.

Address	Description
Tracking	
0x00094262	Start of vertical search window (% from top of image)
0x0009426C	End of vertical search window (% from top of image)
0x0009421C	Maximum number of teat candidates to find (≤ 8)
0x000941EA	Camera orientation
0x000941CC	Maximum teat width in mm
0x000941D6	Minimum teat width in mm
0x0009419A	Height of output images
0x00094190	Width of output images
0x0009417C	Auto exposure target luminance while tracking
0x00094172	Tracking exposure (0 to 32000ns, -1 = auto exposure)
0x00094226	3D coordinate type (0 = World, 1 = camera)
Calibration	
0x00094118	Height of output images
0x0009410E	Width of output images
0x000940FA	Auto exposure target luminance while calibrating
0x000940F0	Calibration exposure (0 to 32000ns, -1 = auto exposure)
Video Only	
0x000940B4	Auto exposure target luminance for video only
0x000940AA	Video only exposure (0 to 32000ns, -1 = auto exposure)
0x00094096	Resolution (0 = QCIF, 1 = CIF, 2 = 4CIF, 3 = 2CIFV, 4 = 2CIFH)

Table 3-2 – Peek/Poke address parameters

3.1.2.c.3 - Virtual Member Functions

The use of virtual member functions allows the server to communicate to a client application. The virtual member functions are overrides of functions called by the server. This means that when the program running on the server calls a particular function it runs a virtual member function defined in the client application instead of the member function defined on the server. In this way a client application receives prompts at key instances in the operation, allowing the client application to dictate the course of action taken. Table 3.3 lists the virtual member functions in the `IceTeatTracker` class.

Function	Description
OnCalibrateDone	Calibration completed.
OnError	Called to report an error.
OnPosition	Called to report teat locations.
OnVideoFrame	Called to report a video frame

Table 3-3 – Virtual Functions in the IceTeatTracker class

The functions listed in Table 3.3 are all called internally on the occurrence of a particular event. The `OnCalibrateDone` function is called by the server upon the completion of a successful calibration. At this point it is necessary to write the calibration data to the hard drive of the client. The `OnCalibrateDone` function is redefined (overridden) with a virtual member function defined in the client application. When the event of a successful calibration takes place the program flow is diverted to the client application where the code contained in the virtual member function is executed before returning the flow to the server. The code contained inside the virtual member function can therefore dictate what is done on the occurrence of the event, in this case writing the calibration data to an appropriate file located on the client's hard drive. `OnError` is the member function called when an error is encountered. Overriding this with a virtual member function allows the client application to be aware of the error and to control the actions taken. `OnPosition` is called each time a target is identified. This function is overridden to allow the target coordinates to be sent to the client where they can be displayed or written to an output file. `OnVideoFrame` is the function called each time a video frame is received by the server from the cameras. This function is overridden to allow control of how the frames are displayed by the client application, or, if they are being stored, where they are stored on the client and in what file format.

3.1.2.c.4 - The IceUtilityImageHelper Class

This class provides functions to support the display and the storage of the images captured by the cameras. The functions are: `IceRenderBmp`, `IceSaveBmpAsPcx`, `IceSaveBmpAsPpm`. `IceRenderBmp` is used to display the video frames in a Win32 window. `IceSaveBmpAsPcx` is

used to store a frame as a PCX file in a specified location on the client. `IceSaveBmpAsPpm` is used to store a frame as a PPM file in a specified location on the client.

3.1.2.c.5 - The IceErrorLog Class

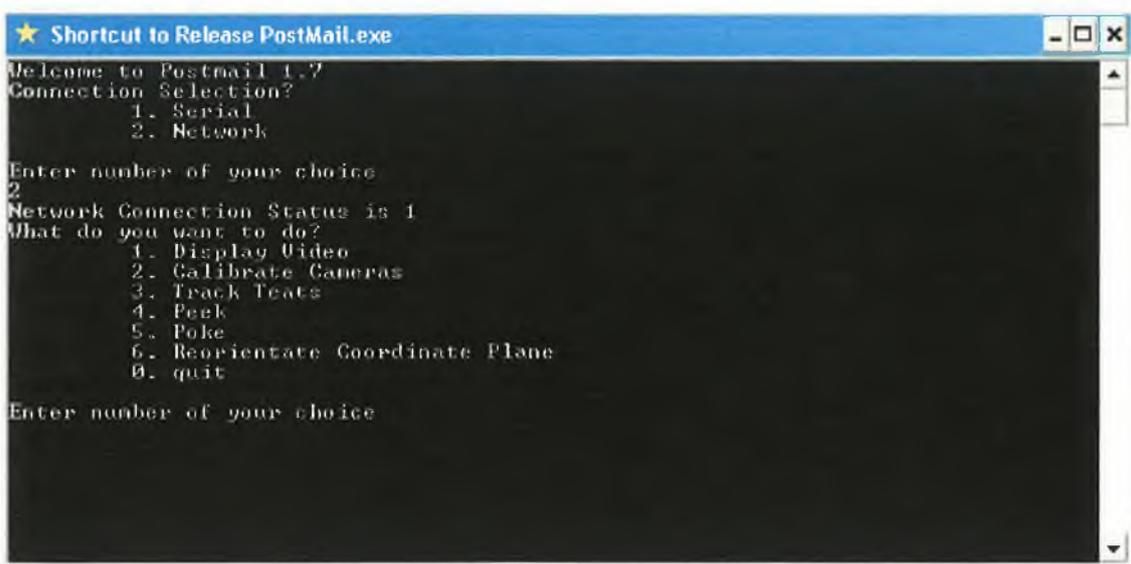
This class provides error logging for the client application. The member functions: `IceIsLogging` (checks error logging status), `IceSetLogFileName` (set the filename of the error log), `IceSetMaximumLogFileSize` (Maximum log file size in bytes, -1 for unlimited size), `IceSetMessageServer` (defines the ip address of the error message server), `IceStartLogging` (Start error logging), `IceStopLogging` (Stop error logging), `OnLogError` (override-able function – called when there is an error in the error logging process), are used to control the logging of errors on the clients hard drive in a specified location.

3.1.2.d - Developed Client Applications

Two client applications were developed for providing the necessary control of the IceTracker using the knowledge contained in the previous section. The first application is a command line interface program entitled 'PostMail'. Pictured in Fig 3.10 it is primarily used for its ability to easily enquire and alter the value of system parameters but it can also control operations such as tracking and calibration. The menu screens, as seen in Fig 3.10, are selected by entering the appropriate number in the command line. The program informs the user, via a text output whether or not a requested operation has being carried out successfully. For information regarding the source code for the 'PostMail' program refer to Appendix C. The second application is a GUI program entitled 'SinglePageClient'. Its architecture is based on the sample application provided by IceRobotics. It provides the additional functionality required to stream frames of video to disk, as well as take a snapshot of the current view. It also allows the specification of the minimum and maximum distances a detected teat can be located from the cameras-changing this parameter requires recompilation of the program.

3.1.2.d.1 - Single Page Client

The GUI of the Single Page Client is seen in Fig 3.11. The overall functionality of the program resembles closely that of the IceRobotics sample application. The differences (apart from the Peek/Poke display) are the actions taken in the event of video frame being received by the IceTraker. The OnVideoFrame function has been overridden as previously described so that whilst tracking, all the frames are steamed to the hard drive where they are stored as PCX image files. The 'FrameGrab' button provides the functionality to save the current frames being displayed in the camera windows to the hard drive. This is useful for taking stereo snap-shots during the 'Display Video' operation.

A screenshot of a Windows-style window titled "★ Shortcut to Release PostMail.exe". The window contains a black background with white text. The text reads: "Welcome to Postmail 1.7", "Connection Selection?", "1. Serial", "2. Network", "Enter number of your choice", "2", "Network Connection Status is 1", "What do you want to do?", "1. Display Video", "2. Calibrate Cameras", "3. Track Teats", "4. Peek", "5. Poke", "6. Reorientate Coordinate Plane", "0. quit", "Enter number of your choice". The window has standard Windows window controls (minimize, maximize, close) in the top right corner.

```
★ Shortcut to Release PostMail.exe
Welcome to Postmail 1.7
Connection Selection?
  1. Serial
  2. Network
Enter number of your choice
2
Network Connection Status is 1
What do you want to do?
  1. Display Video
  2. Calibrate Cameras
  3. Track Teats
  4. Peek
  5. Poke
  6. Reorientate Coordinate Plane
  0. quit
Enter number of your choice
```

Fig 3.10 – View of output of PostMail program upon start-up

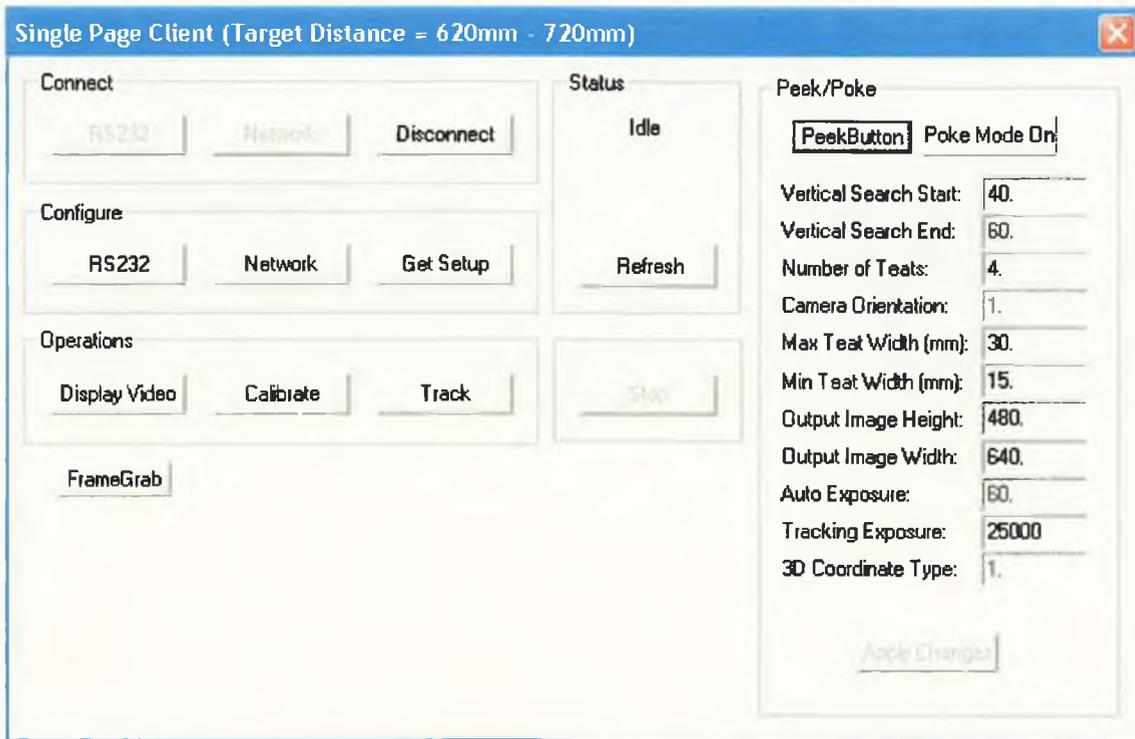


Fig 3.11 – Single Page Client Application window

3.2 - System Assessment

The assessment of the IceRobotics TeatTracker system is divided into two categories:

- 1) The success in identifying teats in the images.
- 2) The accuracy of the system in locating identified targets

3.2.1 - Identification of Teats

This section describes the performance of the IceRobotics in identifying target teats. This assessment is without regard to the accuracy of the position estimates. To assess system performance the following conditions were considered:

- Identification of un-obscured teats
- Identification of partially obscured teats
- Identification of teats when they are partially occluded from one of the cameras viewpoints

- Identification of teats when there are teat shaped objects, not in the target region, but in the cameras view
- Identification of teats when there are non-teat shaped objects in the target region
- Identification of closely grouped teats
- Identification of teats orientated at an angle
- Identification of moving teats
- Identification of teats with background movement

There are terms that arise in the assessment which must be specified. 'Target Teats' refers to objects in the scene, within the cameras' viewpoint, that represent cow teats. These are intentional targets for the vision system to detect. The 'Target Region' is the search volume to which the system is restricted. This is dictated by maximum and minimum distances a target can be from the camera (the variables are hard-coded and changing them requires recompiling the client application) and the vertical search region in the images (controlled using the vertical search region parameters- which can be altered using the Peek/Poke functionality). A target is 'Partially Obscured' when the view of the target from the camera is obstructed in such a way that the entire teat is not visible but the same portions of the target are visible to both of the cameras in the stereo rig. A target is 'Partially Occluded' when one of the cameras has a view of the complete teat but the other camera does not. 'Teat Shaped Objects' are objects not intended as targets for the system to detect but are of teat proportions in the images.

Testing has shown the IceRobotics system to be capable of fulfilling all of these criteria, but only in controlled conditions. The system's performance is not so robust that the criteria are readily fulfilled; the system requires constant adjustment and tweaking. Table 3.2 lists the internal system parameters that can be changed using the Peek/Poke functionality. Among these is the parameter controlling the maximum number of teats that the system should detect, the value can be a whole number in the range of zero to eight. If set to eight then the system will output up to eight targets from the scene considered to be possible teat candidates. This means that if there were ten teat shaped objects in the target region then the system would output up to the eight most likely teats.

Before an object in the scene is considered to be a teat it must pass certain identification tests contained within the architecture of the IceRobotics Vision System.

3.2.2 - Teat Identification in Practice

The teat identification performance of the IceRobotics system, in the scenarios specified in Section 3.2.2, is documented in this section. Tests have been documented as examples of both the failings and the successes of the system in each case. It is necessary to note that in achieving each example of the system operating correctly, the careful adjustment of the system set up was required. This adjustment includes small changes in the positions of the targets relative to the camera and variation of the lighting conditions.

3.2.2.a - Criteria 1: Target Teats in Target Region

Fig 3.12 shows the identification results for four un-obscured teats. The dummy teats have been arranged to represent the standard formation of a cow's udder, the two rear teats closer together than the two front teats. This setup simulates the camera rig being positioned underneath the cow and looking backwards towards the udder. The 3D display is a plan view of the detected teats. The scene in Fig 3.12 is an ideal scenario and may be regarded as a benchmark against which further trials may be compared. The targets are well lit and the system is successfully identifying them. The output is stable (all four targets are continuously detected) and the system has successfully detected the spatial relationships of the targets (as seen in the 3D Display plan view).

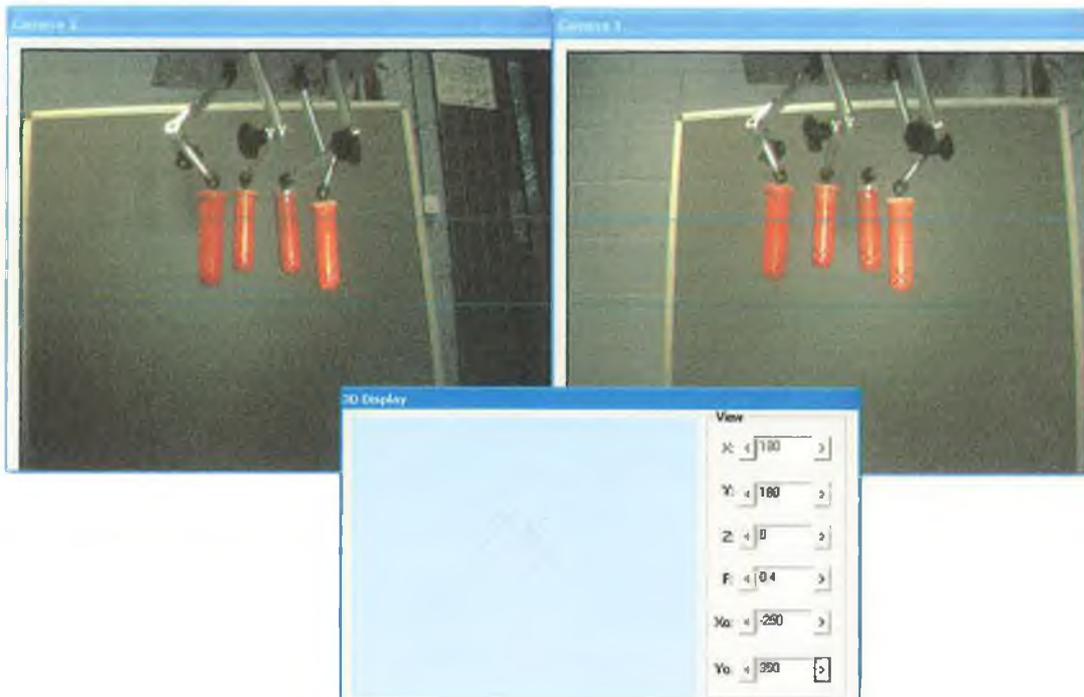


Fig 3.12 – Four target teats in standard udder formation – successful identification

3.2.2.b - Criteria 2: Partially Obscured Teats in Target Region

Fig 3.13 and Fig 3.14 show partially obstructed views of the cameras from the top and from the bottom respectively. When the cameras have a view of the ends of the teats the system is still able to detect them, as seen in Fig 3.13. If the teat ends are not visible, identification is not possible as shown in Fig 3.14.

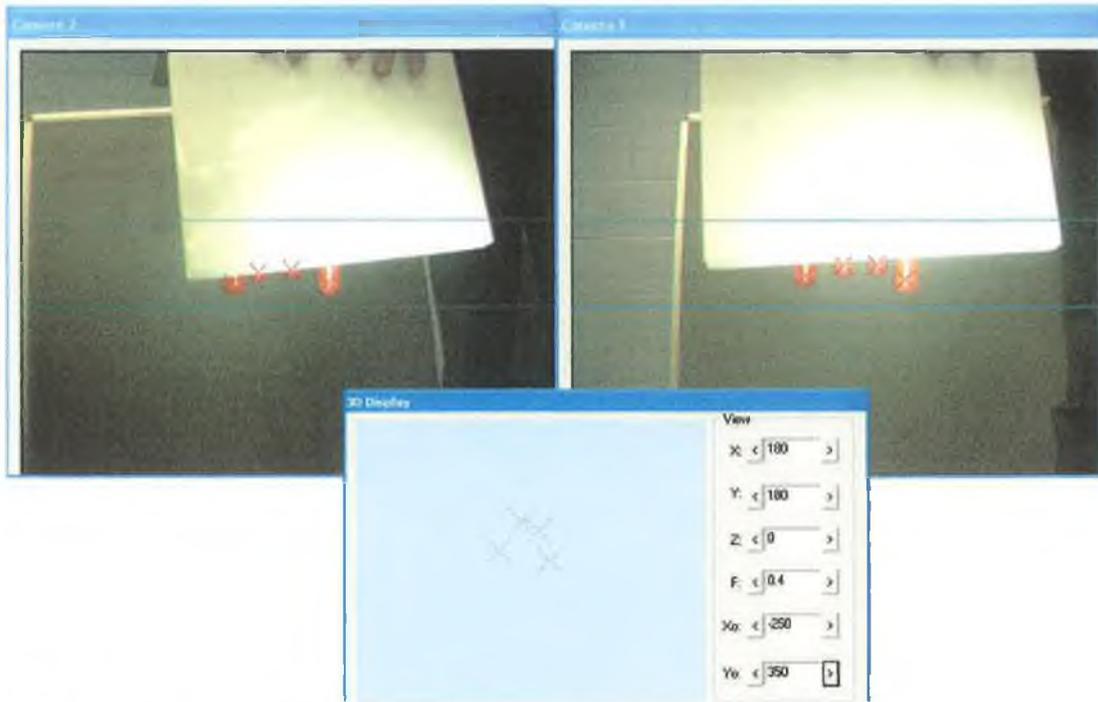


Fig 3.13 – Partially obscured from top, successful detection

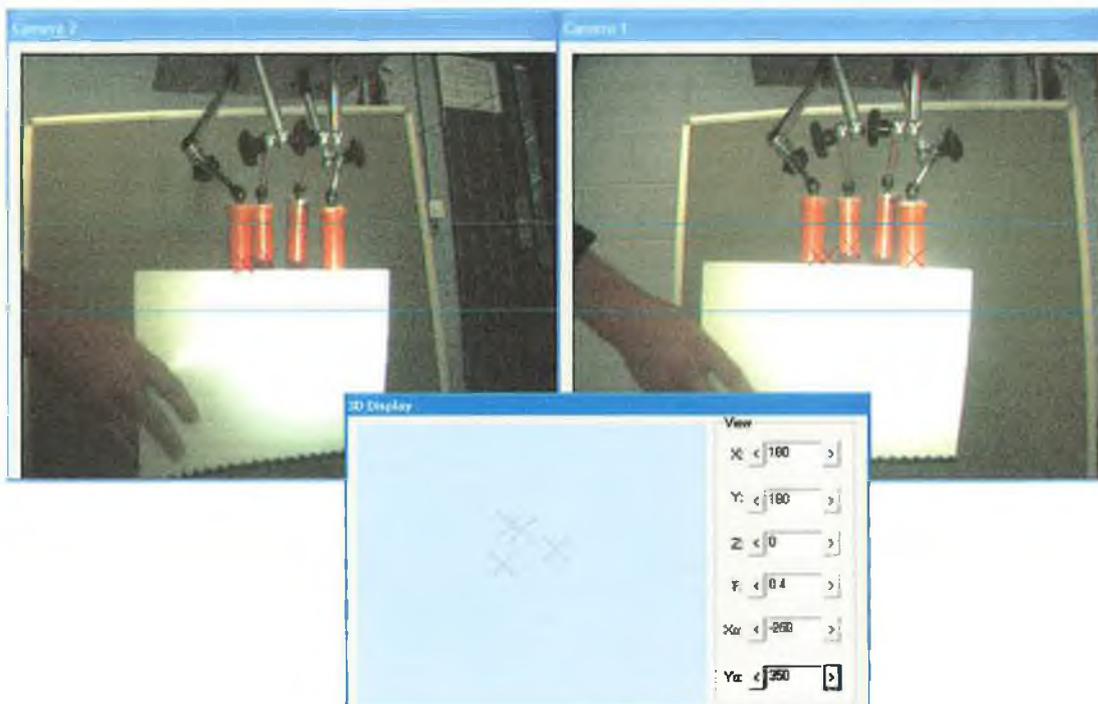


Fig 3.14 – Partially obscured from bottom – unsuccessful detection

3.2.2.c - Criteria 3: Partial Occlusion of Target Teats in Target Region

Identification of partially occluded teats is shown in Fig 3.15. The rear right teat (back left as seen in image) is partially occluded in the left camera image by the front right teat. The system can still identify the teat using the teat information from the non-occluded right camera view and the partial view of the teat available in the left image. If enough of the bottom of the teat can be seen by the left camera then a successful identification can be made.

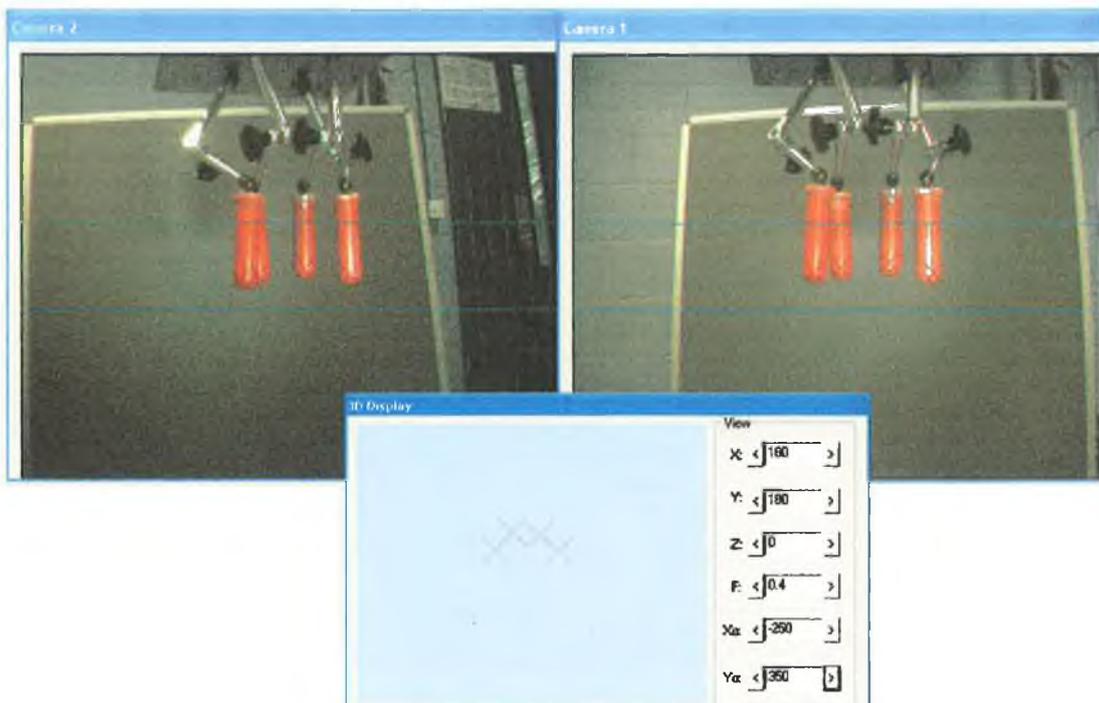


Fig 3.15 - Partially occluded teat in left image - successful detection

Fig 3.16 is another example of the system successfully identifying occluded teats, this time the teat is occluded in both of the camera views. The bottom of the teat can still be seen in both the left and right images.

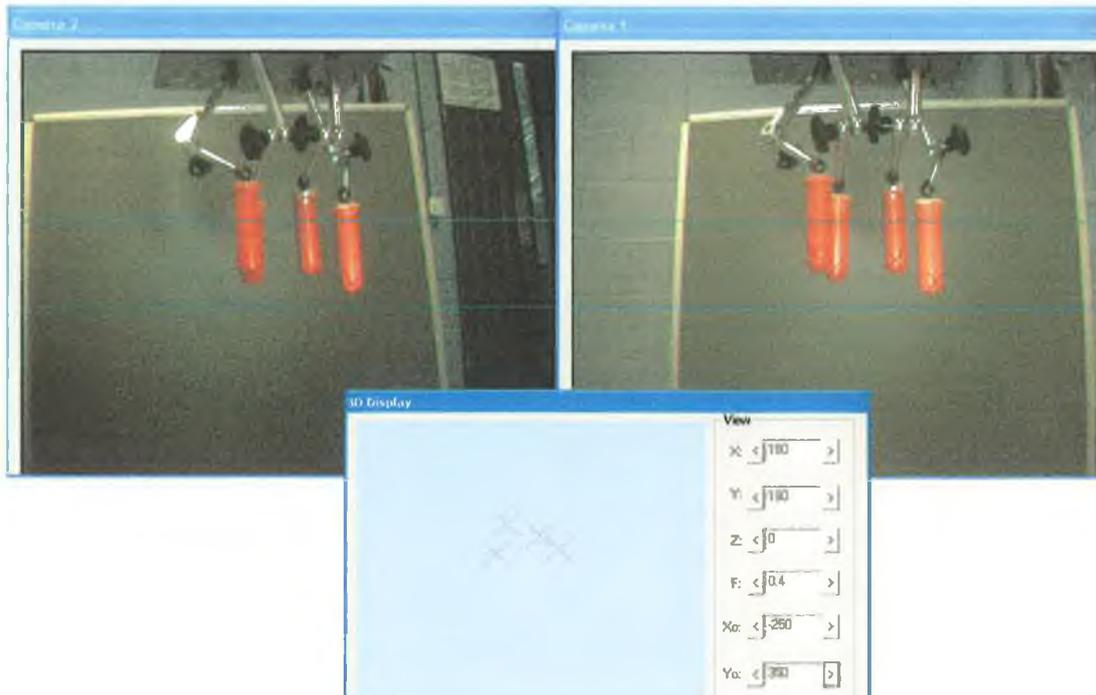


Fig 3.16 – Partially occluded teat in both the left and right camera views

3.2.2.d - Criteria 4: Teat Shaped Objects outside of the Target Region

A teat shaped object is any object that resembles the basic shape of a cow teat. For an object in the background to resemble a cow teat from the camera's point of view it needs to be bigger than an actual teat due to perspective. Fig 3.17 and Fig 3.18 contain images of the scene in which a milking cup is suspended upside down in the background. The overall geometry of the milking cup in the images resembles that of a teat. It is the abrupt end of the cup and its long proportions that create this effect. In both of these examples the milking cup is successfully ignored and all four of the target teats are successfully detected.

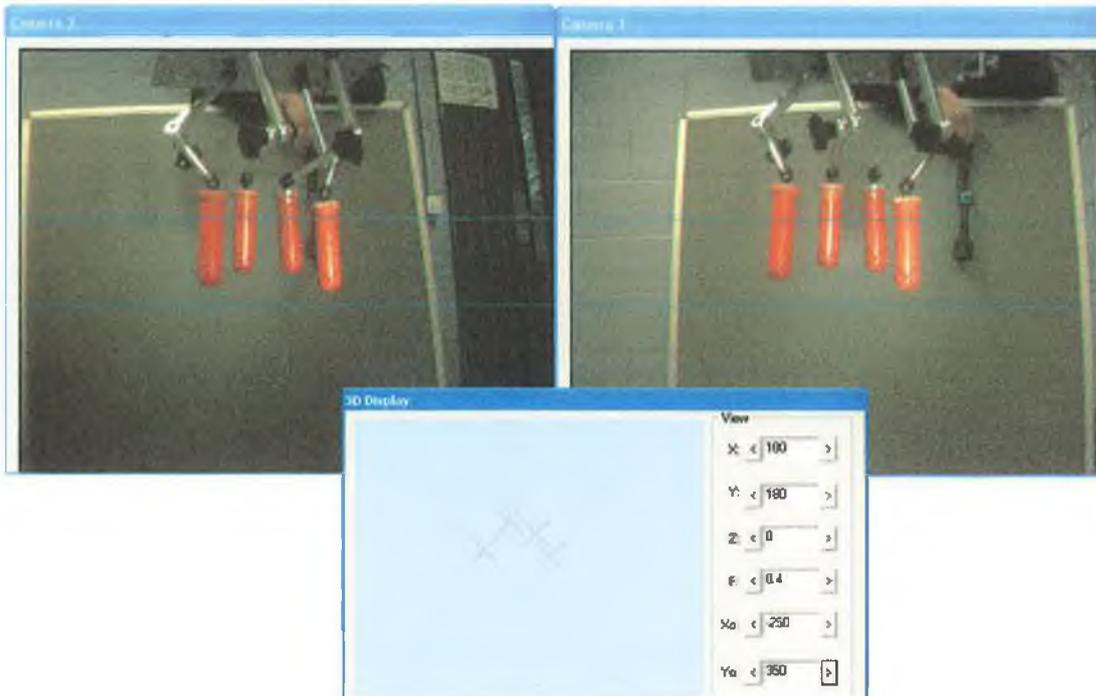


Fig 3.17 – Teat shaped object outside of target region – successfully ignored

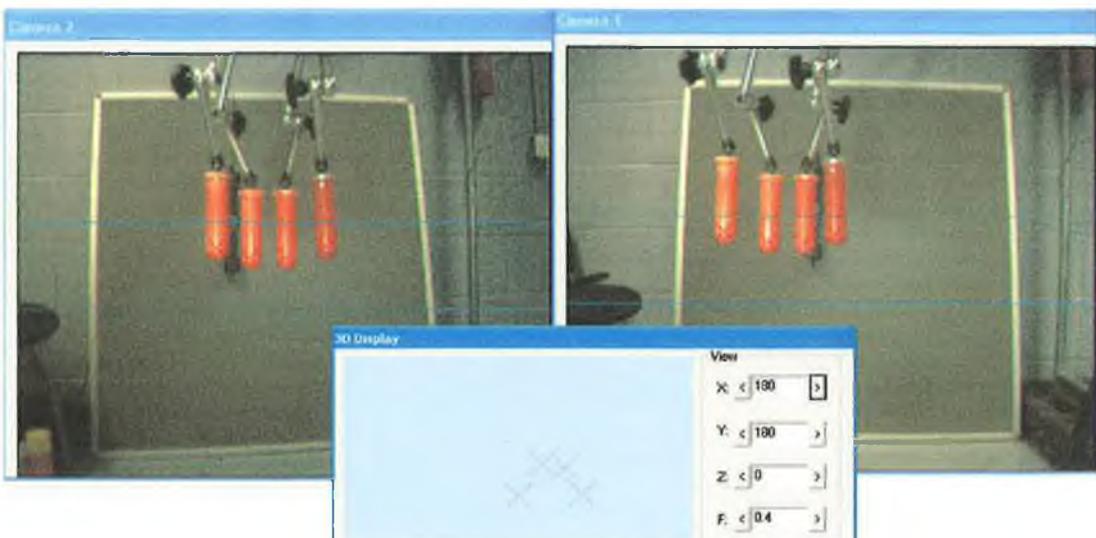


Fig 3.18 – Teat shaped object outside of target region –successfully ignored

Fig 3.19, Fig 3.20 and Fig 3.21 contain example of where a teat shaped object outside of the target region is falsely identified as a teat. In Fig 3.19 there is a single milking cup suspended upside down in the background, behind the dummy udder and outside of the

target region. A false correspondence is made between the rightmost teat in the left image and the left edge of the milking cup in the right image.

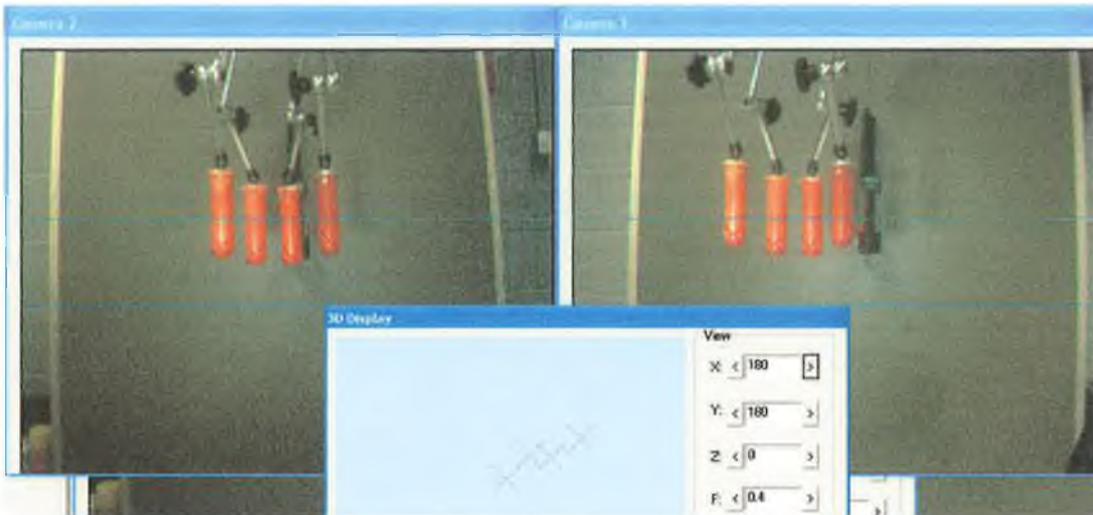


Fig 3.19 – Single teat shaped object outside of target region – false correspondence with rightmost teat

A correct correspondence for an object outside of the target region would result in a disparity that correlated to a target distance from the cameras that is not within the range of the target region. In the case of these examples, with the teat shaped object in the background, the disparity would be too small, correlating to a distance from the camera rig that is larger than that permitted for a target teat. In order for an object in the background to cause a false identification the system must find a false correspondence between the object in question in one image of the stereo pair, with a different object in the other image. In Fig 3.19 the correspondence is made between the milking cup and a teat. In Fig 3.20 there are now two milking cups in the image. A false correspondence is found between them such that the resulting disparity correlates to a target depth within the target region.



Fig 3.20 – Two teat shaped objects outside of target region – false correspondence between objects

In Fig 3.20 a false correspondence is made between the milking cups in the left image with the edge of a brick of the wall in the background in the left image.

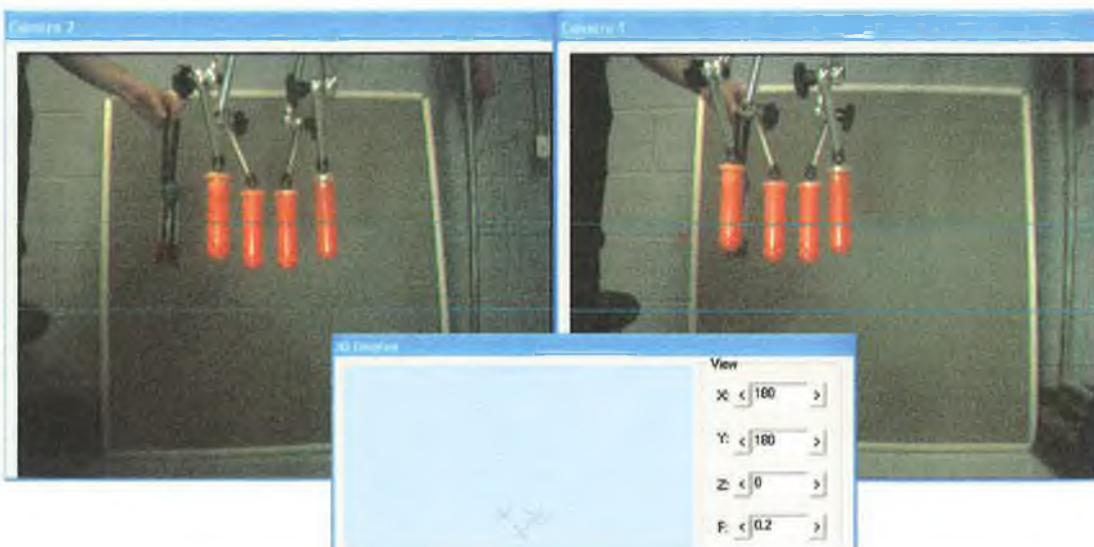


Fig 3.21 – Single teat shaped object outside target region, false correspondence with wall in background

Careful placement of the milking cups in the scene was required to produce the results seen in this section. Overall the system performs well in ignoring teat shaped objects in the target region, but as can be seen there are numerous situations that can arise where the system is confused by the background scene.

3.2.2.e - Criteria 5: Non Teat Shaped Objects in the Target Region

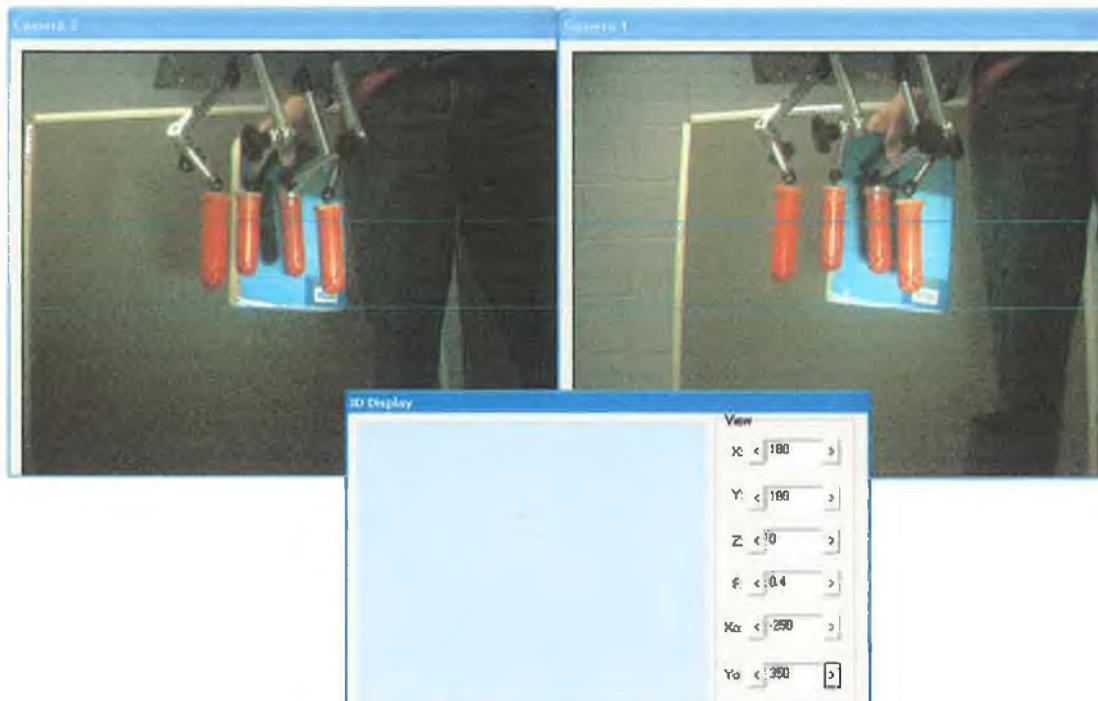


Fig 3.22 – Non teat shaped object in target region, successfully ignored

Fig 3.22 shows a book being held behind the teats but in the target region. As can be seen it is not mistakenly identified as a teat and all the actual teats are identified.

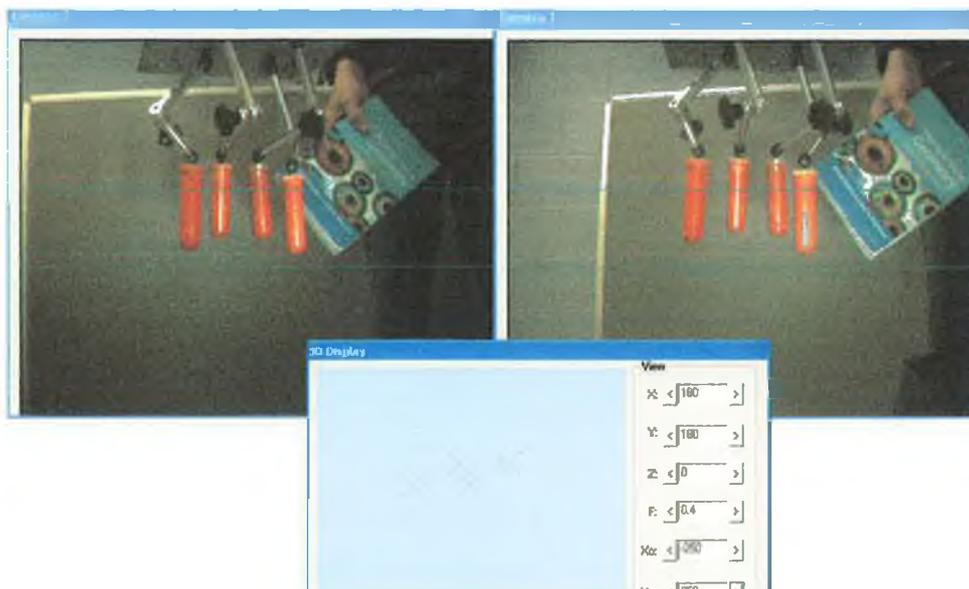


Fig 3.23 - Non teat shaped object in target region, incorrectly identified as a teat

In Fig 3.23 the system is seen to identify the cover of the book as a teat in favour of the two unidentified teats. This is a perfect example of the shortcomings of the system. From these images there is no clear reason as to why the system would consider the pattern on the book cover more likely to be a teat than actual teats objects clearly defined and illuminated within the target region. (Note: there are 4 'X's' visible in the 3D display window but only 3 marked targets in the video stream. The reason for this is that the displays do not refresh in perfect synch and the displays were in different states when the screenshot was taken.)

3.2.2.f - Criteria 6: Teats Grouped Closely Together

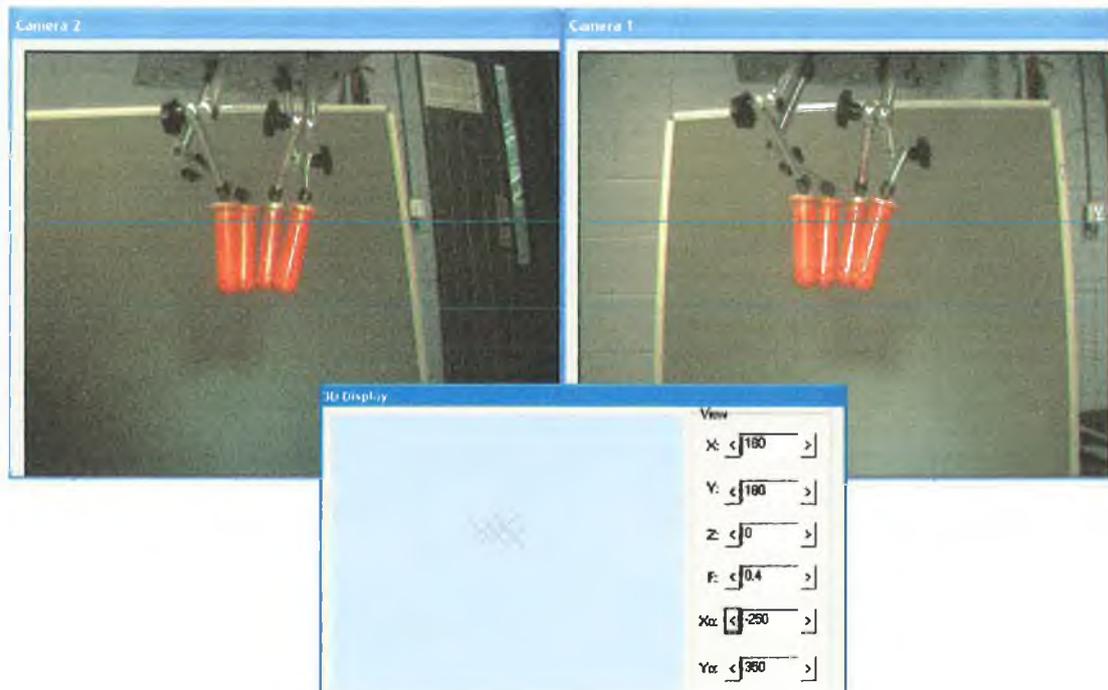


Fig 3.24 – Teats in target region bunched closely together – successful detection

The IceTracker is able to identify the teats even when they are bunched closely together as shown in Fig 3.24. Again it can be seen that the tracker performs well in dealing with occlusions.

3.2.2.g - Criteria 7: Teats Angled within the Target Region

Fig 3.25 illustrates the ability to successfully identify teats that are orientated at different angles and located within the target region. From left to right the teats are approximately orientated with respect the camera views as follows: 45° to the left in the images, 45° toward the cameras, 45° away from the cameras and 45° to the right in the images. The system performs well in identifying the angulated teats.

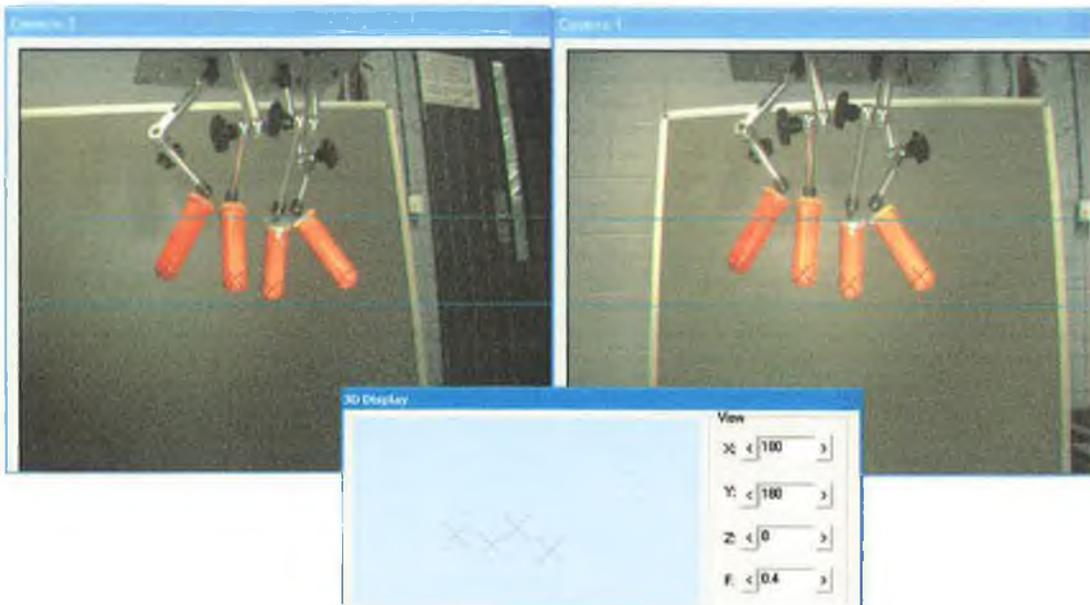


Fig 3.25 - Teats in target region orientated at different angles

3.2.2.h - Criteria 8: Moving Targets

The IceRobotics system operates at a frequency of 10Hz, meaning that ten stereo pairs of images are captured every second. The system is good at tracking a teat moving in the target region provided that in each captured frame the teats are in a position with suitable illumination that they would be identified should the scene be static. As can be seen in the following section small variation in either the position or the illumination can have detrimental effects on the identification process. However, it has been concluded that should these effects be eliminated the real time tracking of teats would perform well. A short exposure time is required to prevent blurring in the images of targets that are

moving quickly. This in turn requires bright lighting as the short exposure time of each image allows little time for light reflected from the targets to be incident on the CCDs.

3.2.2.i - Criteria 9: Background Scene is moving

The IceRobotics system is prone to making false identifications when the background scene is moving. In Fig 3.21 it is seen that an object in the background can be misinterpreted as being a teat. If there is regular movement of multiple objects in the background scene then it is likely that at some point the objects will interact in such a way that they appear to be a teat from the systems point of view. Improvements to the system are necessary before it could be incorporated in an AMS. The fact that the robot which applies the milking cups may be moving in the background scene means that the system needs a means of successfully ignoring such objects.

3.2.2.j - Instabilities: Minor changes in position

Even when all the conditions are favourable (lighting, static plain background) and setup parameters are appropriate (targets in focus, targets are in centre of target region, system calibrated) minor changes in the positions or in the orientation of a target that is successfully identified can lead to the system failing to identify it as seen in Fig 3.26 and Fig 3.27.

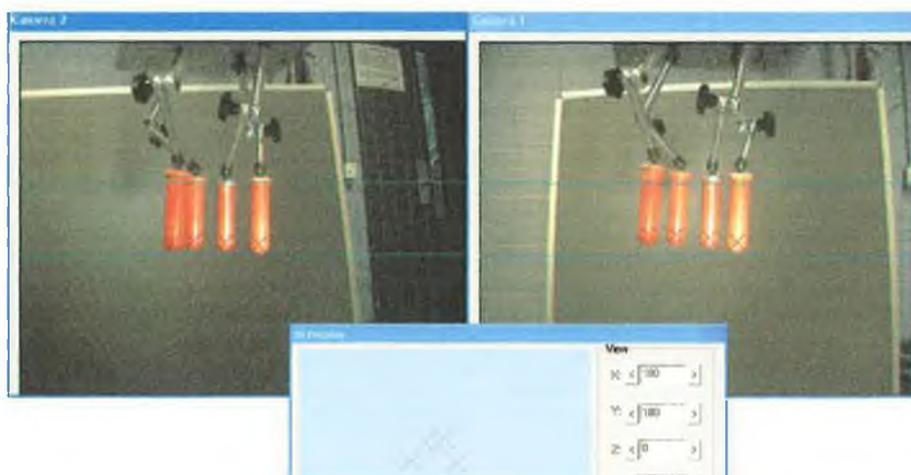


Fig 3.26 – Favourable conditions, all teats identified

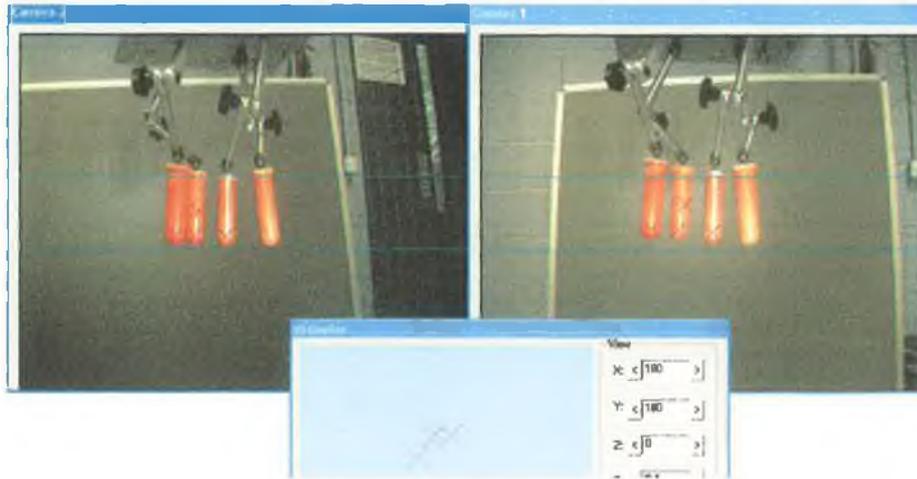


Fig 3.27 – Slight Change in position, conditions favourable, unsuccessful detection

It is seen in Fig 3.26 and Fig 3.27 that a slight change in the position of the right most teat in the image leads to the system failing to identify it. There are still four targets identified in Fig 3.26 but now there is one half-way up the teat, second from the left. The fact that the system is giving preference to this non-target over a teat within the target region is a serious failing.

3.2.2.k - Instabilities: Minor Changes in Illumination

Minor changes in the illumination of the targets can seriously affect the identification process. A minor change is considered as a change in the position of the light source and the angle of incidence of the emitted light on the targets without dramatically reducing the definition or intensity of the targets within the images. Fig 3.28, Fig 3.29 and Fig 3.30 are snapshots between which the light source has been moved. In Fig 3.28 it is to the right of the camera rig, in Fig 3.29 it is to the left of the camera rig and in Fig 3.30 it is above the camera rig. The targets are successfully identified in both Fig 3.28 and Fig 3.29, however if attention is paid to the telemetry output it is seen that the system considers the targets to be in a different location. This is an issue of accuracy and is not dealt with in this section, but it does indicate influence of lighting on the identification process. This is seen in Fig 3.30, the light source is not positioned above the camera rig, and the system

fails to identify all four teats and the leftmost teat is now detected as being in a completely different location.



Fig 3.28 – Light source to right of camera rig

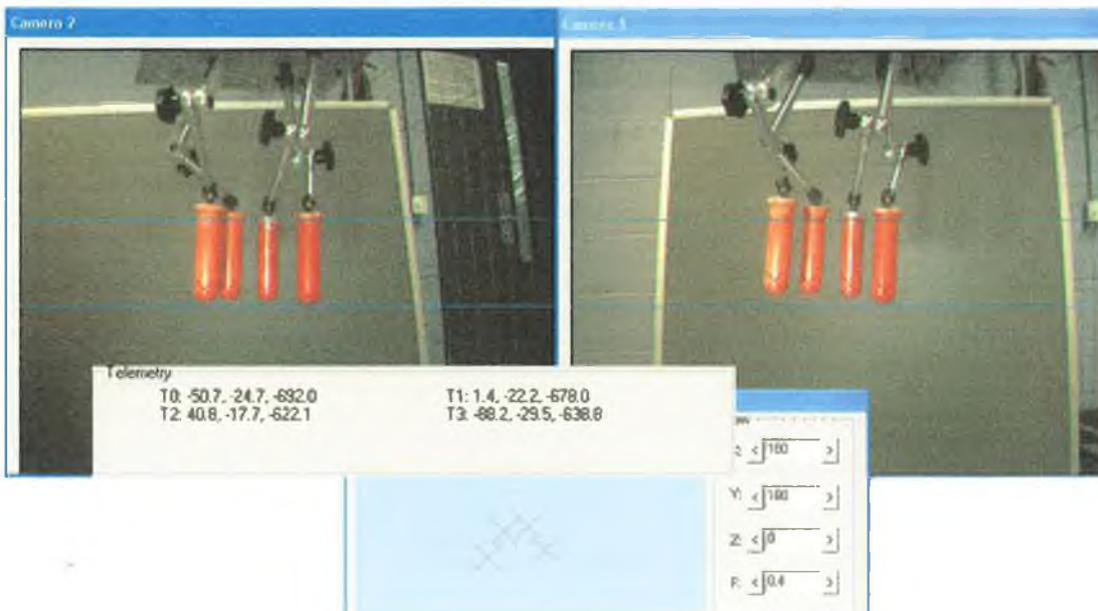


Fig 3.29 – Light source to left of camera rig

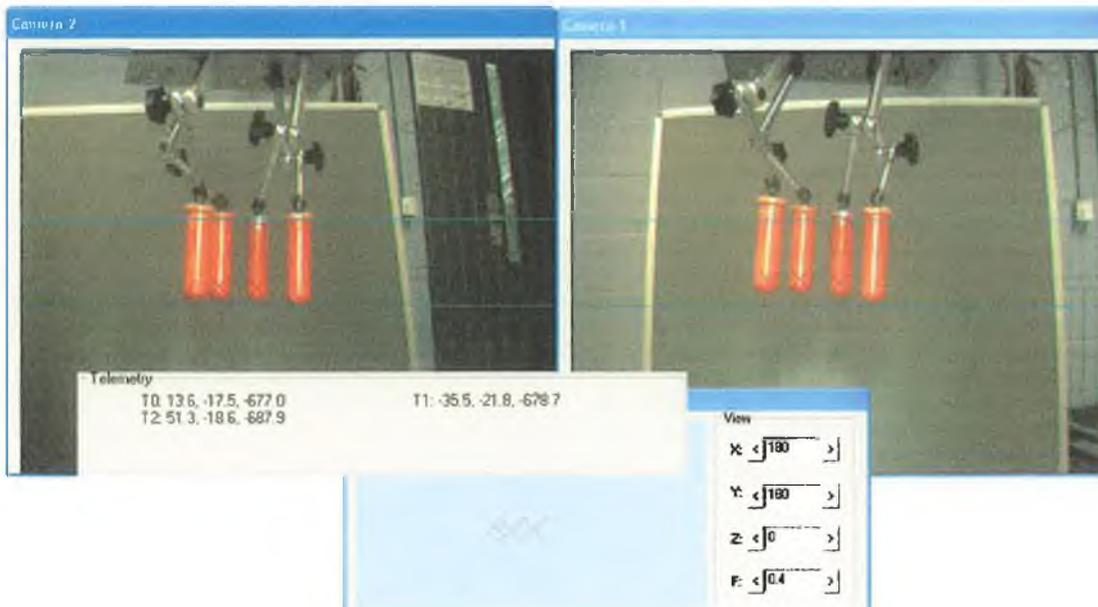


Fig 3.30 – Light source above camera rig

3.2.2.1 - Instabilities: Switching of identified points in stable conditions

Even in constant, stable conditions (constant static illumination source, static background scene, and static targets) the IceRobotics system output fluctuates. The output of where the system considers a teat to be located in 3D space switches between different values. Table 3.4 is a sample output of the system whilst tracking a single teat in stable conditions. Half way through the sample data shown, there is a large switch in the output. The Z component of the 3D coordinate varies by up to 30mm. In general for the three coordinate components, x, y and z, there are also slight changes from each location output to the next, but usually less than a millimetre in any axis.

-6.223817,-5.090349,-642.570154	-6.223895,-5.091552,-642.571245	-6.225998,-4.984682,-643.146384
-6.223891,-5.091489,-642.571189	-6.223894,-5.091536,-642.571231	-6.225998,-4.984682,-643.146384
-6.223920,-5.091945,-642.571603	-6.223894,-5.091536,-642.571231	-6.224052,-5.037160,-642.796209
-6.223920,-5.091945,-642.571603	-6.151099,-6.845633,-630.291507	-6.223269,-5.071921,-642.601766
-6.223972,-5.092739,-642.572324	-6.128918,-7.380098,-626.549927	-6.223269,-5.071921,-642.601766
-6.223972,-5.092739,-642.572324	-6.103953,-7.981664,-622.338558	-6.253151,-4.428567,-642.780498
-6.223923,-5.091986,-642.571640	-6.103953,-7.981664,-622.338558	-6.253151,-4.428567,-642.780498
-6.223911,-5.091797,-642.571468	-6.078987,-8.583229,-618.127190	-6.239861,-4.732781,-642.646450
-6.223911,-5.091797,-642.571468	-6.054021,-9.184795,-613.915822	-6.230388,-4.947786,-642.574584
-6.223903,-5.091677,-642.571359	-6.237609,-5.858776,-640.580677	-6.230388,-4.947786,-642.574584
-6.223903,-5.091677,-642.571359	-6.238534,-5.217858,-643.162297	-6.224980,-5.069530,-642.546230
-6.223903,-5.091677,-642.571359	-6.238534,-5.217858,-643.162297	-6.222616,-5.122114,-642.542312
-6.223898,-5.091602,-642.571291	-6.236217,-4.952736,-644.029967	-6.239090,-4.803957,-642.696676
-6.223896,-5.091565,-642.571257	-6.244994,-4.562622,-646.210918	-6.239090,-4.803957,-642.696676
-6.223896,-5.091565,-642.571257	-6.244994,-4.562622,-646.210918	-6.244517,-4.705846,-642.749227
-6.223895,-5.091552,-642.571245	-6.229988,-4.904155,-643.758197	-6.229423,-4.988900,-642.620177
-6.223895,-5.091552,-642.571245	-6.229988,-4.904155,-643.758197	-6.225094,-5.070266,-642.583472
-6.223895,-5.091552,-642.571245	-6.225998,-4.984682,-643.146384	-6.225094,-5.070266,-642.583472

Table 3-4 – Sample output of detected teat location in stable conditions

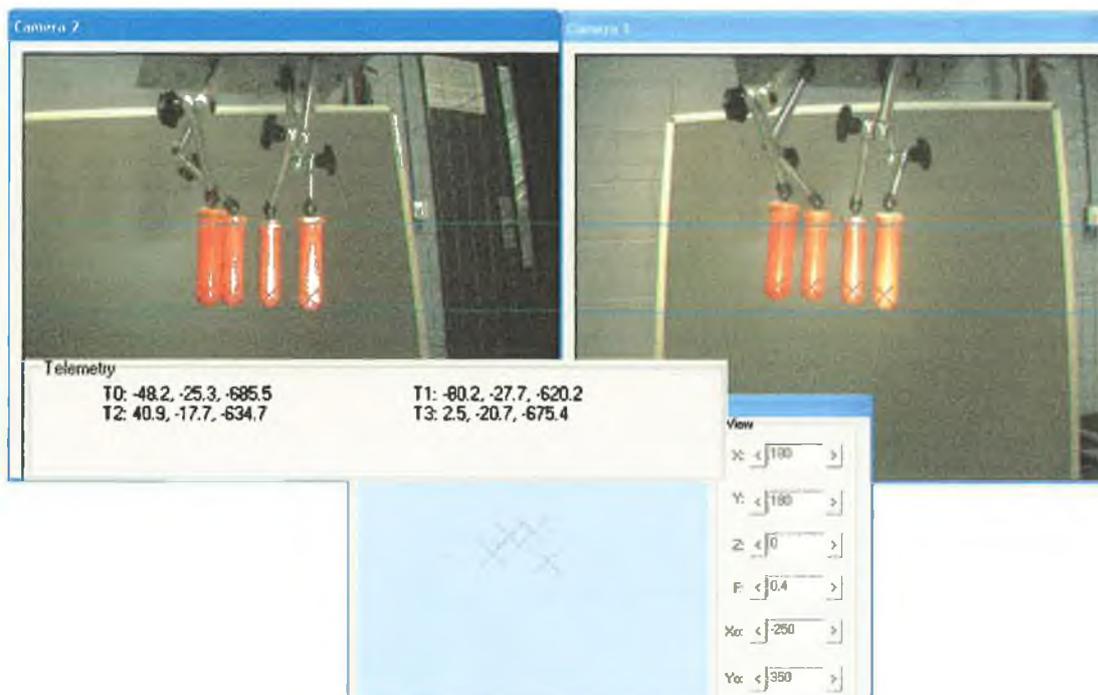


Fig 3.31 – Tracking targets in constant conditions

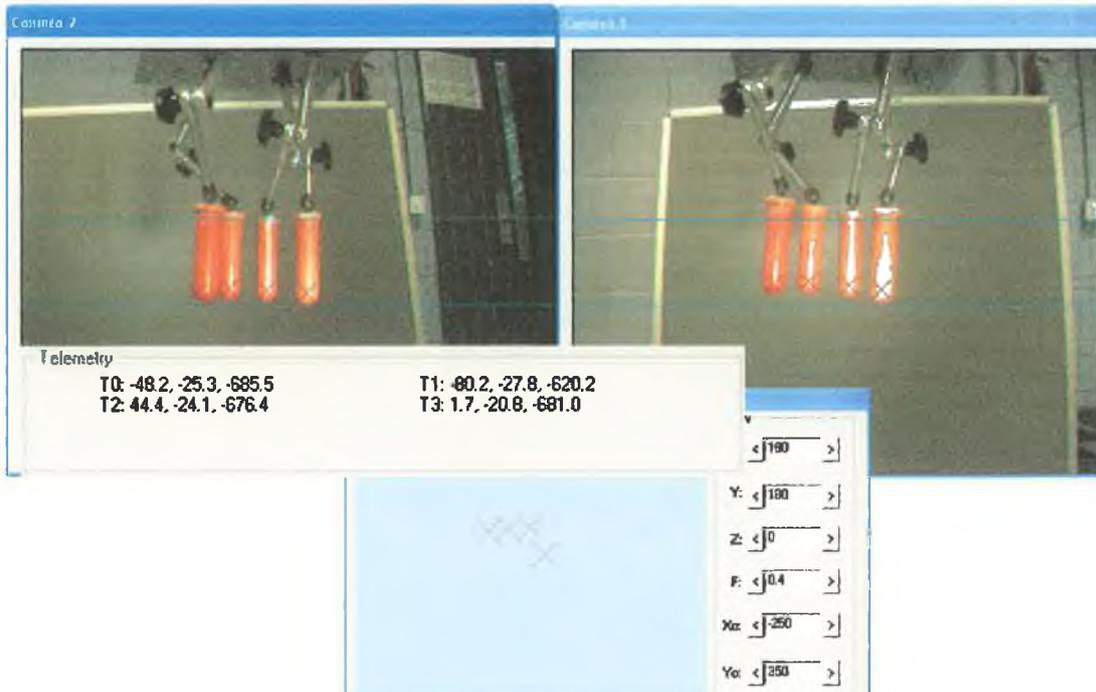


Fig 3.32 – Tracking targets in constant conditions: switching in telemetry output

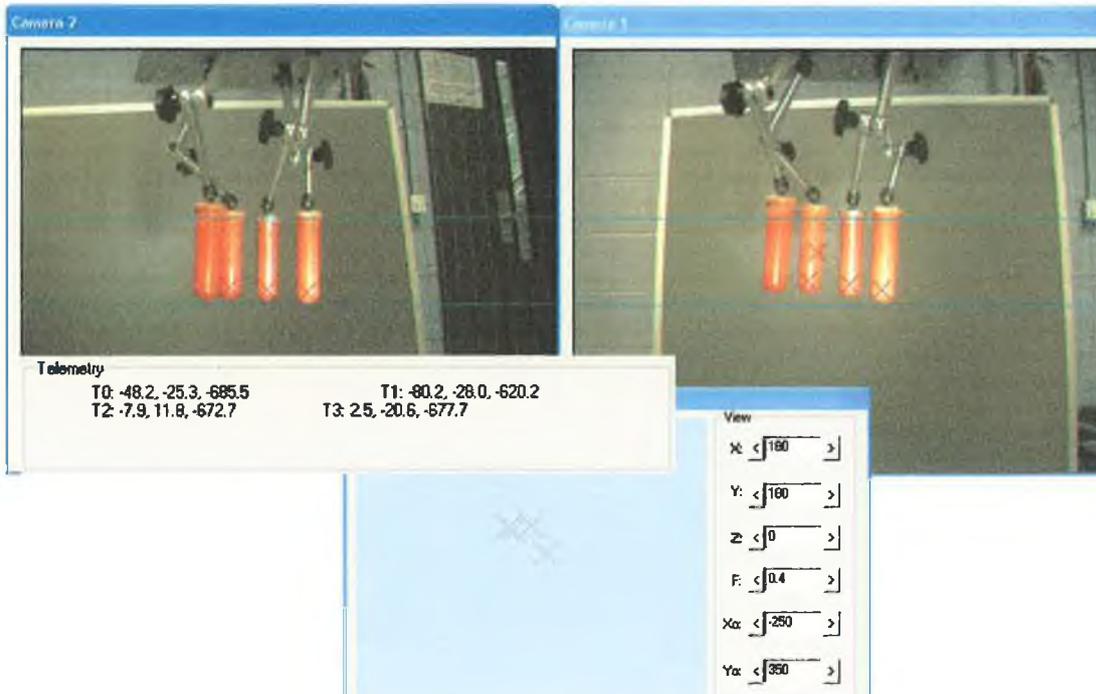


Fig 3.33 – Tracking targets in constant conditions: switching in output: failed detection

Fig 3.31, Fig 3.32 and Fig 3.33 are a sequence of screen shots taken while the system was tracking four teat targets under constant, favourable conditions. The output of the teat locations is seen to vary in the telemetry display and the 3D display. In Fig 3.33 the system fails to correctly identify all four teats. There was no change in the environment conditions in the time between capturing the contents of Fig 3.31 and Fig 3.33 (or Fig 3.32). This sequence clearly demonstrates the problems the system has in consistently and reliably identifying teats.

3.2.2.m - Miss-identifications: Shadows

The task of successfully ignoring a non teat object that closely resembles a teat is complicated. The need for further development of the system in this area is made apparent in Fig 3.34.

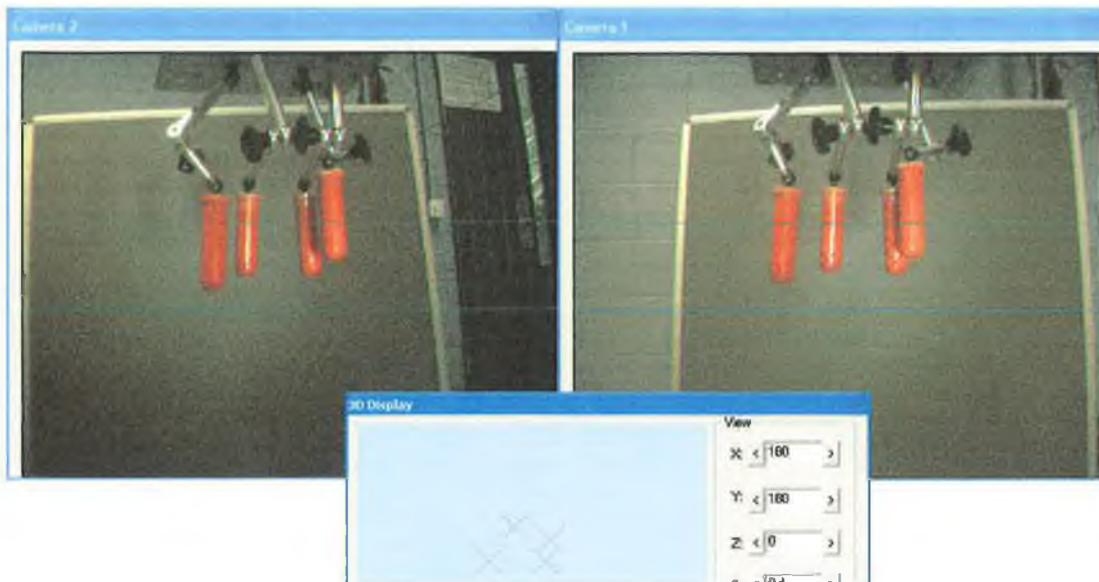


Fig 3.34 – Shadow incorrectly identified as a teat

In Fig 3.34 the shadow of a front teat on a rear teat is incorrectly identified as a teat. Whereas this is really an illumination issue (correct lighting positioning would eliminate shadows) the example serves to demonstrate the readiness of the system to identify any teat shaped artefact in the image as in fact being a teat.

3.2.3 - Accuracy of System in Locating Teats

This section examines the accuracy of the IceRobotics system in locating target teats by measuring the difference in the real world positions of target teats and the 3D position estimates made by the system. At the time of acquiring the system IceRobotics claimed it produced measurements to an accuracy of $\pm 2\text{mm}$. This claim was later changed to $\pm 5\text{mm}$. This section is divided into four further sections: 1) The base measurement of target positions, 2) relating camera measurements with real world measurements, 3) setting up of the system for optimal accuracy and 4) results of the accuracy tests.

3.2.3.a - Base Reference Measurements

To assess the level of accuracy of the IceRobotics system the 3D position coordinates that it outputs are compared to known accurate reference measurements. Differences in the values correspond to the error in the vision systems measurements. A dummy udder rig was designed and built to allow teats to be freely positioned and oriented in space. Once determined accurately in a desired location the teats can be secured there. The position of the teats is then determined accurately using a 3D digitiser.



Fig 3.35 – Dummy udder rig

The dummy udder rig is pictured in Fig 3.35. A 1.5m tall steel frame supports a steel plate (in the centre of the image), the height of the plate is adjustable. Attached to the plate are four dummy teats by means of four linkages. The linkages allow infinite positional adjustment and orientation of the teats. Each linkage contains two ball and socket joints at the extremities and one rotational joint at the hinge (midpoint of linkage). The joints can all be locked simultaneously by tightening the screw handle.



Fig 3.36 – Microscribe digitiser

The 3D digitiser is pictured in Fig 3.36. It is called a Microscribe and is manufactured by the Immersion Corporation [43]. The stylus can be manually moved within a spherical volume around the base of the Microscribe. The 3D position of the tip of stylus, the scribe, is calculated by the Microscribe on demand. The accuracy of the Microscribe is within ± 1 mm.

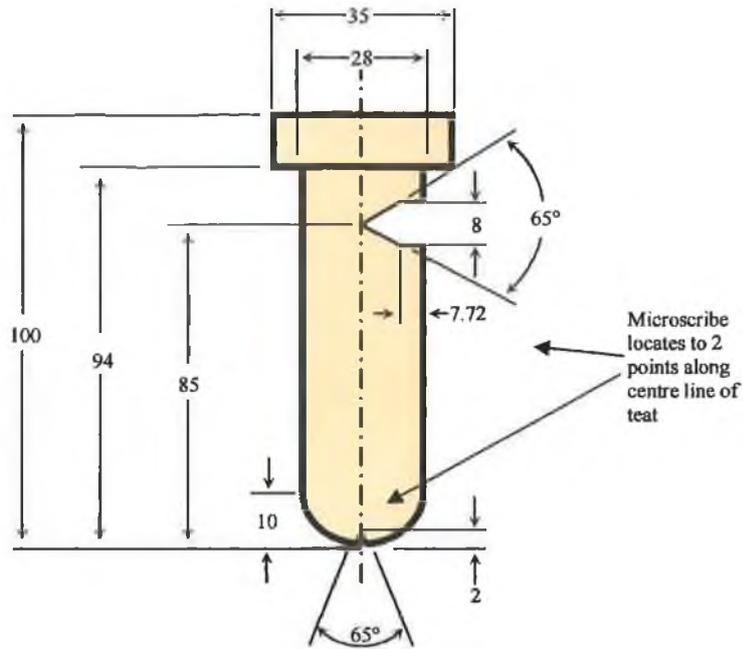


Fig 3.37 – Cross-section of dummy teat, holes locate scribe tip to central axis

The dummy teats are manufactured from aluminium. There are two holes in each designed to locate the tip of the scribe to two different points on the central axis. The teat dimensions are seen in cross-section of Fig 3.37.

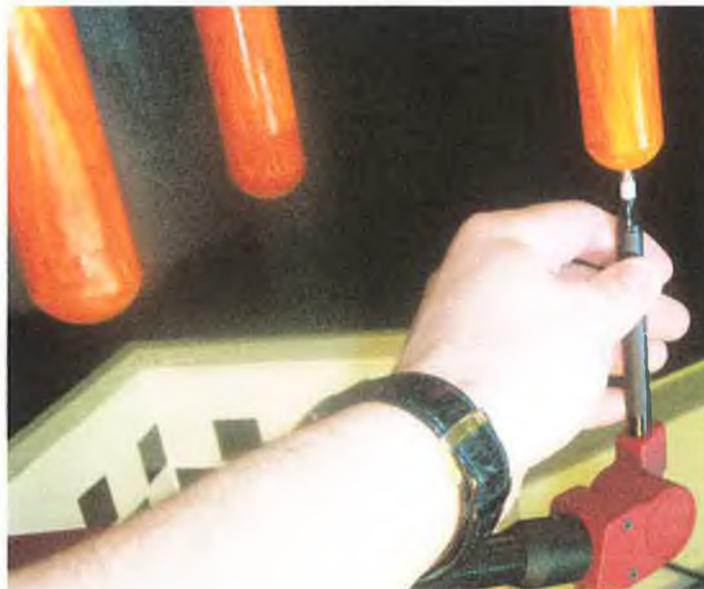


Fig 3.38 – Manually locating the tip of scribe to the guide hole in the teat

Fig 3.38 illustrates the manual location of the scribe to the guide hole in the dummy teat. The scribe is held there whilst a 3D coordinate is captured. The Microscribe coordinates are with reference to its internal origin and reference frame. This is automatically established when the unit is switched on but it can also be redefined manually. The orientation of the coordinate frame is established manually by defining three points on a plane to which the XY plane must be parallel. Another point is chosen to represent the new origin. A further two points are defined, one on the positive X axis and another on the positive Y axis. These points constrain all three axis of the reference frame at the new origin. A point is chosen by bringing the tip of the scribe to the chosen location in space and pressing the capture foot pedal.

3.2.3.b - Relating Camera Measurements to Base Measurements

To be able to make direct comparisons between the positions measured with the Microscribe and the positions measured with the cameras the coordinates must be converted into the same reference frame; this will be referred to as the World Reference Frame throughout this chapter. By redefining the Microscribe reference frame to coincide with the World Reference Frame any point measured with the Microscribe will be a World point. The coordinates given out by the camera are with respect to the systems internal coordinate system, defined during calibration. The origin of this frame is somewhere in space between the two cameras. Instead of trying to find where in space this point is, a transform is established that will map any points in the cameras reference frame to that of the World Reference Frame. Such a transformation can be derived if the coordinate values of sufficiently many points are known with respect to both of the reference frames. If there are three points known coplanar which form the vertices of a right angled triangle and the right angle vertex is the origin of the World Reference Frame then these may be used to define the transformation. The World Reference Frame is physically defined in space by three dummy teats in a right angled formation as seen in Fig 3.39. The three teats are screwed into 3 accurately machined holes on the base plate. Fig 3.40 contains a printout from a CMM (Coordinate Measuring Machine) used to check that the teats in the three-teat pattern do in fact form a right angled triangle.

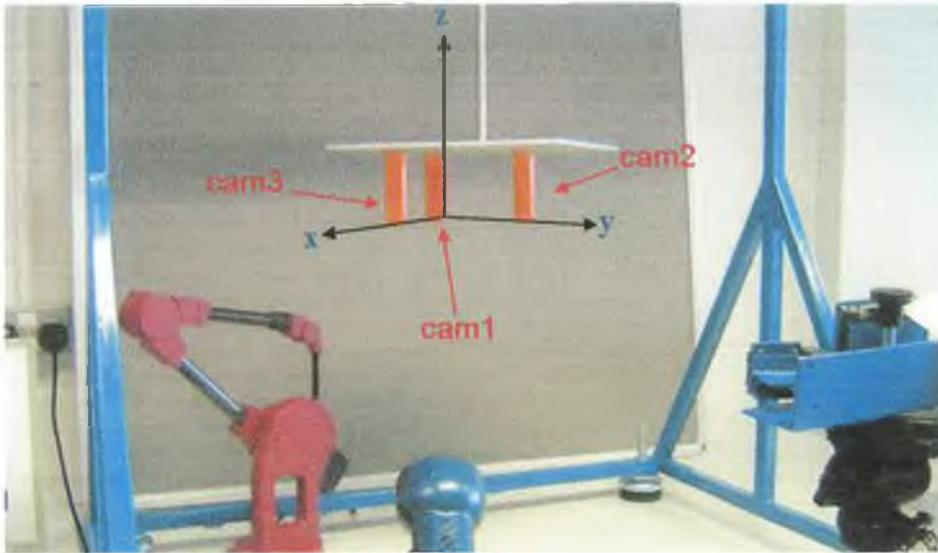


Fig 3.39 – 3 Teat Reference Frame

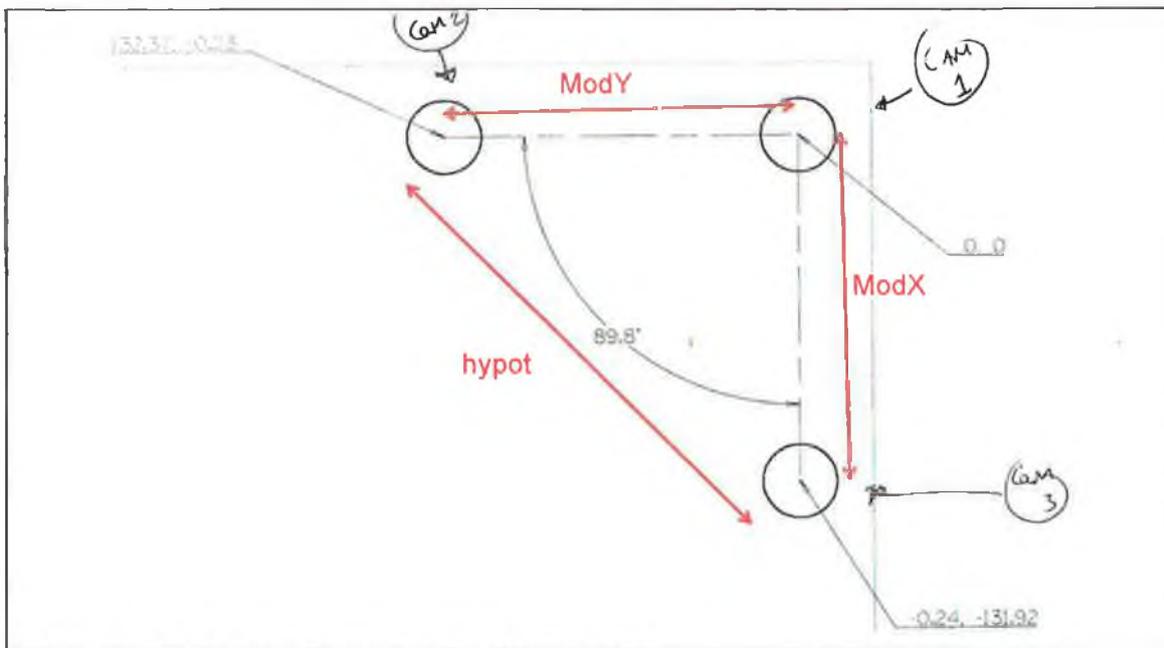
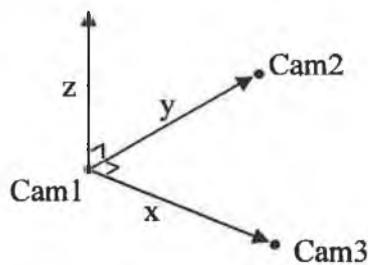


Fig 3.40 – CMM printout: measurements of 3 teat pattern

It is seen in Fig 3.40 that the three teat pattern is in close approximation to the right angle triangle condition. Also, the tips of the teats are all coplanar and the distances from the two outer teats to the origin are known. The coordinate system of the Microscribe is set up to coincide with the World Reference Frame (defined by the tips of the three teat pattern) by locating the scribe to the purposely machined holes in the end of the teats. The

transformation that converts coordinates from the camera system to the World Reference Frame is built using the coordinate correspondences in the two reference frames. The reason that the teat pattern technique has been employed here is now obvious: the IceTracker can identify the three teats so that their 3D coordinates in the camera reference frame are known. The coordinates of the same three teats in the World Reference Frame are also known: the middle teat is the origin of the World Reference Frame and thus (0,0,0). The leftmost teat as the camera views the scene (as in Fig 3.39) defines the X-axis, its coordinate is (132,0,0). The remaining teat, defining the Y-axis, has the coordinate (0,132,0). The transformation Matrix is therefore built as in Method 3.1.

Method 3.1



$${}^{World}T_{Camera} = \begin{bmatrix} X_x & Y_x & Z_x & Cam1_x \\ X_y & Y_y & Z_y & Cam1_y \\ X_z & Y_z & Z_z & Cam1_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The transform ${}^{World}T_{Camera}$ transforms coordinates in the World Reference Frame to point in the IceRobotics Vision system coordinate frame. The inverse of this transform can be used to convert the output of the IceTracker into World coordinates.

$${}^{Camera}T_{World} = \begin{bmatrix} X_x & Y_x & Z_x & Cam1_x \\ X_y & Y_y & Z_y & Cam1_y \\ X_z & Y_z & Z_z & Cam1_z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1}$$

where X (X_x, X_y, X_z) is the unit vector in the X direction, Y (Y_x, Y_y, Y_z) is the unit vector in the Y direction and Z (Z_x, Z_y, Z_z) is the unit vector in the Z direction.

$$X = \frac{Cam3 - Cam1}{|Cam3 - Cam1|} \quad Y = \frac{Cam2 - Cam1}{|Cam2 - Cam1|} \quad Z = X \times Y$$

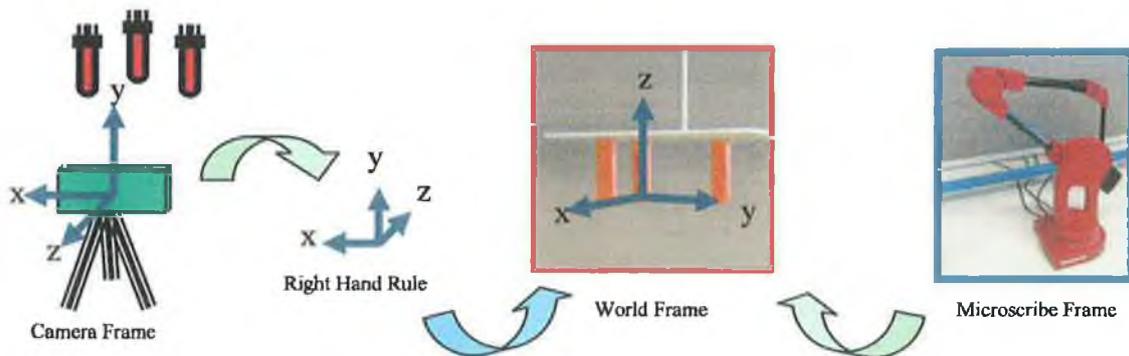


Fig 3.41 – Different coordinate systems mapped to World Reference Frame

Fig 3.41 illustrates points in the different coordinate frames of the Microscribe and the vision system being transformed to the same World Reference Frame. It is in this reference frame that the measurements from the two systems can be directly compared allowing the accuracy of the IceRobotics system to be assessed. The illustration in Fig 3.41 has a right hand rule conversion of points in the camera frame before they are converted into the World Frame. The reason for this is that the coordinates given out by the IceRobotics system are not with respect to a standard right hand rule configuration reference frame; the direction of the Y-axis must be reversed in order for it to conform. This is simply achieved by multiplying the Y-component of the camera coordinates by -1.

3.2.3.c - Setting up IceRobotics System for Accuracy Testing

The stereo camera rig is infinitely adjustable. The cameras can be set up in various different configurations as well as being located in different positions relative to the targets. This is illustrated in Fig 3.42.

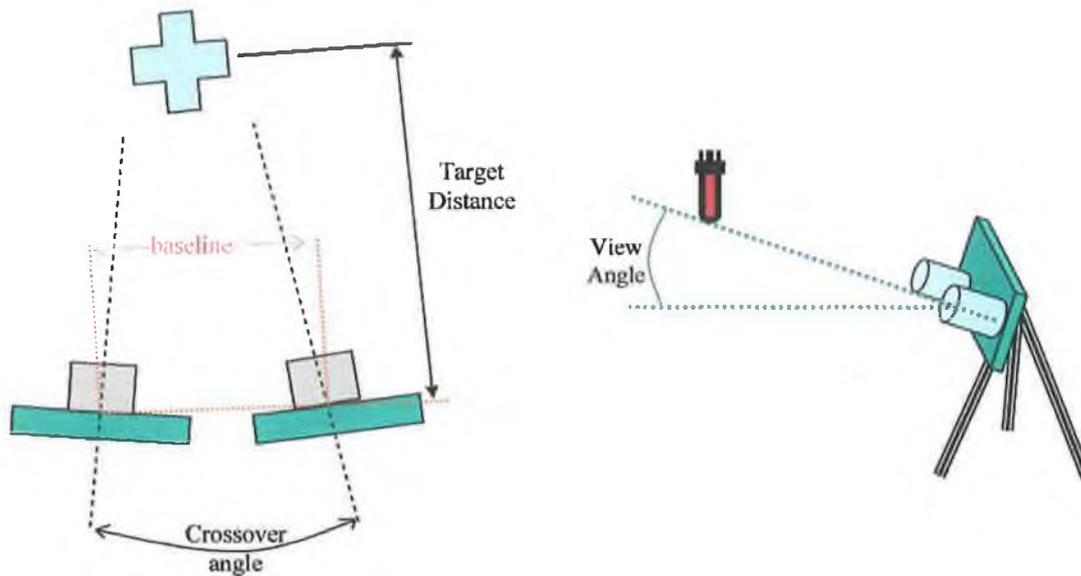


Fig 3.42 – Parameters in setting up cameras

In Fig 3.42 the different camera setup parameters are illustrated. The camera baseline is the distance between the two camera centres. The crossover angle is the angle made by the intersection of the normals defining the principal planes of each camera. This describes how much the cameras are turned in towards each other. The target distance represents how far the camera rig has been positioned from the targets. The view angle represents the height of the camera rig with respect to the lowest point of the targets.

To optimise the accuracy of the measurements during the accuracy assessment these parameters must be suitably chosen beforehand. This has been done using a process of trial and error. Controlled tests were performed in which each parameter was gradually varied until the error of the system in measuring the positions of the three-teat target was minimized. The error was assessed by measuring the absolute distances between the coordinates outputted by the system for the three teats. These distances should correspond to the lengths of the sides of the three teat triangle: 132mm, 132mm and the hypotenuse 186.68mm ($\sqrt{132^2 + 132^2}$). The first step in setting up the cameras was the horizontal alignment. The cameras were adjusted to that an object appears at the same height in both of the images. This was done manually using the horizontal blue line of the vertical search region as a guide. This is seen in Fig 3.43.

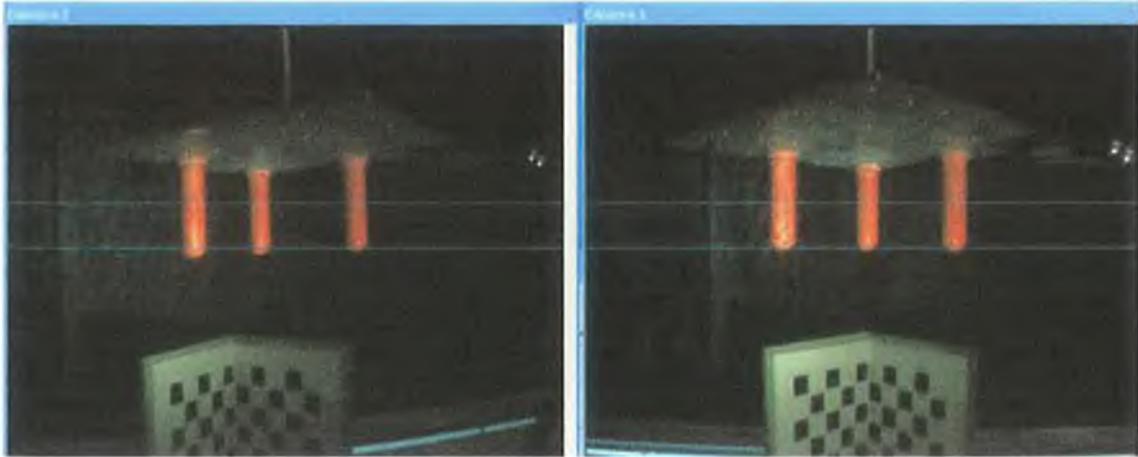


Fig 3.43 – Horizontal alignment of the stereo cameras

After a change in any of the camera parameters the system must be recalibrated. To ensure accurate calibration the calibration target must be in focus. This follows for the teat identification and measurement process, the targets should be in focus. The calibration pattern is therefore held the same distance from the cameras as the eventual targets, and the focus is adjusted to produce optimal results at this distance. The best focus was found using a sheet of text as an aid as seen in Fig 3.44. The sheet is held at the target distance and the focus is adjusted until it can be read on the client display.



Fig 3.44 – Focus adjusted so that the text can be read on the client video display

Increasing the camera baseline will reduce the error in the triangulation of the 3D coordinates. A larger baseline means that all of the lengths making up the triangle in the triangulation process are longer so the percentage error in measuring these lengths is reduced. However, increasing the baseline of the cameras reduced the ability of both cameras to see the same target. Increasing the cross-over angle will bring an object back into the two camera views. However, both cameras will not be able to see both sides of the object. A target teat in the centre of the views will have its left side obscured from the right camera and its right side obscured to the left camera. Increasing the cross-over angle will also centre the target in both of the images. This has the effect of reducing the disparity in the point correspondences. This will increase the percentage error in measurements of the disparity. There is therefore a trade-off between the accuracy of the teat position measurements and the ability of the system to identify the teats in the first place. To find the balance the camera baseline is adjusted to a maximum value that still allows the system to successfully identify the teats. The cross over angle is minimised in the same way. The optimised setup parameters are illustrated in Fig 3.45 and Fig 3.46 and Table 3.5.

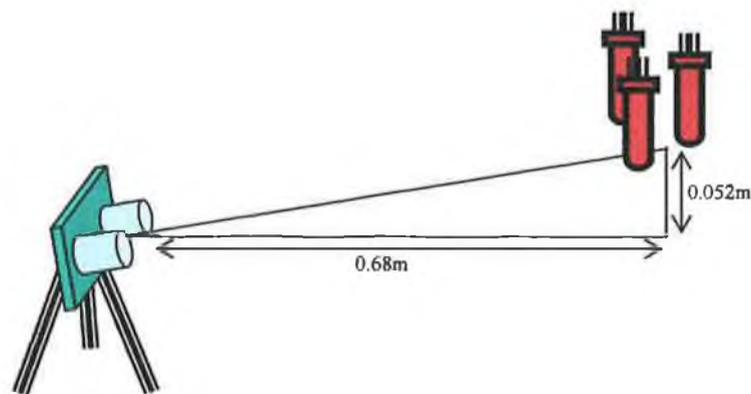


Fig 3.45 – Distance and height of camera relative to target centeroid

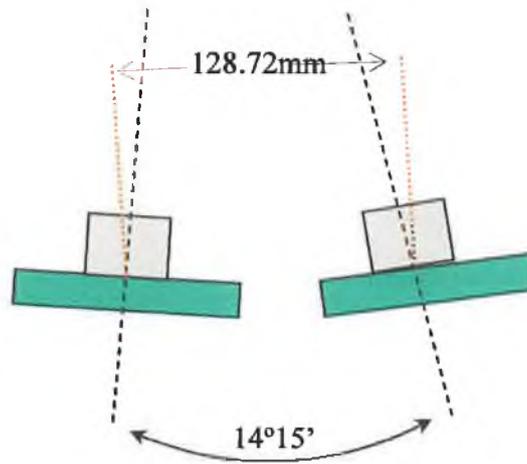


Fig 3.46 – Baseline and angle between principal axes (cross-over angle)

Minimum Distance	Maximum Distance	Seach Window	Max Texts	Teat Width	Image Dimension	Auto Exp Target	Exposure	Illumination	Comment
200	1000	45-55	3	28-20	640x480	x	5000	LED below right Lamp below left night far lights on dark	3 matched very stable average taken
Cam1	Cam2	Cam3	medXc (132mm)	medYc (132mm)	hypot (166.67mm)				
-36.2888	-123.1898	64.3880	131.9614	131.5671	166.0682				
25.3798	32.1458	32.7843				Average Absolute Error:	0.623566667		
732.7412	634.1901	647.7513	Abs Error:	0.0398	0.4329	-1.3992	Average Percentage Error:		
			% Error:	0.03%	0.33%	-1.06%	Root Mean Squared Error:		
						1.465146071			

Table 3.5 – Optimised Setup Parameters

The top row of Table 3.5 contains the optimised system parameters. These are the variables that can be set using the Peek/Poke functionality. ‘Illumination’ documents the lighting conditions during the test. An LED array and a 60W lamp were used as light sources, their positioning as indicated in the table. ‘night’ refers to the ambient light from the sun., in this case it was minimal as it is night time. ‘far lights on’ refers to the overhead fluorescent lights at the other end of the room in which the test was conducted, ‘dark’ is a description of how the scene appears without the intended light sources of the LED array and the lamp. The comments ‘three matched’ and ‘very stable’ refer to the teat identification performance of the system. In this case all three teats are located and the

output is stable. 'average taken' refers to the fact that the system was allowed to run for an extended period and the position outputs were written to disk. The teat coordinates are calculated as the average of all the samples. This is to minimise the effects of any instabilities or dither in the output. The second row of Table 3.5 contains the three coordinate outputs from the system, Cam1, Cam2 and Cam3. The three lengths of the triangle sides (see Fig 3.40) are calculated: 'ModXc', 'ModYc' and 'hypot' and displayed in the table. The error of the system measurements are given as absolute distances in millimetres and as a percentage of the overall length measured. The RMS (root mean squared) error is the square root of the sum of the squares of the errors in the three lengths.

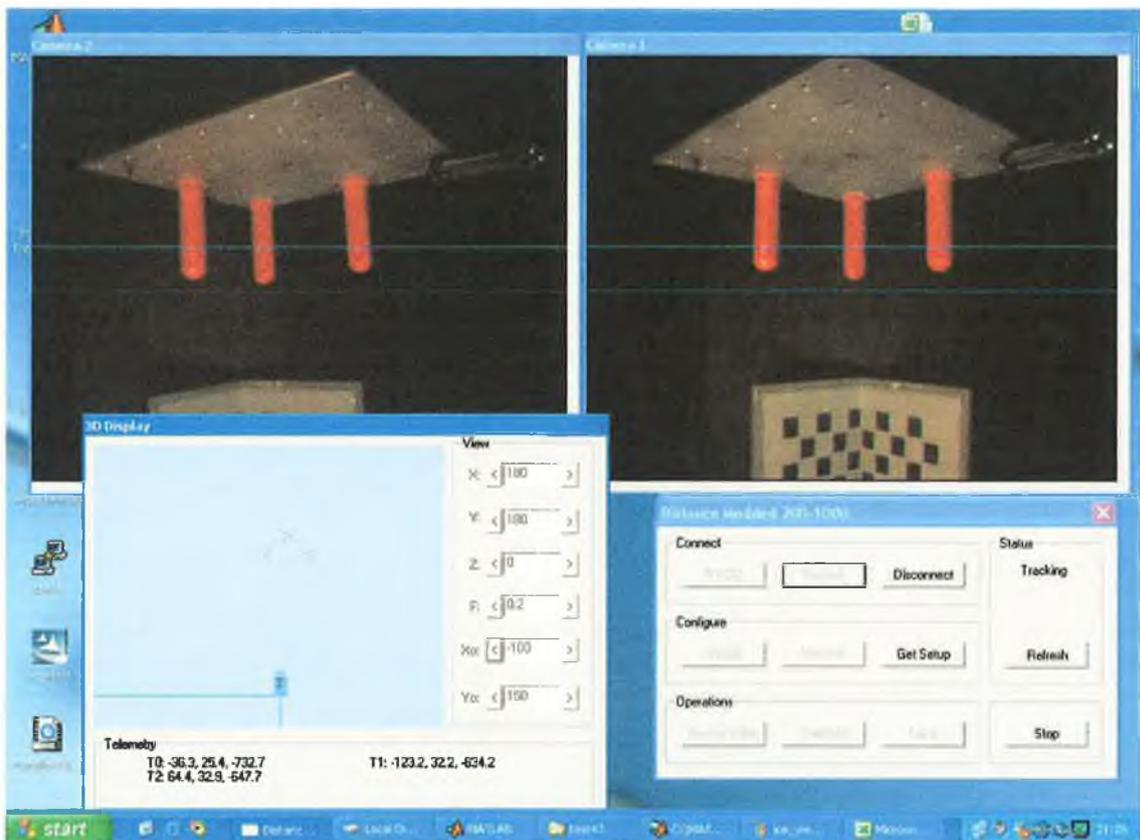


Fig 3.47 – Screen shot from client during Setup Parameter optimisation

These points are used to build the camera to World transformation as in Method 3.1.

3.2.3.d - Accuracy Testing

The goal of the accuracy testing is to discover the error of the system in determining the position of a target teat. The target teat was placed in different positions throughout the target region. The system was detecting the teat and outputting its location at each of these positions. The Microscribe digitiser was used to acquire a comparison measurement for each teat position. The measurements coordinates from the vision system and the digitiser coordinates were transformed to the same World Reference Frame. The difference in these World points is the error in the vision systems measurements.

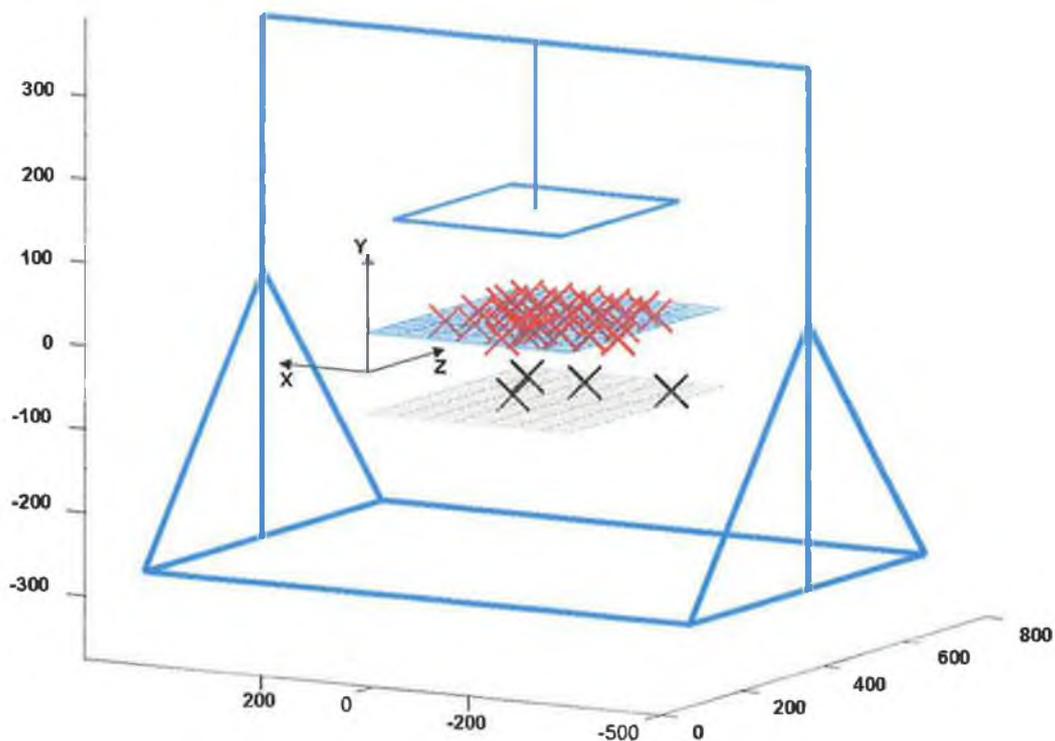


Fig 3.48 – Range of physical locations of teat ends in World Space for accuracy testing

Fig 3.48 illustrates the range of physical location of the teat ends in World space for the accuracy tests. A single teat was moved to each location indicated and its position coordinates were measured using both the vision system and the Microscribe. In accordance with the axis in Fig 3.48 the camera is located at (0, 0, 0).

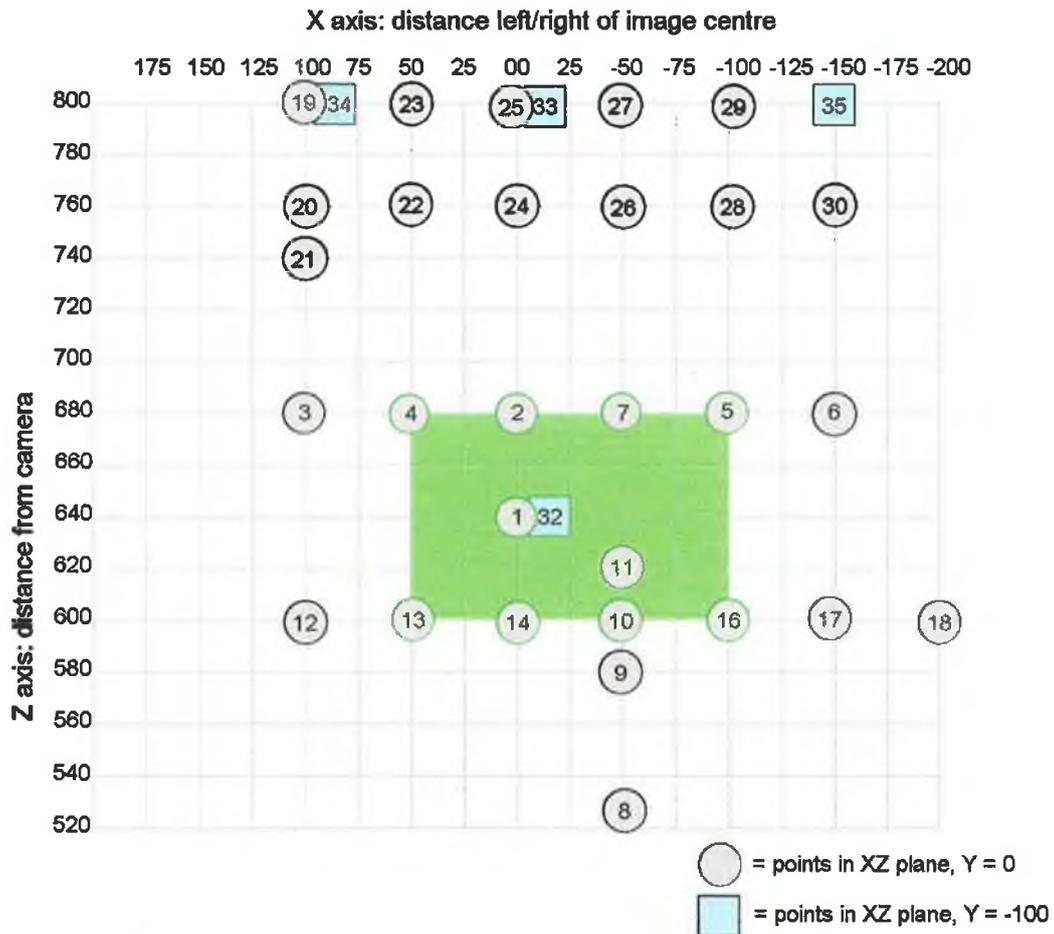


Fig 3.49 – Plan view graph of teat locations for tests 1 -35

The tests numbered 1 to 35 (excluding 15&31) were carried out with the teat located as in Figure 3.48. The Z co-ordinate represents how far the teats are from the camera, the X coordinate represents how far to the left or to the right of the camera the teat is. With respect to these axes the camera is located at (0, 0, 0). The green region corresponds to the region within the green box in Fig 3.52.

3.2.3.e - Accuracy Testing Results

Test #	XY Plane		Camera Coordinates In World Reference Frame			Microscribe Coordinates In World Reference Frame			Difference			Absolute Error (mm)
	X	Z	X	Y	Z	X	Y	Z	X	Y	Z	
1	640	0	30.407	-20.275	-12.273	35.486	-15.972	-11.941	-5.079	-4.303	-0.332	6.67
2	680	0	13.219	-53.991	-5.131	12.866	-51.959	-4.372	0.353	-2.032	-0.759	2.20
3	680	100	94.619	-120.409	-7.868	84.116	-129.438	-6.270	10.503	9.026	-1.598	13.94
4	680	50	51.312	-93.697	-10.840	44.870	-97.086	-10.078	6.441	3.499	-0.762	7.37
5	680	-100	-69.086	4.584	2.985	-70.908	4.229	3.733	1.822	0.356	-0.748	2.00
6	680	-150	-105.725	47.671	-7.691	-120.382	40.384	-5.625	14.657	7.287	-2.065	16.50
7	680	-50	-26.279	-18.554	-15.226	-26.288	-18.007	-14.811	0.009	-0.547	-0.415	0.69
8	525	-50	73.502	98.424	-13.357	84.606	101.594	-13.493	-11.104	-3.170	0.137	11.55
9	580	-50	36.268	55.792	-18.867	43.574	59.272	-18.645	-7.306	-3.480	-0.242	8.10
10	600	-50	21.542	39.434	-12.006	25.898	42.758	-11.573	-4.356	-3.324	-0.433	5.50
11	620	-50	10.862	24.329	-17.033	15.499	28.394	-16.179	-4.637	-4.065	-0.654	6.23
12	600	100	140.144	-54.452	-12.714	135.900	-59.328	-10.720	4.244	4.876	-1.995	6.76
13	600	50	104.960	-24.840	-10.479	103.395	-26.994	-9.334	1.665	2.154	-1.145	2.90
14	600	0	65.176	9.406	-10.101	68.723	12.296	-9.609	-3.547	-2.889	-0.492	4.60
16	600	-100	-15.868	70.797	-8.081	-12.961	72.005	-7.257	-2.908	-1.268	-0.624	3.29
17	600	-150	-52.648	107.014	-22.108	-63.131	101.691	-21.207	10.489	5.324	-0.901	11.79
18	600	-200	-91.156	135.164	-9.621	-121.622	122.405	-5.215	30.465	12.759	-4.406	33.32
19	800	100	19.063	-210.121	11.694	1.601	-222.084	19.482	17.462	11.963	-1.768	21.24
20	760	100	35.365	-177.407	10.852	21.502	-186.186	12.332	13.863	8.779	-1.680	16.49
21	740	100	50.360	-159.842	5.348	34.411	-171.208	7.543	15.949	11.366	-2.195	19.71
22	760	50	-5.061	-147.783	-2.288	-15.963	-151.844	-0.642	10.902	4.061	-1.646	11.75
23	800	500	-26.363	-175.423	13.016	-39.634	-181.010	12.960	13.271	5.586	0.056	14.40
24	760	0	-44.292	-113.724	13.372	-51.293	-114.745	14.347	7.001	1.021	-0.976	7.14
25	700	0	-75.614	-142.690	13.095	-84.431	-143.670	14.262	8.817	0.980	-1.168	8.95
26	760	-50	-80.849	-79.015	0.114	-88.704	-81.322	1.409	7.855	2.307	-1.295	8.29
27	800	-50	-105.887	-117.643	-6.685	-115.440	-120.398	-5.481	9.752	2.755	-1.204	10.21
28	760	-100	-117.644	-56.254	18.311	-127.085	-60.174	19.328	9.440	3.920	-1.017	10.27
29	800	-100	-139.976	-78.338	17.725	-153.626	-84.666	19.735	13.850	6.328	-2.010	15.36
30	760	-150	-152.294	-17.396	5.678	-176.107	-29.950	8.289	23.812	12.553	-2.591	27.04
32	640	0	16.197	-35.395	-106.127	4.343	-44.354	-105.858	11.853	8.959	-0.263	14.86
33	800	0	-78.357	-155.643	-96.529	-90.064	-157.422	-94.662	11.706	1.779	-1.867	11.99
34	800	100	21.459	-217.608	-96.975	1.012	-233.435	-98.347	20.446	15.827	1.372	25.89
35	800	-150	12.418	-53.222	-4.122	8.614	-53.041	-3.906	3.805	-0.181	-0.218	3.82

Average Error for entire Test Region **11.24**

Table 3-6 – Accuracy Test Results

Table 3.6 contains the results of the accuracy tests. The coordinates are the outputs (in millimetres), for each test location, of the vision system and the Microscribe converted to the World Reference Plane. The difference between the two measurements is shown on the right side of the table. The root mean squared value of these three component values is displayed in the rightmost column. This is regarded as the error produced by the vision system. The XY plane values highlighted in grey are on a plane 100mm below the XY plane highlighted in white.

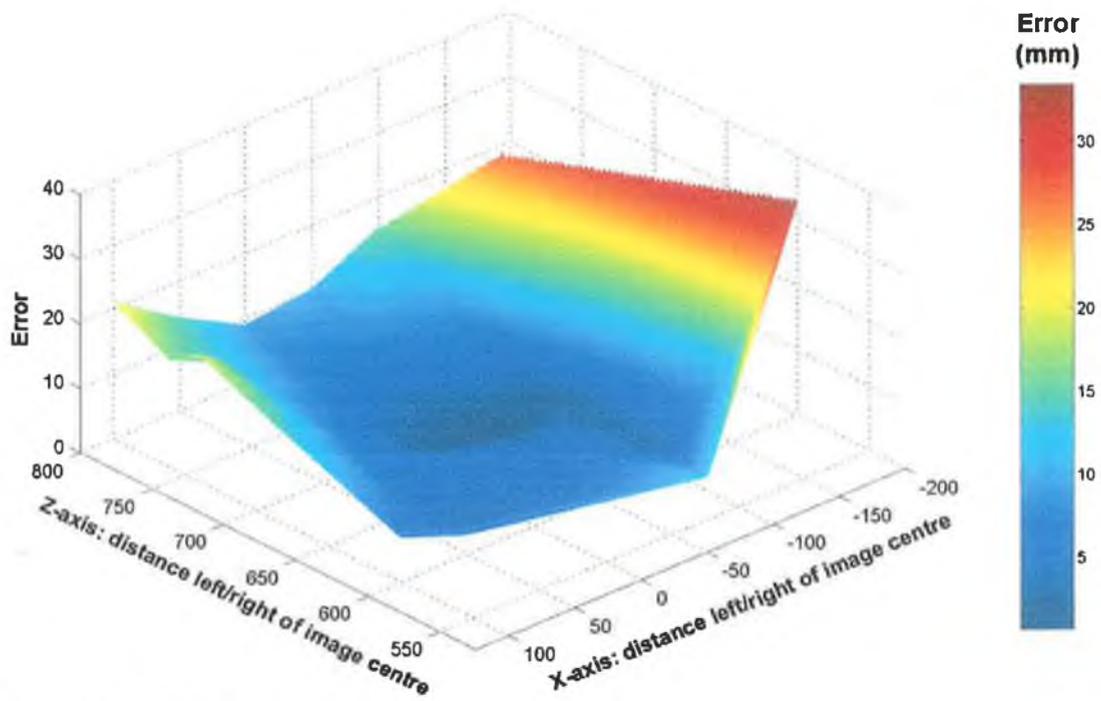


Fig 3.50 – Absolute Error surface

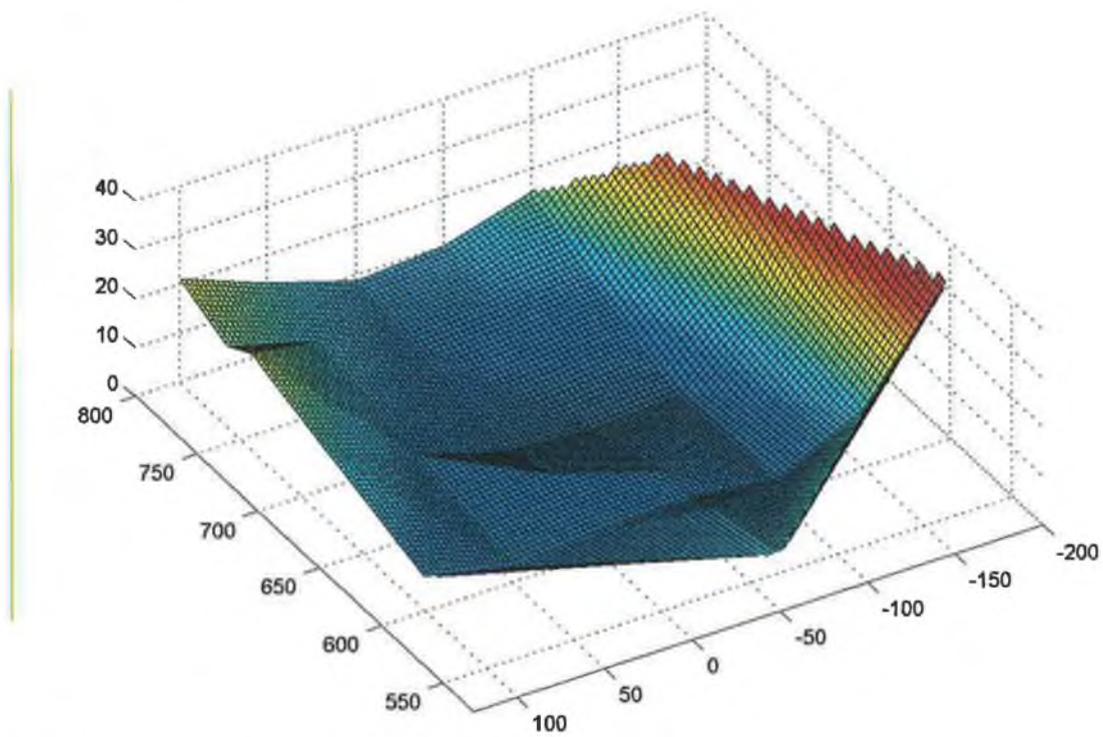


Fig 3.51 – Absolute Error surface with mesh lines displayed

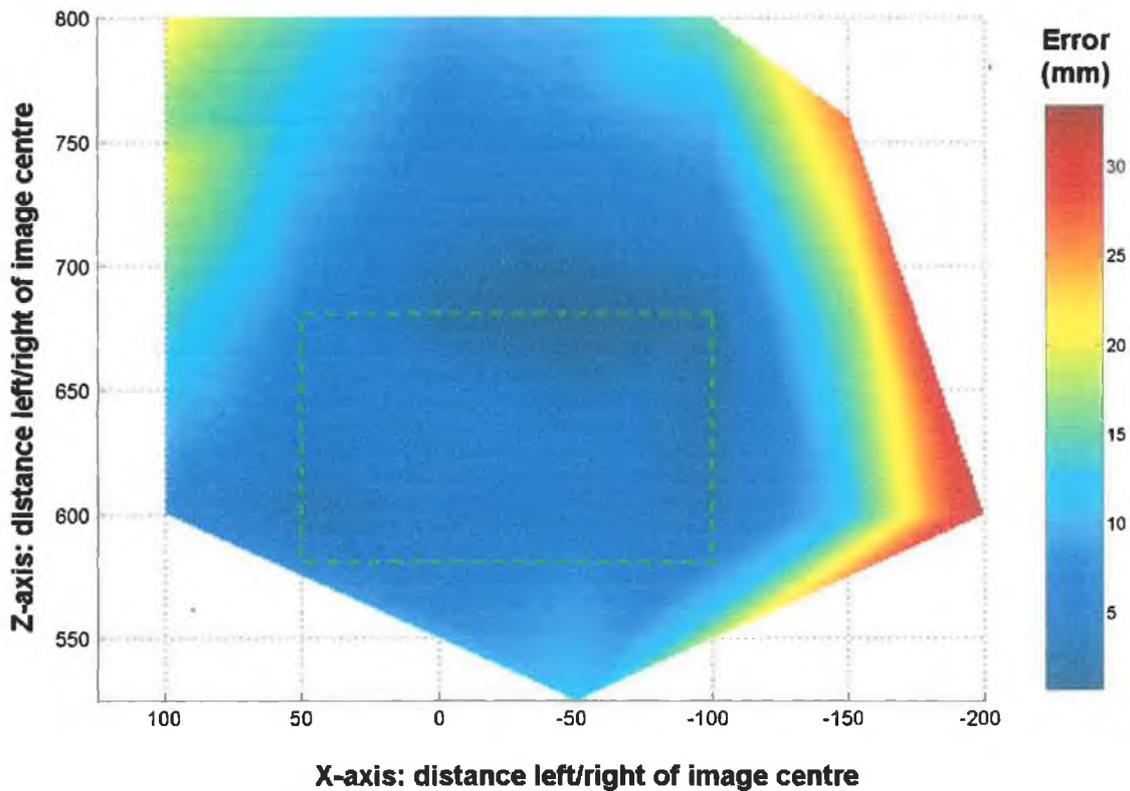


Fig 3.52 – Absolute Error Surface, plan view

Fig 3.50 and Fig 3.52 show the absolute error of the vision systems measurements. In Fig 3.50 the height of the graph is the absolute error in millimetres. Fig 3.52 is a plan view of the previous plot. The absolute error is represented by colour as indicated by the legend. There is a grid overlaid on the surface of the plot in Fig 3.51 to aid the spatial visualisation of the error. Table 3.7 contains only the tests located within the green region of Fig 3.49. This region is also indicated in Fig 3.52. The average absolute error of these ten tests is 4.14 millimetres. The region is 150mm wide (parallel to image plane) by 80mm deep (perpendicular to image plane).

	X	Y	Z	
1	-5.079	-4.303	-0.332	6.67
2	0.353	-2.032	-0.759	2.20
4	6.441	3.499	-0.762	7.37
5	1.822	0.356	-0.748	2.00
7	0.009	-0.547	-0.415	0.69
10	-4.356	-3.324	-0.433	5.60
11	-4.637	-4.065	-0.854	6.23
13	1.565	2.154	-1.145	2.90
14	-3.547	-2.889	-0.492	4.60
15	-2.908	-1.288	-0.324	3.29

Average Error 4.14

Table 3.7 – Tests within the green region

Chapter 4 - Teat Angulations

4.1 - Outline

This chapter presents the algorithms developed and the knowledge used to determine the angular orientation of teats by visual inspection. Simultaneously attaching 4 milking cups to an animal, using an automated mechanical system, demands detailed information regarding the location and the orientation of the teats. The IceRobotics system only provides a coordinate representing the end of the teat in 3D space. It does not account for the angle at which the teat is orientated. The algorithms developed here build upon the functionality of the IceRobotics system, making use of the teat coordinates provided and the camera calibration data acquired.

4.2 - Algorithm

In defining the angulations of cow teats an assumption is made: the teats are axisymmetric about a central axis. It is this central axis which is used to define the angle at which the teat is orientated. The angular information is contained in two angles given from the central axis to two orthogonal planes. These planes are defined by the World Reference Frame; this is illustrated in Fig 4.1. The goal is to find two 3D points along this central axis to determine a vector describing the teat angle. These 3D points can be found from a stereo pair of images each containing a view of the teat. The 2D centreline of the teat is found in each image. Correlating points are found between the pair of images along each of the 2D centrelines, these are used to triangulate 3D points. The central axis of the teat intersects the triangulated points.

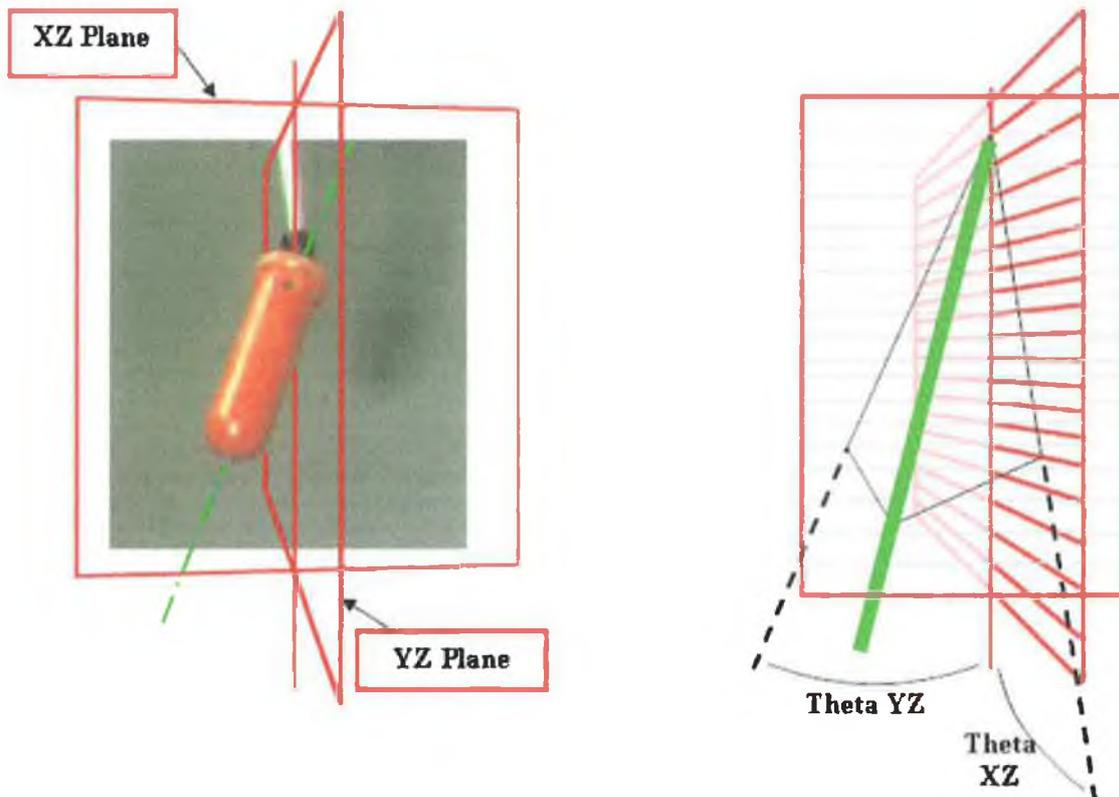


Fig 4.1 - Teat Angle described as angles from 2 planes

Find two 3D points along the centre line of the teat

If two 3D points, which are on the central axis of the teat, are known, then a vector describing the teat angle is uniquely defined. The steps taken in finding these two points are as follows:

1. Convert 3D teat coordinate provided by IceRobotics System into 2D image coordinates in the stereo pair of images also provided by the IceRobotics system
2. Convert the stereo pair of images into greyscale
3. Crop each image in the pair to a region of interest containing the teat in question based on the 2D image coordinates
4. Isolate the edges in the cropped images using an edge detector
5. Match straight lines to the edges found
6. Isolate lines corresponding to the sides of the teats in each cropped image

7. Translate side-line segments from position in cropped image to appropriate position in full-size image
8. Calculate the line which bisects the lines corresponding to the 2 sides of the teat in both images. The bisectors found are the centre lines of the teat in the corresponding image
9. Chose two arbitrary points in the left image that are elements of the centre line of the teat in the left image
10. Calculate the Epipolar lines in the right image corresponding the two chosen points in the left image
11. Find the points of intersection of the Epipolar lines with the centre line of the teat in the right image, these points correspond to the points on the centre line in the left image
12. Triangulate the 3D position of 2 points on the vector describing the teat angle using the 2 pairs of corresponding points from the stereo pair of images
13. Transform vector to the World Reference Frame for direct comparisons with test rig measurements
14. Further transform the vector to the Test Rig Reference Frame for illustration purposes

4.2.1 - Convert 3D teat into 2D image coordinates in the stereo pair of images

The output of the IceRobotics system is a group of point coordinates in the form (X, Y, Z) representing the location of teats in 3D space, with respect to the IceRobotics reference frame. The number of coordinates relates to the number of teats identified by the system. The same images used by the IceRobotics system are used to determine the teat angles, these images are streamed to disk. The location of the teats in these images can be determined using the coordinates outputted from the system in conjunction with the camera calibration data.

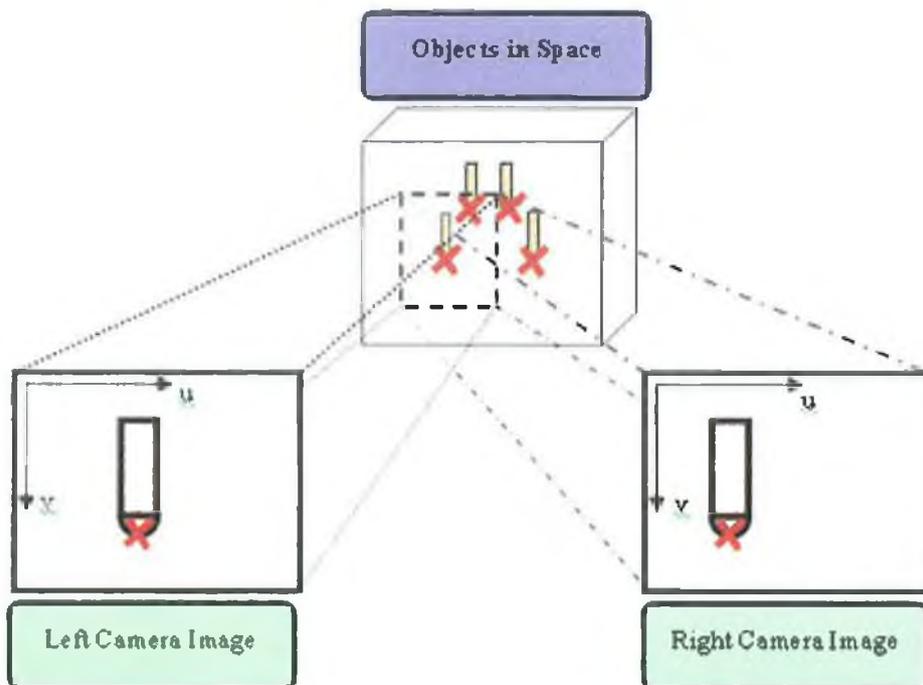


Fig 4.2 - Teat location in images of left and right cameras

The IceRobotics system determines and provides a general projection matrix for each of the two cameras, P_0 and P_1 , where P_0 is the Left Camera's projective matrix and P_1 is the Right Camera's projective matrix. The information contained in these matrices is used to map 3D coordinates in the World Reference Frame to 2D coordinates in the images of the left and right cameras, according to the relationship

$$\mathbf{x} = \mathbf{P}\mathbf{X} \quad (4.1)$$

where \mathbf{x} is an Image coordinate and \mathbf{X} is a World coordinate. The 3d coordinates, relating to the detected teat ends, outputted by the IceRobotics system are relative to the Camera Reference Frame [see Chapter 2.1.1.b for additional theory]. In order to apply the previous relationship the coordinates must first be transformed into the World Reference Frame. This is achieved either by pre-multiplying the Camera Coordinate by the inverse of the Extrinsic Matrix to find the point \mathbf{X} . Now $\mathbf{x}_{\text{left}} = (\mathbf{P}_0)\mathbf{X}$, where \mathbf{x}_{left} is the coordinate in the left image plane and is of the form $[U, V, S]$.

$$\mathbf{u} = \frac{\mathbf{U}}{\mathbf{S}} \quad (4.2)$$

$$\mathbf{v} = \frac{\mathbf{V}}{\mathbf{S}} \quad (4.3)$$

where $[\mathbf{u}, \mathbf{v}]$ is the pixel coordinate and \mathbf{S} is the scale factor of the homogenous coordinate. Alternatively, the point \mathbf{x} can be found by pre-multiplying the Camera Coordinate by the Intrinsic Matrix. The Camera Projection Matrices can be factorised into the Intrinsic and Extrinsic parameters [44] (see Appendix A.1.2).

Worked Example of Finding Teats in Images based on 3D coordinate outputs.

(Values are rounded off for clarity)



Fig 4.3 – Right Camera View of Dummy Teats

The 3D coordinate for the rightmost teat in the image of Fig 4.3 was found by the IceRobotics vision system to be

$$\mathbf{X}_{\text{cam}} = \begin{bmatrix} -91.1 \\ -40.9 \\ -673.36 \end{bmatrix} \quad (4.4)$$

where X_{cam} is a coordinate in the right camera coordinate frame (The sign of the Y component of this vector has been inverted to make coordinate correspond with the right hand rule). For this worked example the Projective Camera Matrices were calculated as follows:

$$P_0 = \begin{bmatrix} 652.105 & 15.264 & -81.719 & -200270.695 \\ 169.768 & -583.198 & 259.334 & -119704.771 \\ 0.562 & 0.051 & 0.825 & -641.590 \end{bmatrix} \quad (4.5)$$

$$P_1 = \begin{bmatrix} 621.486 & -18.507 & -261.278 & -183414.715 \\ 193.105 & -604.515 & 239.252 & -127938.532 \\ 0.726 & 0.073 & 0.684 & -682.356 \end{bmatrix} \quad (4.6)$$

It should be noted that the IceRobotics system goes against the convention of this work by referencing the camera coordinates with respect to the Right camera. P_0 and P_1 are for the right and left camera respectively.

P_0 decomposes using Givens Rotations as follows:

$$K_0 = \begin{bmatrix} 585 & 5.5 & 300 \\ 0 & -598 & 280 \\ 0 & 0 & 1 \end{bmatrix} \quad (4.7)$$

$$R_0 = \begin{bmatrix} 0.8267 & -0.0095 & -0.5626 \\ -0.0209 & 0.9987 & -0.0475 \\ 0.5623 & 0.0510 & 0.8254 \end{bmatrix} \quad (4.8)$$

$$\bar{C}_0 = \begin{bmatrix} 369 \\ 132.6 \\ 518.2 \end{bmatrix} \quad (4.9)$$

$$Ext = \begin{bmatrix} 0.8267 & -0.0095 & -0.5626 & -12.2445 \\ -0.0209 & 0.9987 & -0.0475 & -100.0567 \\ 0.5623 & 0.0510 & 0.8254 & -641.8885 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.10)$$

Where K_0 is the intrinsic camera matrix, R_0 is the rotation matrix, C_0 is the camera centre and Ext_0 is the extrinsic camera matrix. This decomposition was carried out using a Matlab script “decompose.m” (see Appendix C). The point X_{cam} is transformed from the camera coordinate frame to the image plane by pre-multiplying by the camera matrix K_0 .

$$x_0 = \begin{bmatrix} 585 & 5.5 & 300 \\ 0 & -598 & 280 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -91.1 \\ -40.9 \\ -673 \end{bmatrix} = \begin{bmatrix} -255480 \\ -163940 \\ -673 \end{bmatrix} \quad (4.11)$$

$$x_0 = \begin{bmatrix} U_0 \\ V_0 \\ S_0 \end{bmatrix} = \begin{bmatrix} -255480 \\ -163940 \\ -673 \end{bmatrix} \quad (4.12)$$

$$\begin{bmatrix} u_0 \\ v_0 \end{bmatrix} = \begin{bmatrix} U_0 / S_0 \\ V_0 / S_0 \end{bmatrix} = \begin{bmatrix} 379.6 \\ 243.6 \end{bmatrix} \quad (4.13)$$

where $[u_0, v_0]$ is the 2D pixel coordinate in the right image.

To next step is to find the location of the same teat in the left image. Firstly, X_{cam} is transformed from the camera coordinate frame of the right camera to the World coordinate frame to find the point X_{world} by pre-multiplying by the inverse of the extrinsic matrix of the right camera.

$$X_{world} = \begin{bmatrix} 0.8267 & -0.0095 & -0.5626 & -12.2445 \\ -0.0209 & 0.9987 & -0.0475 & -100.0567 \\ 0.5623 & 0.0510 & 0.8254 & -641.8885 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} -91.1 \\ -40.9 \\ -673 \\ 1 \end{bmatrix} = \begin{bmatrix} -84.1 \\ 58.3 \\ 15.55 \\ 1 \end{bmatrix} \quad (4.14)$$

The image coordinate in the left image, x_1 , is now found by pre-multiplying by the projective matrix of the left camera.

$$x_1 = \begin{bmatrix} 621.486 & -18.507 & -261.278 & -183414.715 \\ 193.105 & -604.515 & 239.252 & -127938.532 \\ 0.726 & 0.073 & 0.684 & -682.356 \end{bmatrix} \begin{bmatrix} -84.1 \\ 58.3 \\ 15.55 \\ 1 \end{bmatrix} = \begin{bmatrix} -240810 \\ -175690 \\ -730 \end{bmatrix} \quad (4.15)$$

$$x_1 = \begin{bmatrix} U_1 \\ V_1 \\ S_1 \end{bmatrix} = \begin{bmatrix} -240810 \\ -175690 \\ -730 \end{bmatrix} \quad (4.16)$$

$$\begin{bmatrix} u_1 \\ v_1 \end{bmatrix} = \begin{bmatrix} U_1/S_1 \\ V_1/S_1 \end{bmatrix} = \begin{bmatrix} 330.6 \\ 241.2 \end{bmatrix} \quad (4.17)$$

where $[u_1, v_1]$ is the 2D pixel coordinate in the left image.

The left and right camera views in Fig 4.4 show the projections of the 3D World point onto the left and right image planes. These 2D pixel coordinates are marked with a blue cross.

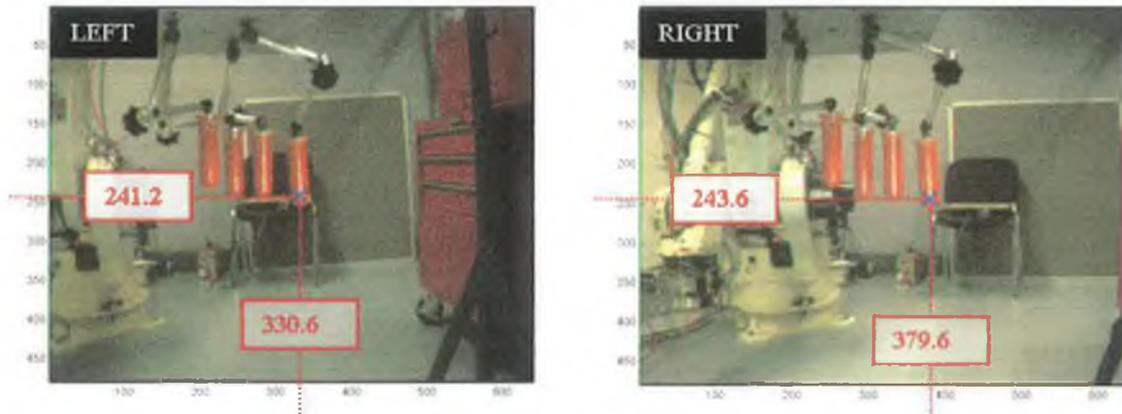


Fig 4.4 – Blue X represents World point projected onto left and right image planes

4.2.2 - Convert the stereo pair of images into greyscale

The stereo pair of images captured by the IceRobotics system are in a *.pcx file format. The images are of dimensions 640x480 pixels and have an RGB colour map. These images can be imported directly into the workspace of Matlab and stored as a 641x481x3 matrix. The '3' represents the number of layers; there is one for each of the three colour channels: Red, Green and Blue. The width and height dimensions increase by 1 because Matlab labels the upper left element of a matrix as (1,1), whereas by convention the upper left corner of an image has the coordinates (0,0). The Canny Edge Detector, included in the Matlab Image Processing Toolbox performs edge detection on grayscale images. A grayscale image is of the dimensions $W \times H \times 1$, where W is the width and H is the height. There is only one colour channel. The intensity (ranging from 0 - Black to 255 - White) value of the corresponding pixel is stored in each element of the $W \times H$ matrix. To preserve the information contained separately in each of the three colour channels of the images they are converted into grayscale images. This is done using the Matlab command `rgb2gray()`.

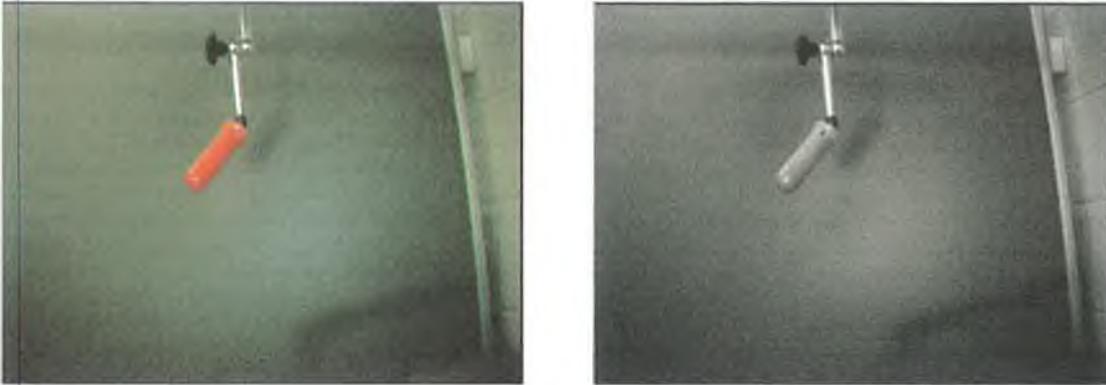


Fig 4.5 – The original RGB image on left and it's conversion to greyscale on right

This operation is performed and contained in the script `pack_cropconvert.m` (see Appendix C). Incidentally, Fig 4.6 contains a conversion to greyscale using only the Red channel of the original image. As can be seen the teat is much more clearly isolated than in Fig 4.5, unfortunately real cow teats are not generally bright red in colour. Although some cows do have teats of colours other than black and white, not all cows do and for this reason it was decided to always include all colour channels in the conversion. The grey-level intensities are an average of the intensities contains in the three colour channels.

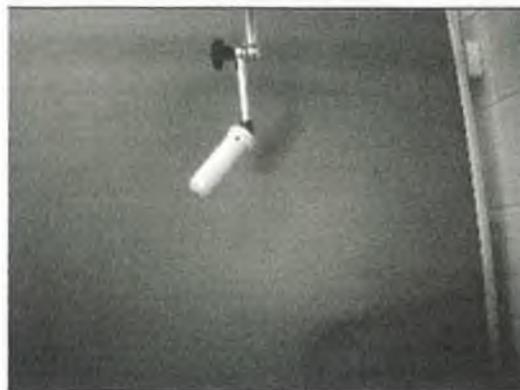


Fig 4.6 – Red Channel of RGB image

4.2.3 - Crop images to Region of Interest

In determining the teat angulations each teat is processed individually. The images are therefore cropped to a region of interest (ROI) containing the teat in question. The ROI is determined based on the pixel coordinates of the teats in the images (t_{uv}), previously

found from the 3D teat coordinates. The crop size and location is chosen so that the entire teat is ensured to be contained in the cropped image. The pixel coordinate t_{uv} corresponds to the bottom of the teat; the centre of the ROI will therefore be above this point. The vertical distance from the centre of the ROI to t_{uv} is half the chosen height of the ROI. The horizontal location of the centre of the ROI is the same as that of t_{uv} . The size of the ROI is chosen based on the average size of a cow teat and the distance of the cameras from the udder. In laboratory tests using dummy teats and the cameras approximately 0.7 metres from the centre of where the udder would be, the necessary ROI was found to be [160, 140]. Since there is no knowledge about the teat angulations the ROI must be big enough to encompass all possible teat angles. The horizontal size of the ROI accommodates teat angles of $\pm 50^\circ$ in the image plane. The vertical size accommodates the teat when it has an angle of 0° in the plane orthogonal to the image plane (i.e. straight down). The image cropping is performed using the Matlab function `imcrop()`, it is implemented by the function `pack_cropconvert.m` (see Appendix C).

An example of this image cropping is seen in *Fig 4.7*.

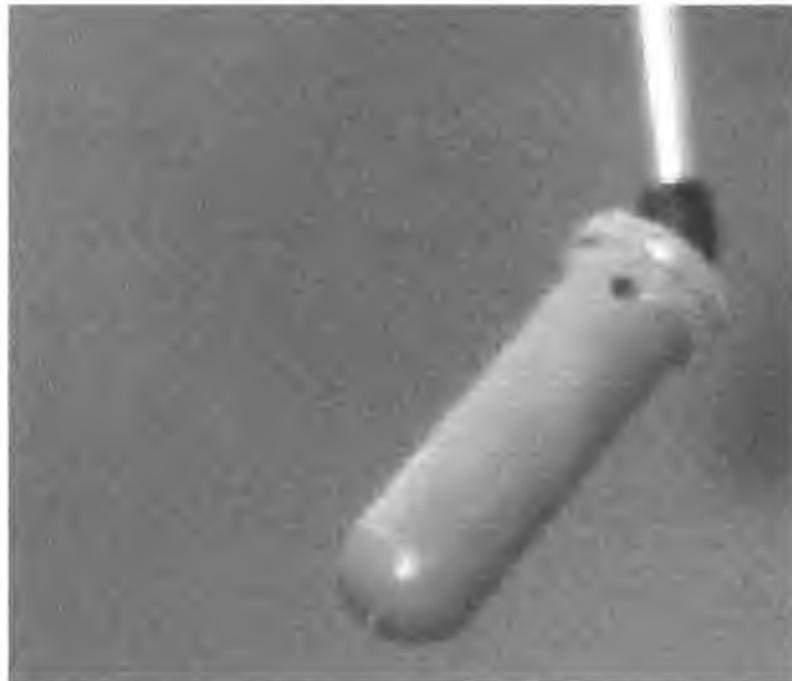


Fig 4.7 – Cropped Region of Interest

The next stage is to isolate edges in the images that could represent the sides of the teat. It may be desirable to alter the properties of the cropped image before this, so that the edges stand out better. Possible adjustments are brightness, contrast, gamma, and sharpness. The images acquired during testing did not require this tweaking to allow the edges to be detected. This is due to the controlled background and lighting conditions.

4.2.4 - Isolate the edges in the cropped images

Fig 4.8 shows the edges detected in the cropped image using a Canny Edge Detector. The algorithm of this Edge Detector is described in Chapter 2.2.1. It is implemented in the *Matlab Image Processing Toolbox* [45]. The function is called as follows:

```
edge_image = edge(input_image,'canny',threshold,sigma)
```

where 'input_image' is the input image (in this case the left cropped image) and 'sigma' is the standard deviation of the Gaussian smoothing filter (a stage in the algorithm to remove image noise). The term 'threshold' is a 1x2 matrix that defines what is considered as a weak edge or a strong edge. By setting it to a null matrix Matlab automatically determines suitable threshold values. The output image ('edge_image') is seen in Fig 4.8. To achieve this image sigma was set at 1.3 and the threshold was set to $[0, 0]^T$.



Fig 4.8 - Edges detecting using Canny Edge Detector

4.2.5 - Find Straight lines in Edge Image

The outline of the teat is easily seen in the edge image of Fig 4.8. However, there is unwanted edge information that makes isolating the edges representing the sides of the teats difficult without prior knowledge of the teat outline. To isolate the sides of the teat in the image straight lines will be fitted to the segments of edge data that approximate straight lines. This process is called 'Line Detection' and is carried out using the Hough Transform, as discussed in Chapter 2.2.2. The function `findlines.m` (see Appendix C) is called to detect lines in the edge image. The following three functions are called within this script, the code contained in these was obtained from *Digital Image Processing Using Matlab* [46]

```
pack_hough.m  
pack_houghpeaks.m  
pack_houghlines.m
```

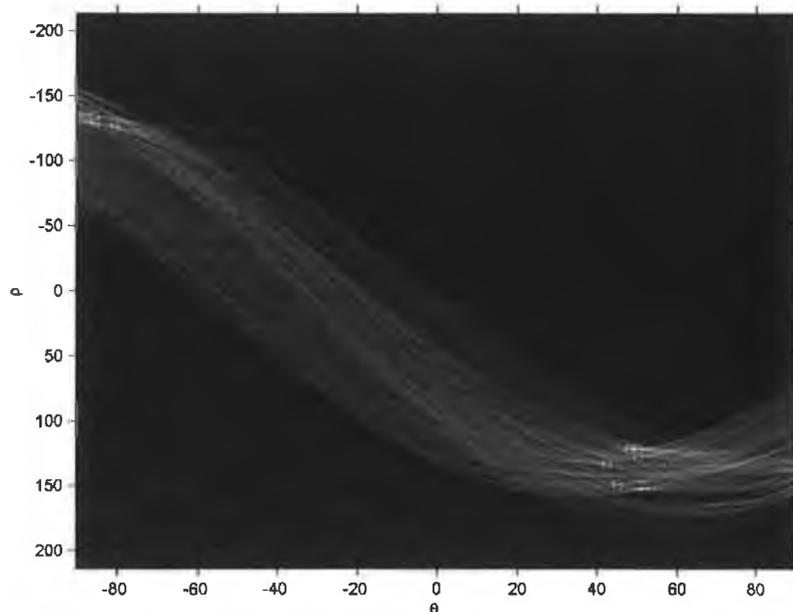


Fig 4.9 – Hough Transform Display with peaks selected

The Hough transform of the left cropped image Fig 4.8 is displayed in Fig 4.9. Each pixel in the image has been transformed and is represented by a curve traversing all the possible values of θ and ρ . The peaks of the transform are formed where these curves

overlap. Overlapping occurs when pixels in the image are collinear. The parameter values, θ and ρ , at the location of the peaks, form the equation of an infinite line in its normal representation:

(Equation 2.50)
$$x \cos \theta + y \sin \theta = \rho$$

The number of peaks to find is a user chosen input value. Peaks of greater intensity are given preference by the peak selecting program; these correspond to the longest and strongest lines in the edge image. In the example shown in *Fig 4.9* the number of peaks to find is set to 10, this is to ensure that the edges representing sides of the teats are still selected in the event of there being stronger edges, other than the sides of the teats, in the image.



7.10 – Detected Lines segmented to fit with original left and right edge images

The 10 returned peaks for each image represent 10 line equations in each image. These line equations are now matched to line segments in the edge image. This process, as well as the peak selection process is described in detail in Chapter 2.2.2. The matched line segments for both the left and the right images are seen in *Fig 4.10*. A peak qualifies as a line segment based on segment length and edge pixel correlation. All 10 peaks do not necessarily qualify, as seen in the right image. The criteria for a successful line match are not strict in these examples. This is to allow for the fact that the teats sides may not form

a perfectly straight and strong edge. There may be gaps in the edge that is detected. By allowing less certain matches it ensures that the sides of the teats are included. Fig 4.11 shows the matched line segments for the same edge images using stricter match criteria.

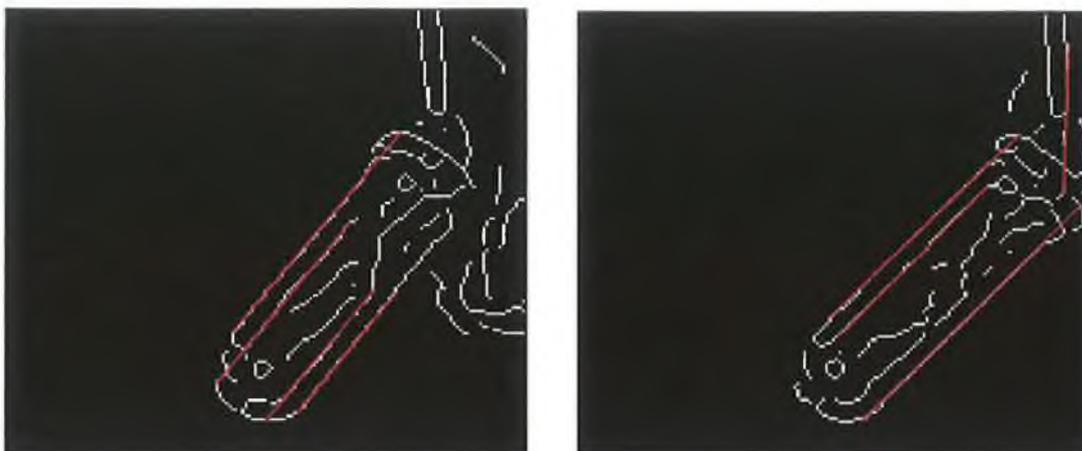


Fig 4.11 – Left and Right edges images with matched lines overlaid, strict matching criteria

4.2.6 - Selecting Lines corresponding to sides of Teat

The Hough transform line detector finds numerous matches for lines in the edge images, as seen in Fig 4.10. The next stage is to determine which lines correspond to the sides of the teat. This is implemented using a process of elimination: all the lines that couldn't correspond to the sides of the teats are filtered out until a pair of lines remains. These lines are then checked to ensure that they could represent the sides of the teats. This process is carried out by the functions contained in `pack_linefilter.m` (See Appendix C). The control function `pack_linefilter` is called in the following manner:

```
[returnline_left,returnline_right]= pack_linefilter(lines_left,  
                                                    lines_right,cropdata,cropsizes)
```

'lines_left' and 'lines_right' contain the end-points of the matched line segments found using the Hough line detector. 'cropdata' contains the pixel location of the teat in the image, upon which the crop region of interest is based, this point is known as the 'croppoint'. 'cropsizes' contains the dimensions of the crop region. The above return parameters 'returnline_left' and 'returnline_right' contain the equations of the lines

representing the sides of the teat. These equations are in the form $ax + by + c = 0$. The script `pack_linefilter.m` contains various sub-routines that can be called in the process of eliminating the lines unlikely to correspond to the sides of the teats. These functions will now be listed and briefly described. For more detail regarding the operation of these functions refer to the comments contained within the code. See Appendix C for details.

4.2.6.a - Line Selection Utilities

4.2.6.a.1 - Equationcal

```
[line_length,midpoint,midpoint_distance,lines_equations,slope,x_value_at_ycrop]=
    equationcal(lines_data,localcroppoint)
```

This function returns: the equations of all of the lines in the form $ax + by + c = 0$, the length of each line segment, the midpoint of each line segment and the absolute distance of the midpoint from the croppoint as well as the horizontal displacement of the midpoint from the croppoint. The slope of each line is also returned. The equations of the lines are evaluated using the point slope formula:

$$y - y_1 = m(x - x_1) \quad (4.18)$$

where (x_1, y_1) is a point on the line (either of the endpoints) and m is the slope. The slope is calculated using the 2-point slope formula:

$$m = \frac{(y_2 - y_1)}{(x_2 - x_1)} \quad (4.19)$$

As the lines approach a vertical orientation the slope approaches infinity. In the event of this the function output 'slope' is flagged as infinite and the equation of the corresponding line will be of the form:

$$ax + c = 0 \quad (4.20)$$

4.2.6.a.2 - Setsideofcropmatrix

```
sideofcropmatrix = setsideofcropmatrix(lines_data,localcroppoint,x_value_at_ycrop)
```

This function returns a matrix stating which side each line is on relative to the croppoint (as the image is viewed). The matrix contains 1 element for each line, if it has a value of “-1” it is on the left side, if “1”, it on the right side. This is achieved by comparing the x value of the line at the y value of the croppoint.

4.2.6.a.3 - Perpendistances

```
[normal_distance,max_norm_distance,normal_crop_distance]= perpendistances(lines_data,  
midpoint,lines_equations,localcroppoint,slope)
```

The purpose of this function is to determine a value that represents how far apart any two line segments are. It is important to note that the vast majority of the lines would intersect if they were extended. For this reason it is difficult to evaluate the notion of separation between line segments. To achieve it, the function finds the perpendicular distance from the midpoint of one line segment to each of every other detected line. This process is then repeated for each line segment. The steps to find the perpendicular distances are:

4.2.6.a.3.1 - Determine Slope of Normal Line

A line which is perpendicular to the line segment in question will have a slope that is the negative reciprocal of the slope of the segment This is governed by the fact that The product of the slopes of two perpendicular lines is equal to -1.

4.2.6.a.3.2 - Determine Equation of Normal Line

The normal line goes through the mid-point of the line segment and has a negative inverse slope. The equation is found using the Point – Slope formula (Equation 4.18)

4.2.6.a.3.3 - Point of Intersection

The normal distance is the distance between the midpoint of the line segment and the point of intersection of the normal line with another line segment. This point lies on both the 'other' line segment and the normal line and therefore must satisfy both equations of lines. This gives a system of two equations, with two unknowns, that can be solved simultaneously to find the point of intersection. (See Appendix A.2.1.b)

4.2.6.a.3.4 - Determine Perpendicular Distance

The distance between the line segment midpoint and point of normal intersection is determined using the distance formula:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (4.21)$$

The same method as outlined in the previous steps is used to determine the distance between the midpoints of the line segments and the crop point.

4.2.6.a.4 - Inoutmatrix

The 'inoutmatrix' is a singular row matrix with one column for each line detected by the Hough Line Detector. Each element of the matrix is set to either '1' or '0' and each element corresponds to a particular line. The matrix starts out with all of its elements set to '1'; this means that every line could possibly be the lines of interest. As the program progresses, lines established as not being possible matches have their corresponding 'inoutmatrix' variable set to zero. This means they will no longer be considered by the filter functions as a possible match. Once there are only two non-zero elements remaining in the 'inoutmatrix' there are only two lines remaining and the program exits. If the filtering has been successful then the returned pair of lines will match the sides of the teat. Each filter accepts the 'inoutmatrix' as an argument and will return a modified 'inoutmatrix' with elements set to zero for lines that have been eliminated.

4.2.6.a.5 - Minimise Midpoint to Croppoint

```
function [inoutmatrix] = filter_min_midpoint(inoutmatrix,lines_data,midpoint_distance,  
                                             mean_midpoint_distance)
```

This function is designed to eliminate lines whose midpoints are a large distance from the crop point. This is achieved by calculating the distance from the midpoint of each line to the local crop point and comparing it to the average distance for all of the lines. If the distance for a particular line is greater than that of the average it is eliminated, otherwise it is kept.

4.2.6.a.6 - Remove lines too close to croppoint

```
function [inoutmatrix] = filter_normfromcrop(inoutmatrix,lines_data,normal_crop_distance,  
                                             input_maxdistance)
```

This function filters out lines that are ‘too close’ to the crop point of the teat based on the perpendicular distance from midpoint to croppoint. The distances are compared to a hard coded value ‘input_maxdistance’, which is an argument of the function. The other arguments include the distance of each line to the croppoint, the equations of the lines and the ‘inoutmatrix’. If the distance is less than ‘input_maxdistance’ then the corresponding term of the ‘inoutmatrix’ is set to zero.

4.2.6.a.7 - Remove lines too far from croppoint

```
function [inoutmatrix] = filter_farfromcrop(inoutmatrix,lines_data,normal_crop_distance)
```

This filter operates in much the same way as the *Remove lines too close to croppoint* filter, except that it eliminates lines that have a perpendicular distance greater than the hard coded value.

4.2.6.a.8 - Maximise midpoint perpendicular distances

```
function [inoutmatrix] = filter_maxnormpair(inoutmatrix,lines_data,normal_distance)
```

This function finds lines that have a large midpoint perpendicular distance from at least one other line segment. At the later stages of the filtering process it is expected that only lines in close proximity to the sides of the teats, and of similar slope to the sides, are remaining unfiltered. If there are numerous line possibilities that are roughly parallel to the central axis of the teat, and are all enclosed by the extremities of the teat, then it follows that the two lines with the largest separation distance will be either side of the centre axis and will in fact lie upon the sides of the teat. This filter finds this pair of lines by establishing the maximum separation distance between any of the unfiltered lines, then eliminating any lines that do not have another line this minimum distance away from it (\pm tolerance (hard coded)). It is not necessarily the case that only two lines will be returned, due to the tolerance. This tolerance is a necessity of the programming implementation. Also, it is not guaranteed that should two lines be returned they are on opposite sides of the central axis of the teat.

4.2.6.a.9 - Maximise midpoint perpendicular distances based on left/right of croppoint

```
function [inoutmatrix] = filter_maxnormpairleftright(inoutmatrix,lines_data,
normal_distance,normal_crop_distance,sideofcropmatrix)
```

This function overcomes the shortcoming of the previous filter. By sorting the lines into groups based on which side of the croppoint they reside it guarantees that a returned pair of lines will be on opposite sides of the teats central axis. It also guarantees to return only two lines as its programming implementation no longer requires the tolerance.

4.2.6.a.10 - Is line within average slope of all lines?

```
function [inoutmatrix] = filter_avg_slope(lines_data,lines_equations,mean_slope)
```

This filter determines the difference between the slope of a line segment and the average slope of all the line segments and compares it to a hard coded constant. If the difference is too big the line is eliminated. Another way to implement the slope constraint would be to iterate this comparison and recalculate the average slope of the remaining lines only.

As the iteration progresses the tolerance of acceptance, based on the slope difference, could be tightened.

4.2.6.a.11 - Is length within mean requirements?

```
function [inoutmatrix] = filter_avg_length(lines_data,line_length,mean_length)
```

This filter operates in the same manner as the previous average slope filter to eliminate line segments that are of length less than that of the mean length. Again, this filter can be iterated, with the mean length being recalculated at the start of each iteration.

4.2.6.a.12 - Least Point Distance

```
function [inoutmatrix] = filter_leastpointdistance(inoutmatrix,lines_data,  
                                                  croppoint,input_maxdistance)
```

The Least Point Distance filter removes line segments not having at least one endpoint within a minimum distance from the crop point. Therefore, using this filter gives preference to lines that finish near to the end of the teat. The filter works by calculating the absolute distance between each endpoint of each line segment. The 'Least Point Distance' for each line is the lesser of the two distances. If this distance is greater than the maximum distance allowed (as determined by the function input argument 'input_maxdistance') then the line is eliminated from the 'inoutmatrix'.

4.2.6.b - Application of Line Filters

The line filter functions are available to be used in any order or in different combinations to achieve the best results. The best combinations will depend on the previous stage of the algorithm, the Hough line detection. Depending on the type of lines that are being detected, different filter combinations will be more suitable. The best combination for the conditions of the laboratory testing was found to be:

```
filter_farfromcrop  
filter_leastpointdistance  
filter_maxnormpairleftright
```

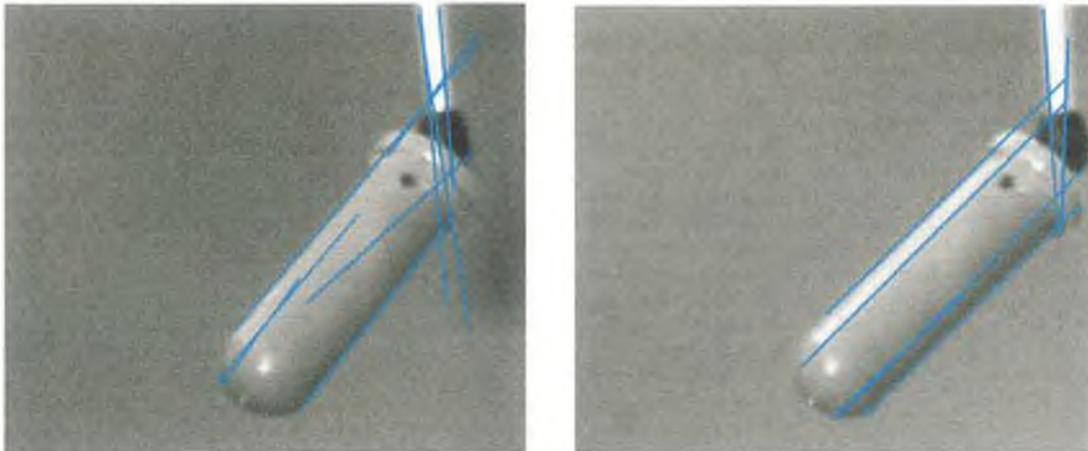


Fig 4.12 – All detected lines prior to filtering overlaid on left and right images

Fig 4.12 shows all of the detected lines in both the left and right images. This is before any filtering has been done, so at this point no lines have been eliminated and all are considered as possibilities for the eventual matches of the sides of the teat.

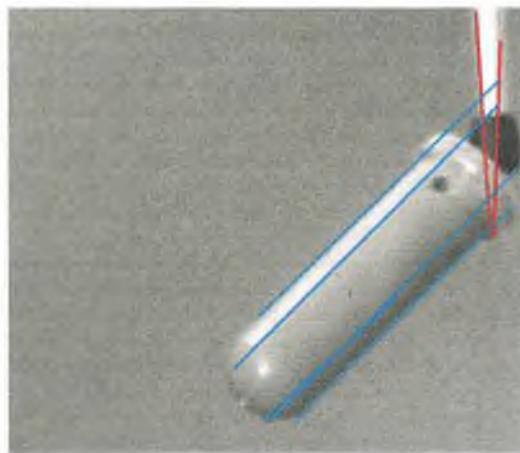


Fig 4.13 – After first filter, eliminated lines in red overlaid on left and right images

Fig 4.13 contains all remaining matches after the first filter (`filter_farfromcrop`) has been applied; these lines are shown in blue. The red lines are the segments that have been eliminated by the filter. These lines were discounted because their perpendicular distance to the croppoint is too great.

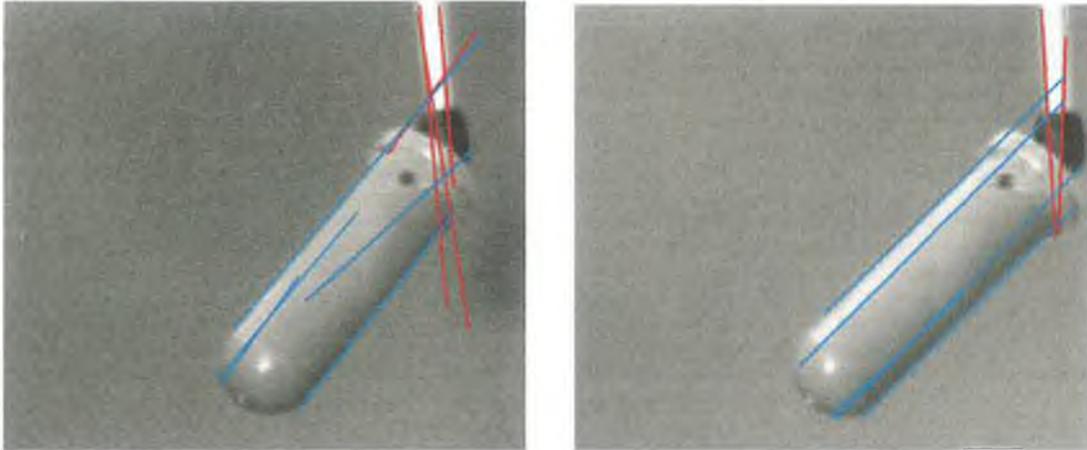


Fig 4.14 – After second filter, eliminated lines in red overlaid on left and right images

Fig 4.14 shows the remaining matches after the second filter (`filter_leastpointdistance`) has been applied. Only one more line has been eliminated in the left image and no further lines have been eliminated in the right image. The line is eliminated in the left image for the reason that the absolute distance from each of its endpoints to the croppoint is too great. This filtering stage would have eliminated more lines had the first filter not been applied.

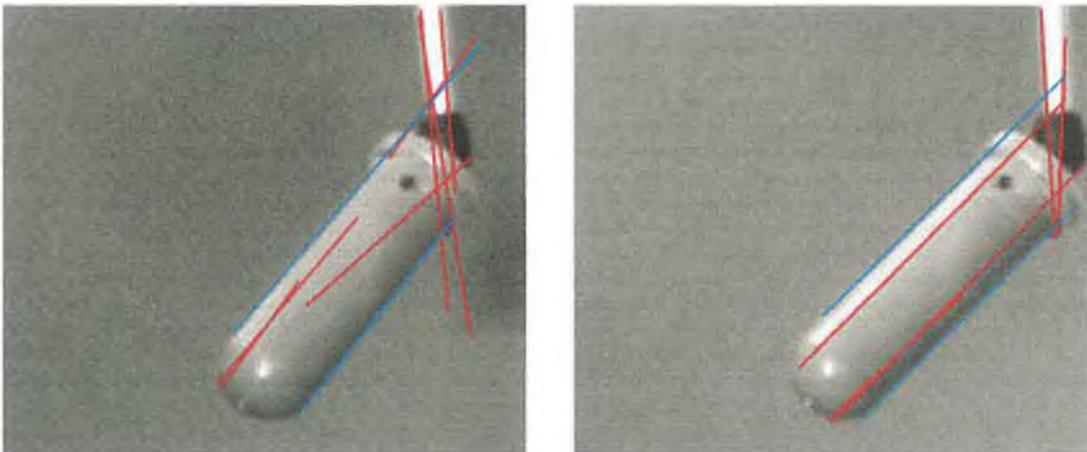


Fig 4.15 – After third filter, eliminated lines in red overlaid on left and right images

Once the third and final filter (`filter_maxnormpairleftright`) has been applied there are only two lines which have not been eliminated. This is seen in Fig 4.15. The filter chooses the pair of lines which have the largest perpendicular distance from each,

ensuring that they fall on opposite sides of the crop point. The filter also ensures that two lines are left remaining. These remaining lines represent the sides of the teat, the 'line filter' program returns the equations of these lines for use in the next stage of the algorithm. The returned line equations are:

Left:

$$-1.2572x - y + 247.5799 = 0 \quad (4.22)$$

$$-1.2131x - y + 193.3386 = 0 \quad (4.23)$$

Right:

$$-x - y + 223.658 = 0 \quad (4.24)$$

$$-0.9827x - y + 178.0634 = 0 \quad (4.25)$$

4.2.7 - Translate Side-lines

The sides of the teats are found in the cropped image containing the teat in isolation. Next, the line information is transferred to the full-size image. This is necessary as future steps in the algorithm, such as stereo reconstruction, rely on pixel coordinates based on the origin of the full-size image. The process is carried out by the function `pack_translatelines` (see Appendix C). The method employed is to determine the translation from the origin of the cropped image to the origin of the full-size image. The position of the croppoint is known in both the original and cropped images; from this a relationship between the origins is established. This relationship is then used to translate the lines of interest, by taking a single point on each line and translating it to the full-size image. The slopes of the lines remain the same; they are used in conjunction with the corresponding translated point to determine the equation of the line in the full-size image. This is done using the point slope formula.

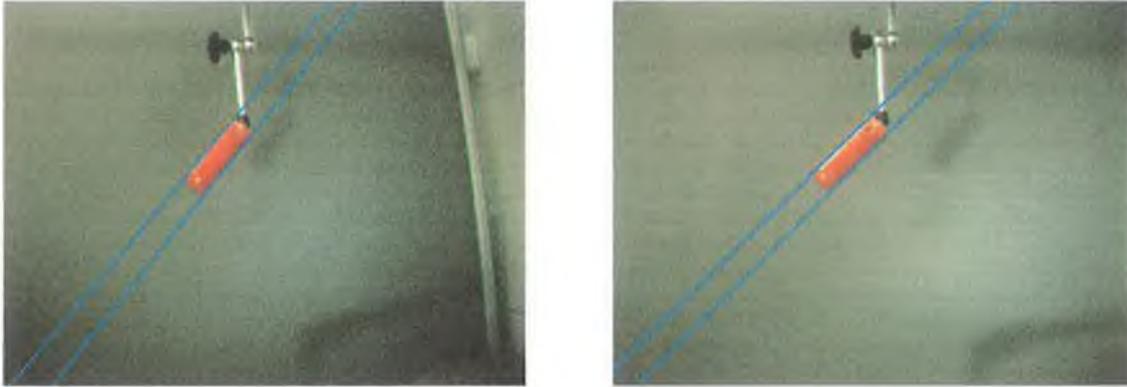


Fig 4.16 – Side-Lines of teats in full-size left and right images

As can be seen in Fig 4.16 the side-lines have been extended to the extremities of the image. The equations of the translated lines are:

Left:

$$-1.2572x - y + 550.4416 = 0 \quad (4.26)$$

$$-1.2131x + y + 489.3686 = 0 \quad (4.27)$$

Right:

$$-x - y + 504.658 = 0 \quad (4.28)$$

$$-0.9827x - y + 455.9835 = 0 \quad (4.29)$$

4.2.8 - Centreline of Teat

The central axis of the teat in each image lies along the bisecting line of the pair of lines in each image (these are the lines which represent the left and right side of the teat as the camera looks at it). The next stage of the algorithm is to calculate the equation of the bisecting line for both the left and right camera images.

The method for finding the bisector is as follows:

- Calculate the angle of interest between the two lines
- Determine the slope of a line that bisects the two lines
- Find the point of intersection of the two lines

- Find the equation of the line through the point of intersection with the previously determined bisecting slope. This line is the bisector (see Appendix A.2.1).

This process is carried out in the function `pack_epipolarcentreline` (see Appendix C).

In the case of this example the **point of intersection** was found to be:

(138.57, -119.16) in the left image

(281.31, -230.85) in the right image

The **slope** was found to be:

-1.2349 in the left image

-0.9913 in the right image

These give the line equations for the **bisectors** in the left and right images:

Left Bisector Line equation:

$$-1.2349x - y + 519.576 = 0 \quad (4.30)$$

Right Bisector Line equation:

$$-0.9913x - y + 480.2155 = 0 \quad (4.31)$$

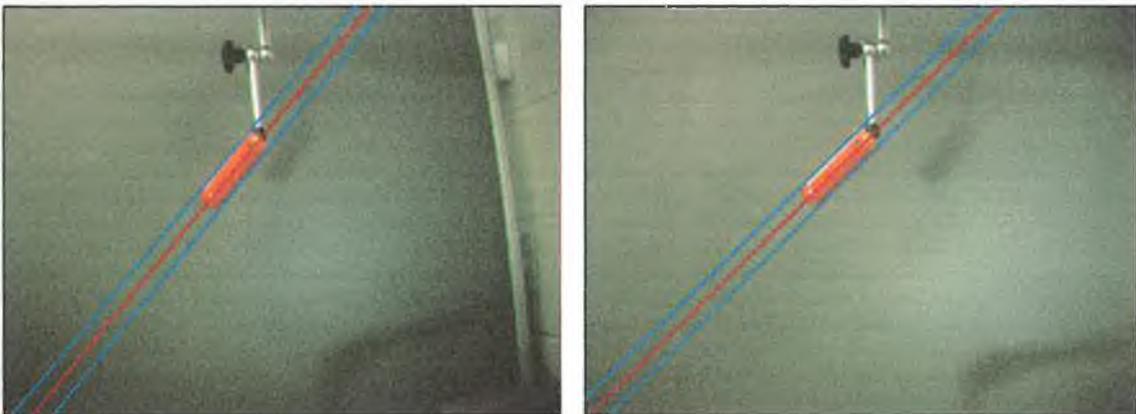


Fig 4.17 – Calculated bisector displayed in red for left and right images

There are specific situations where this algorithm will break down unless precautions are taken; when the lines representing the sides of the teat are parallel, when either of the two lines has an infinite slope (when the line is vertical) and when the resultant bisector is either vertical or horizontal.

4.2.8.a - Side Lines are Parallel

There is no point of intersection between two parallel lines, so strictly there is no bisector either. There is only a locus, the average of the two lines. In order to find this locus a different approach is taken: find a line parallel to the two side lines (of same slope), that is half way between them. This is implemented in the code by evaluating the 'x' component of a coordinate with 'y' component of 100 for each line. Let ' x_L ' equal to that of the left line and ' x_R ' to that of the right. The 'x' component of a point with a 'y' value of 100, for a line half way between the pair, is half the sum of left and right 'x' components; refer to this as ' x_{mid} '. The point (x_{mid} , 100) is therefore an element of the locus. The locus must also be of the same slope as the other two lines since they are parallel. The equation of the locus is then found using the point slope formula.

4.2.8.b - Sides of Infinite Slope

A vertical line has an undefined slope. If one of the lines is vertical the angle between the lines can not be evaluated in the normal manner as it relies on the slope of each line. This problem is avoided by detecting when one of the lines is vertical and finding the slope of the bisector based on the slope of the non-vertical line. The angle between this line and the x-axis (let's refer to it as ' φ ') is determined by getting the inverse tangent of the slope. If the value of φ is a negative angle it is converted to a positive one (by adding 360 if working in degrees). Now, the angle between the two sides is equal to $\frac{1}{2}(90 + \varphi)$ degrees.

4.2.8.c - Vertical/Horizontal Bisector

If the bisector of two lines results in either a vertical or a horizontal line then the coefficient of either the 'x' component or the 'y' component of the bisector line equation

will be zero respectively. Care therefore must be taken when applying formulae involving the slope. It should be noted that it is unlikely that a teat would be orientated in such a way as that the bisector of the two sides is horizontal, and, if there was such a teat, it is unlikely that it would be detected by the Ice Robotics System. A more likely scenario is that the resultant bisector should be a vertical line; this will mean that the 'y' coefficient of the line equation is zero. This gives an undefined slope for the bisector. The line information is valid; the software detects the fact that the line is vertical and changes the line equation format accordingly. The format is:

$$y + c = 0 \quad (4.32)$$

4.2.9 - Arbitrary Points on Centreline

The central axis of the teat has been isolated in both the left and the right images. The overall goal of the algorithm is to evaluate a vector describing the 3D angular information of the teat. To do this, two 3D points on the central axis of the teat must be known. The algorithm triangulates the positions of these two points using 3D reconstruction. A 3D point can be reconstructed if it can be seen in both images and if the pixel coordinates of the images 3D point are known for both the left and the right images. That is to say, a pair of corresponding pixel coordinates must be found in left and right images. This is known as a stereo pair. The algorithm produces the stereo pair by first taking a suitable pixel coordinate in the left image and then finding the corresponding pixel coordinate in the right image. This process is repeated so that two stereo pairs are found. The first step is to choose suitable pixel coordinates in the left image. To choose suitable points in the left image any two points that are elements of the central axis of the teat will do because it doesn't matter what length of 3D vector is calculated. Since the goal is to find the angle at which the teat is located, only the direction of the vector is important. Although any two points can be chosen to define the teat angulations care should be taken to ensure the accuracy and reliability of the 3D reconstruction.

- The points should not be close together. The closer the points are to each other the greater the amplification of any error incurred in reconstructing the 3D points.

- The points should not be close to the extremities of the left image to ensure that they have a corresponding point within the right image. The left and right images are captured using two different cameras with two different viewpoints, and because of this there will always be some regions in 3D space 'seen' by one camera that cannot be 'seen' by another. The term 'seen' does not refer to occlusion. It is not because the view is obstructed that the 2nd camera cannot 'see' the region; it is because it is outside of the cameras field of view.
- The points should be in the central region of the image to improve the accuracy of the matching algorithm. The matching algorithm (which is described later in this chapter in "*Corresponding Points in Right Image*") relies on the Epipolar geometry constraint, which in turn relies on the calibration of the cameras. The camera model and the calibration parameters are more accurate in the central region of the image due to lower lens distortion.

Fig 4.18 shows the two chosen points in the left image of the worked example. The points are highlighted by blue crosses. It can be seen that both of these points are on the centreline of the teat, displayed in green.

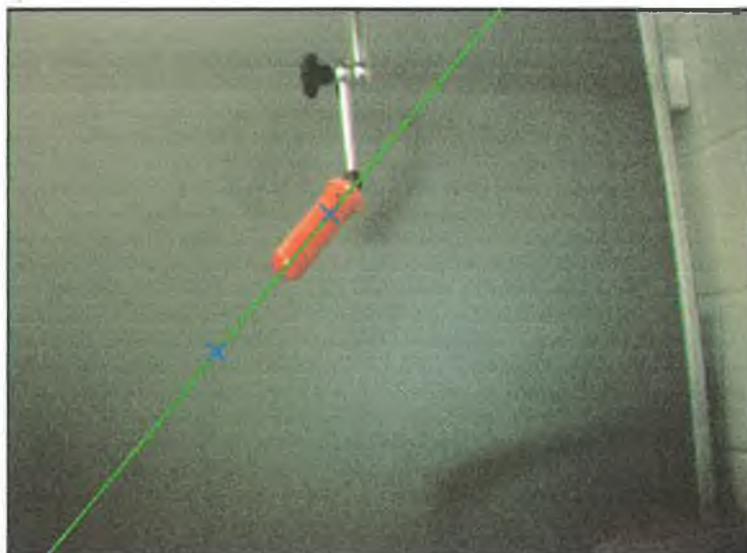


Fig 4.18 – Two chosen points in left image, shown in blue, along green centreline

The method that has been used to choose suitable points is quite simple and is implemented in the function `pack_epipolarcenterline` (see Appendix C). For each point a suitable 'y' component (or height in the image) is selected and the 'x' component is determined by evaluating the 'x' value of the bisector at this 'y' value. In the code the two 'y' components are set at $(\frac{1}{2}(\text{image height}) \pm 60)$. For a vertical teat this will give an absolute difference of 120 pixels between the two selected points. For 45° teat angles it will be 60 pixels, and for a horizontal teat it will be zero pixels. In the worked example the image height is 480 pixels so the 'y' component for the lower point in the image is $(480/2) + 60 = 300$. For the higher point in the image it is $(480/2) - 60 = 180$. Remember, the origin of the image coordinate frame is in the upper left corner, which is why the lower point has a larger 'y' component. By substituting these values into the previously found equation of the centreline the 'x' components of both points is found.

Equation of Bisector in left image :

$$\text{(Equation 4.30)} \quad -1.2349x - y + 519.576 = 0$$

$$\begin{aligned} \text{Low Point:} \quad & -1.2349x - 300 + 519.576 = 0 \\ & \Rightarrow x = 177.81 \end{aligned} \quad (4.33)$$

$$\begin{aligned} \text{High Point:} \quad & -1.2349x - 180 + 519.576 = 0 \\ & \Rightarrow x = 274.98 \end{aligned} \quad (4.34)$$

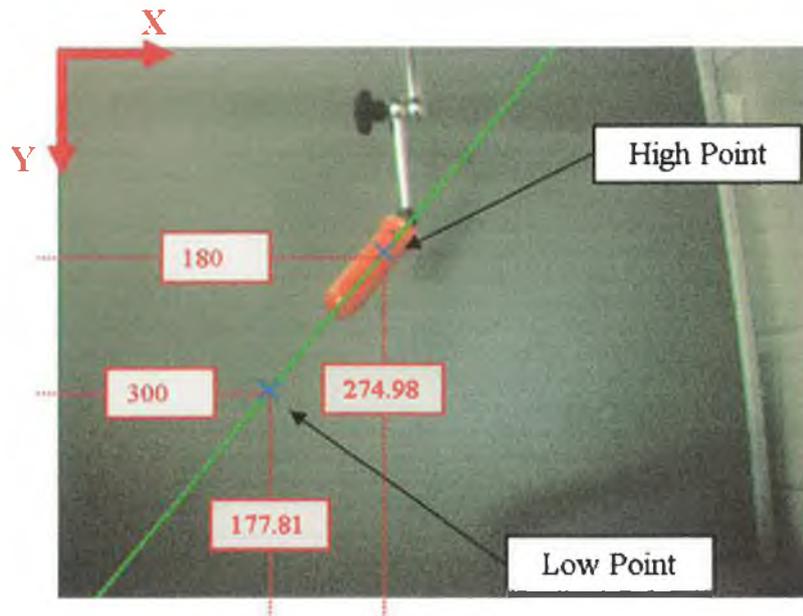


Fig 4.19 – Two chosen points in left image with coordinate values illustrated

4.2.10 - Corresponding Points in Right Image

The next stage of the algorithm is to locate the corresponding pixel coordinates in the right image for the two points that we have chosen in the left image. The task of searching for these points is greatly simplified by the fact that they are known to lie upon the central axis of the teats; it is only necessary to search along the right bisector. It is possible to determine a pixel along the centreline that is a corresponding point thanks to the Epipolar constraint. The Epipolar constraint, as discussed in Chapter 2.1.2.a, describes a line in the right image to which a point corresponding to a point in the left image is confined. This means that if a single point is selected in the left image, a line can be determined, of which the corresponding point in the right image must be an element. This Epipolar line is calculated by multiplying the left image point by the Fundamental Matrix, as detailed in Chapter 2.1.2.b. The corresponding point is also constrained to another line, the centreline of the teat. The point of intersection of these lines is the only point which satisfies both of these constraints (unless the lines are collinear). Calculating the point of intersection of the Epipolar line and the Centreline will yield the corresponding point.

Epipolar Line:

Equation 2.23: $l' = Fx$

where l' is the Epipolar line in the right image of the point 'x' in the left image and F is the Fundamental matrix such that $x'^T Fx = 0$, where x' is the corresponding point in the right image of x .

High Point:

(The given F is calculated in Appendix B.1.1, Equation B.3)

$$l'_{\text{high}} = \begin{bmatrix} -1119338.229 & -8351817.605 & 4623811745.102 \\ 22080246.666 & -3842270.527 & -46853623547.006 \\ -5436382304.916 & 45624867005.477 & -219165010882.963 \end{bmatrix} \begin{bmatrix} 274.9832 \\ 180 \\ 1 \end{bmatrix} \quad (4.35)$$

$$\Rightarrow l'_{\text{high}} = 0.0678x - y + 156.6878 = 0 \quad (4.36)$$

Low Point:

$$l'_{\text{low}} = \begin{bmatrix} -1119338.229 & -8351817.605 & 4623811745.102 \\ 22080246.666 & -3842270.527 & -46853623547.006 \\ -5436382304.916 & 45624867005.477 & -219165010882.963 \end{bmatrix} \begin{bmatrix} 177.8092 \\ 300 \\ 1 \end{bmatrix} \quad (4.37)$$

$$\Rightarrow l'_{\text{low}} = 0.0435x - y + 283.6114 = 0 \quad (4.38)$$

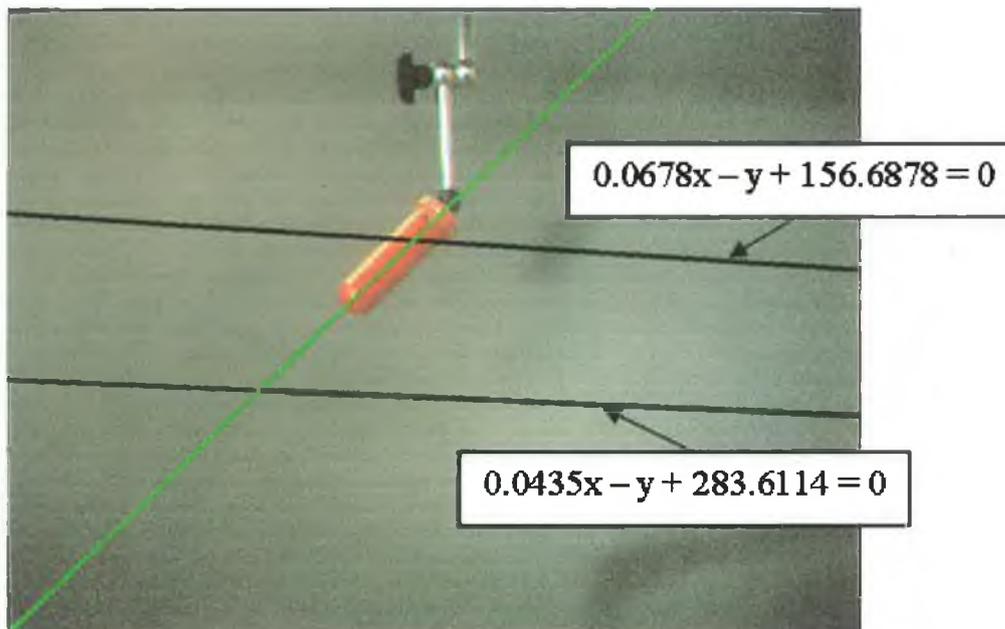


Fig 4.20 – Epipolar lines in right image corresponding to highpoint and lowpoint in left image

Fig 4.20 displays the right image of the teat with centreline and the two Epipolar lines overlaid. The point correspondence of the ‘High Point’ must lie upon the top Epipolar line and the ‘Low Point’ correspondence on the bottom line. It is clearly seen that both of the Epipolar lines intersect with the centreline, and because the corresponding points are also elements of the centreline the points of intersection are in fact the corresponding points. Using the same procedure as described previously in this chapter the points of intersection for this example are found to be:

High Point Intersect: (305.4655, 177.4041)

Low Point Intersect: (189.983, 291.8832)

This is illustrated in Fig 4.21.

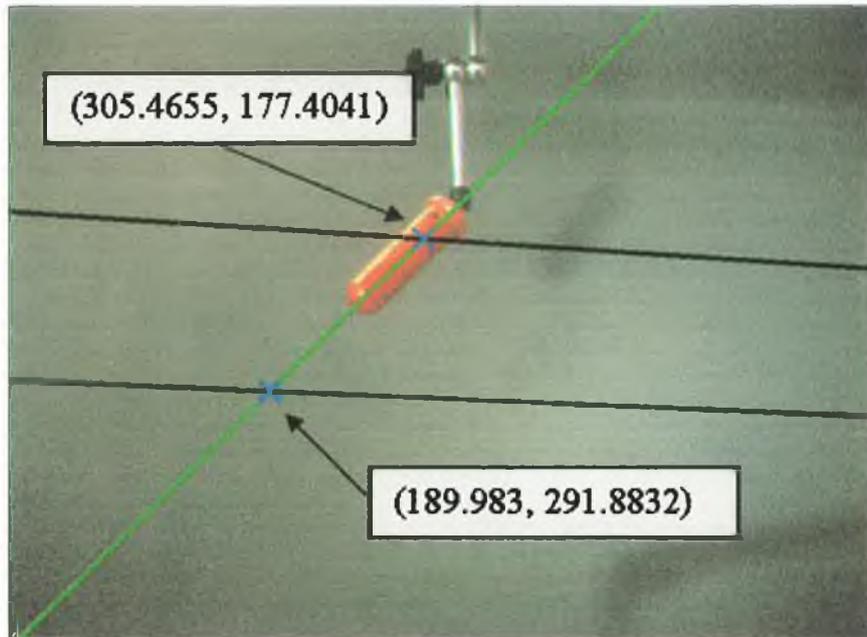


Fig 4.21 – Points of Intersection of Epipolar lines with Centreline

The unique point of intersection between an Epipolar line and the centreline only exists if the lines are not collinear and are not parallel. Since the corresponding point must be on the centreline, there must always be a point of intersection, so the lines cannot be parallel. However, it would be acceptable for there to be more than one point of intersection; this occurs in the event of collinear lines. In practise the centreline and the Epipolar line will never be collinear. Due to the configuration of the stereo camera rig the Epipolar lines produced in the right image by any point in the left image will always be close to horizontal. A teat would need to be angulated at close to 90° before the centreline becomes collinear with an Epipolar line, and, since the AMS is not intended to apply cups to teats angled at greater than 45° it is a reasonable assumption that the lines will never be collinear. Fig 4.22 contains selected points in the left image and their corresponding (colour coded) Epipolar lines in the right image. It is seen that the Epipolar lines are approximately horizontal in all cases.

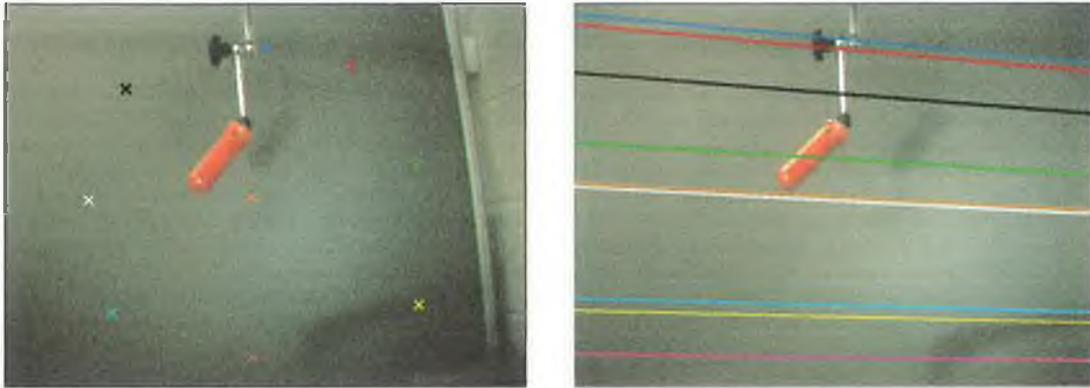


Fig 4.22 – Selected points in left image and corresponding Epipolar lines in right image

4.2.11 - Triangulating 3D Vector

The process of reconstructing a 3D point by triangulation from a pair of corresponding points is described in detail in Chapter 2.1.2.c. In this example the reconstruction has been carried out using the 'stereo_triangulation()' function contained in Caltech's *Camera Calibration Toolbox for Matlab*[®] [47]. The toolbox function is called in the Matlab script `calculate_angle.m` (see Appendix C). The arguments of the function are the point correspondences and the camera calibration parameters. The function returns the triangulated 3D coordinate with respect to the left camera's centre. The homogeneous image correspondence inputs are:

High Point:

$$x_{high} = \begin{bmatrix} 275 \\ 180 \\ 1 \end{bmatrix} \text{ and } x'_{high} = \begin{bmatrix} 305.5 \\ 177.4 \\ 1 \end{bmatrix}$$

Low Point:

$$x_{low} = \begin{bmatrix} 177.8 \\ 300 \\ 1 \end{bmatrix} \text{ and } x'_{low} = \begin{bmatrix} 190 \\ 291.9 \\ 1 \end{bmatrix}$$

These point correspondences triangulate to:

$$X_{high} = \begin{bmatrix} -24.61 \\ -28.34 \\ 632.21 \\ 1 \end{bmatrix}$$

$$X_{low} = \begin{bmatrix} -103.25 \\ 75.43 \\ 549.08 \\ 1 \end{bmatrix}$$

4.2.12 - Transform to World Reference Frame

The two 3D points found in the previous section are with respect to the coordinate frame of the left camera found during calibration (Appendix B.1.2). These two points contain all the information regarding the orientation of the teat, but in terms of how it is orientated relative to the Camera Coordinate Frame. The performance of this angle measurement algorithm is assessed by comparing it to a reference value determined manually. These measurements are with respect to a different coordinate frame: The World Reference frame. The 3D points must therefore be transformed to the World Reference Frame before comparisons can be easily made. The relationship between the Camera Coordinate Frame and the World Reference Frame is established in Appendix B.2.2 as a 4x4 rotation translation matrix. It evaluates to:

$$(Equation B.19) \quad {}^{cam}T_{World} = \begin{bmatrix} 0.7182 & 0.6954 & -0.0252 & 68.1967 \\ -0.6869 & 0.7045 & -0.1782 & 173.7885 \\ -0.1062 & 0.1453 & 0.9837 & -654.8465 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Pre-multiplying the 3D coordinates in the Camera Reference Frame by the above transformation will convert them to coordinates in terms of the World Reference Frame.

$$W_{high} = \begin{bmatrix} 0.7182 & 0.6954 & -0.0252 & 68.1967 \\ -0.6869 & 0.7045 & -0.1782 & 173.7885 \\ -0.1062 & 0.1453 & 0.9837 & -654.8465 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -24.61 \\ -28.34 \\ 632.21 \\ 1 \end{bmatrix} = \begin{bmatrix} 14.91 \\ 58.07 \\ -34.46 \\ 1 \end{bmatrix} \quad (4.39)$$

$$W_{low} = \begin{bmatrix} 0.7182 & 0.6954 & -0.0252 & 68.1967 \\ -0.6869 & 0.7045 & -0.1782 & 173.7885 \\ -0.1062 & 0.1453 & 0.9837 & -654.8465 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -103.25 \\ 75.43 \\ 549.08 \\ 1 \end{bmatrix} = \begin{bmatrix} 32.67 \\ 200.01 \\ -92.81 \\ 1 \end{bmatrix} \quad (4.40)$$

where W_{high} is 3D High Point coordinate and W_{low} is the Low Point coordinate, in the World Reference Frame.

The coordinates are now in the World reference frame, allowing direct comparisons to be made to the base measurements. However, as is seen in Appendix B.2.2, both the base measurements and the vision system estimates are converted into another Reference Frame, referred to as the Test Rig, or Rig, Reference Frame. The reason for this is to aid in the visualisation of the angle information; the physical boundaries and octagonal planes present on the test rig are ideal reference planes from which to describe the test angulations.

4.2.13 - World to Test Rig Reference Frame

The relationship between the Test Rig Reference Frame and the World Reference frame is established and described in Appendix B.2.2. It is described by the 4x4 rotation translation matrix:

$$(Equation B.20) \quad {}^{World}T_{Rig} = \begin{bmatrix} 0.7024 & -0.707 & 0.0519 & -6.0112 \\ -0.0585 & 0.0153 & 1.002 & 11.9959 \\ -0.708 & -0.7056 & -0.306 & -109.93 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Pre-multiplying the 3D coordinates in the World Reference Frame by the above transformation will convert them to coordinates in terms of the Test Rig Reference Frame.

$$TR_{high} = \begin{bmatrix} 0.7024 & -0.707 & 0.0519 & -6.0112 \\ -0.0585 & 0.0153 & 1.002 & 11.9959 \\ -0.708 & -0.7056 & -0.306 & -109.93 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 14.91 \\ 58.07 \\ -34.46 \\ 1 \end{bmatrix} = \begin{bmatrix} -38.38 \\ -22.46 \\ -160.4 \\ 1 \end{bmatrix} \quad (4.41)$$

$$TR_{low} = \begin{bmatrix} 0.7024 & -0.707 & 0.0519 & -6.0112 \\ -0.0585 & 0.0153 & 1.002 & 11.9959 \\ -0.708 & -0.7056 & -0.306 & -109.93 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 32.67 \\ 200.01 \\ -92.81 \\ 1 \end{bmatrix} = \begin{bmatrix} -129.29 \\ -79.68 \\ -271.34 \\ 1 \end{bmatrix} \quad (4.42)$$

where TR_{high} is 3D High Point coordinate and TR_{low} is the Low Point coordinate, in the Test Rig Reference Frame.

4.2.14 - Visualisation and Angle to Plane

The coordinates have been transformed to the Test Rig Reference Frame, seen in Fig 4.23. The view of the rig in this illustration is from the front, slightly above and to the left. The units of the axes are millimetres.

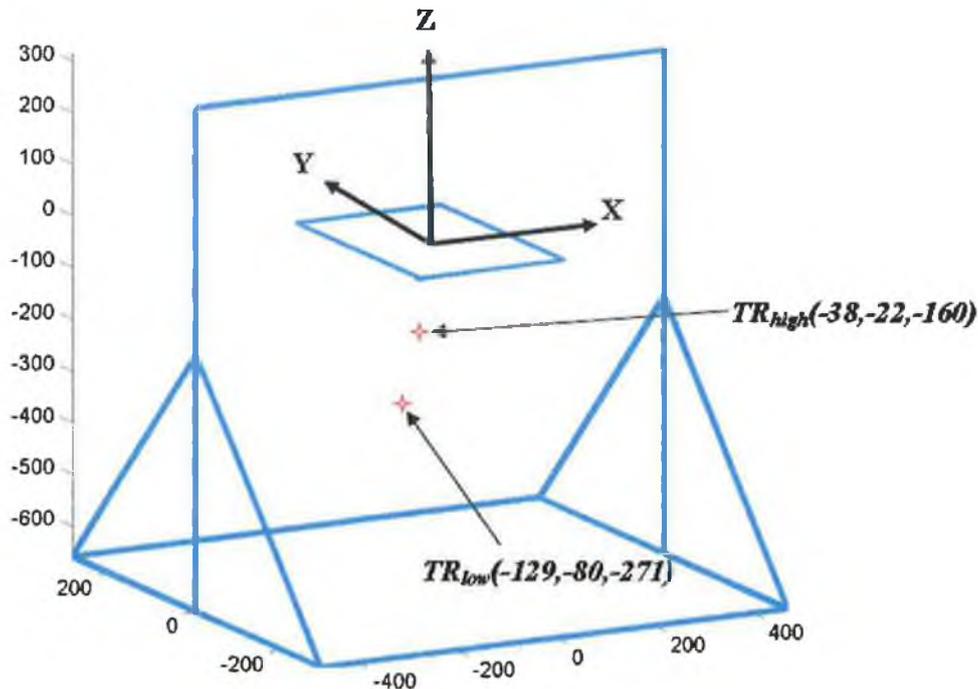


Fig 4.23 – Illustration of test Rig Coordinate frame, units in millimetres

The ‘High’ and ‘Low’ point can be plotted in the Test Rig Coordinate Frame and joined by a line. This line represents the centreline of the teat in 3D space. For the purposes of visualisation the orientation of the teat is described by two angles. One of the reference angles, θ_{YZ} , is with respect to the plane formed by the Y-axis and the Z-axis (YZ-Plane) of the Test Rig Coordinate Frame; the other angle, θ_{XZ} , is with respect to the plane formed by the X-axis and the Z-axis (XZ-Plane). This can also be seen in Fig 4.1 at the start of this chapter. Fig 4.24 shows the two points joined by a line in 3D space on the

left. The right side of Fig 4.24 contains a close up of the line and the reference planes YZ' and XZ'. The plane YZ' is parallel to the YZ plane, XZ' is parallel to the XZ plane. The intersection of these parallel planes forms a separate coordinate frame of a purely translational relationship to the Test Rig Coordinate Frame. The origin of this separate coordinate frame is the 'High' point. These visualisations were generated using the following scripts: `coordframe.m`, `angulate.m`, `display_angle.m`.

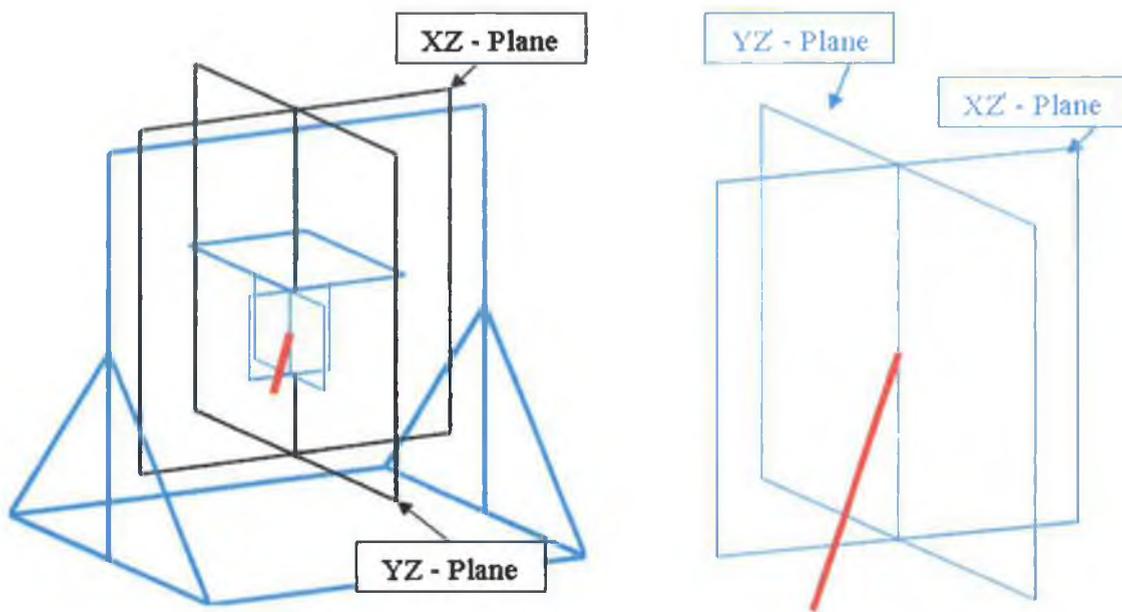


Fig 4.24 – Left: Angle Base Reference Planes; Right: Close up of test and individual reference planes

The 3D line is projected on to both of the 2D planes YZ' and XZ'. From the projection onto the XZ' plane the angle $\theta_{XZ'}$ can be measured (Fig 4.25 left); from the projection onto the YZ' plane $\theta_{YZ'}$ can be measured (Fig 4.25 right).

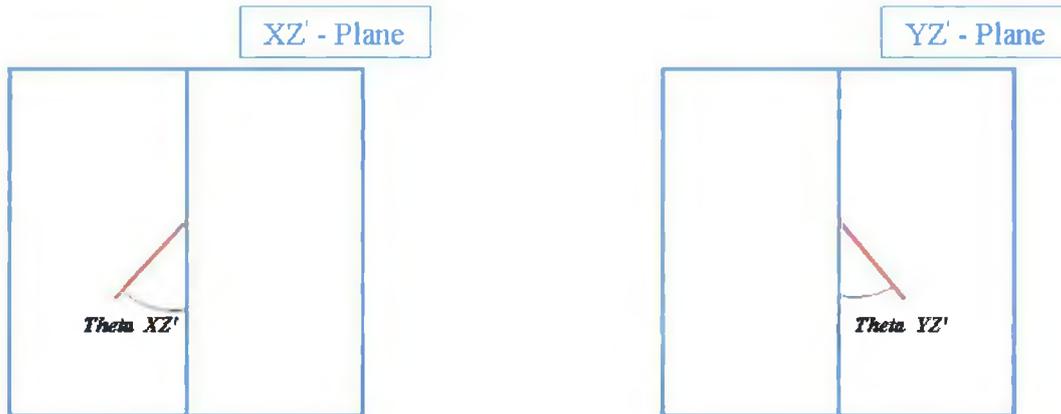


Fig 4.25 – Projections of line onto the parallel planes

The projection onto the planes is carried out in the Matlab script: `calculate_angle.m`. To project onto the YZ' plane the 'x' component of each of the point coordinates (the 'High' and the 'Low') is removed to give a 2D coordinate. Similarly, to project onto the XZ' plane the 'y' component of the point coordinate is removed. A third point is found in each projection to form a right-angled triangle such that the projected line is the hypotenuse, and the right angle being formed by the Z'-axis and a line parallel to the X'-axis. In Fig 4.26 the point (a) is the projection of the 'High Point'; (b) is the projection of the 'Low Point' and (c) is the third point.

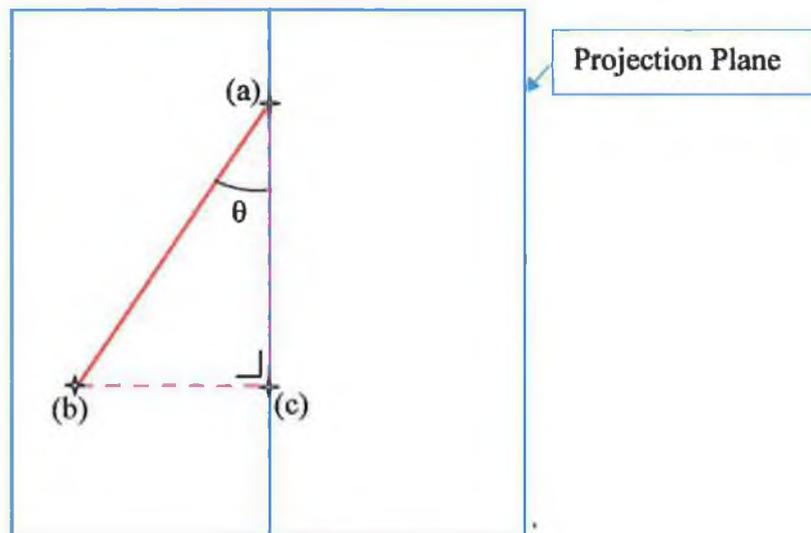


Fig 4.26 – Point (c) makes up a right angles triangle in the projection plane

In the worked example for Teat 11, the points that make up the triangles are found to be:

YZ' – Plane

a (-22.46,-160.4)

b (-79.68, -271.34)

c (-22.46, -271.34)

XZ' – Plane

a (-38.38,-160.4)

b (-129.29,-271.34)

c (-38.38, -271.34)

(Note: although these points are on the parallel planes their coordinates are given in terms of the Test Rig Coordinate Frame)

The third point (*c*) is made up from the other two points.

From Fig 4.26:
$$\cos \phi = \frac{|ac|}{|ab|} \quad (4.43)$$

$$\Rightarrow \phi = \cos^{-1} \left(\frac{|ac|}{|ab|} \right) \quad (4.44)$$

Where:

$$|ab| = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2} \quad (4.45)$$

$$|ac| = \sqrt{(a_1 - c_1)^2 + (a_2 - c_2)^2} \quad (4.46)$$

$$a = (a_1, a_2) \quad (4.47)$$

$$b = (b_1, b_2) \quad (4.48)$$

$$c = (c_1, c_2) \quad (4.49)$$

For the worked example the following is calculated:

YZ' – Plane

$$|ab| = 124.83$$

$$|ac| = 110.9$$

$$\text{Theta}_{YZ} = 27.3^\circ$$

XZ' – Plane

$$|ab| = 143.4$$

$$|ac| = 110.9$$

$$\text{Theta}_{XZ} = 39.3^\circ$$

The known accurate reference measurement (Chapter 5.8 Table 5.1) is -27.9° and -38.2° for Theta_{YZ} and Theta_{XZ} respectively.

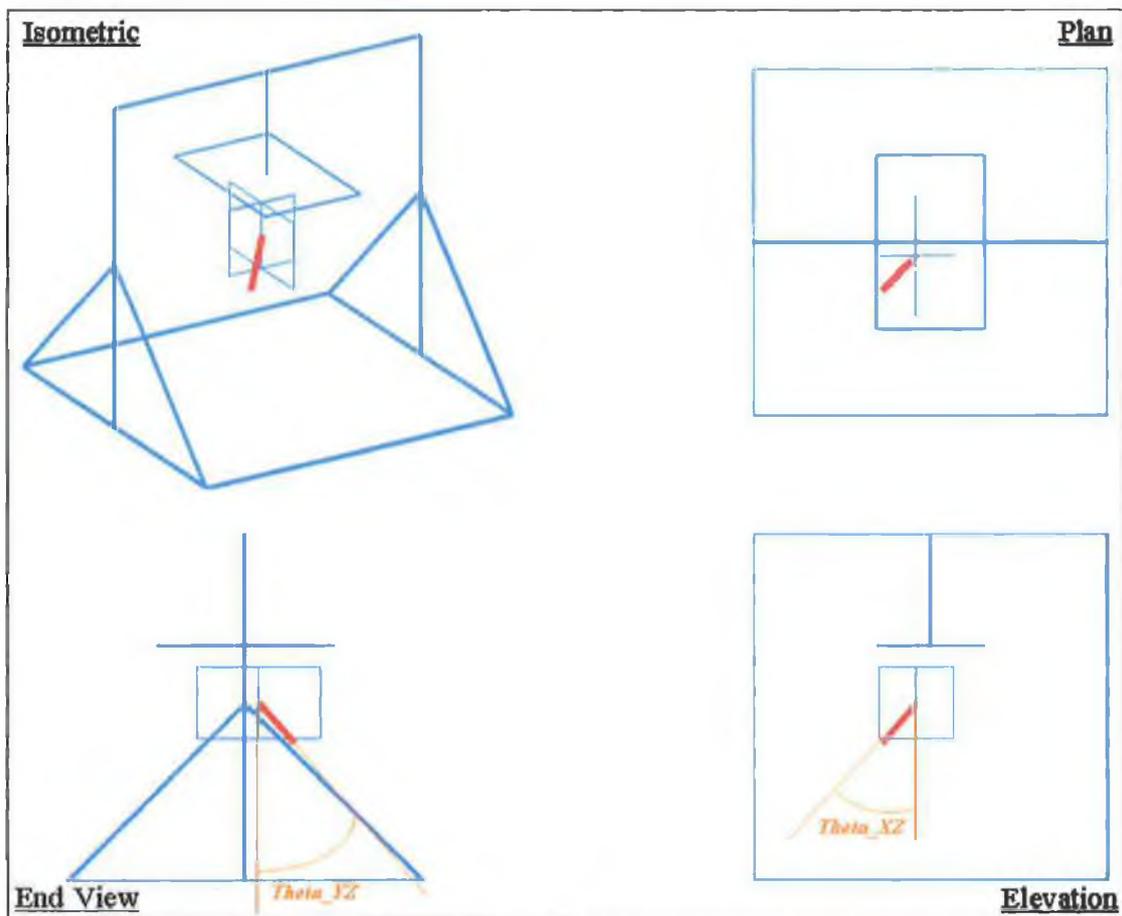


Fig 4.27 – Orthographic Projection: Different Views of Test 11 angulations measured visually

Chapter 5 – Angle Accuracy Test Results

The accuracy of the angle measurement algorithm is assessed by comparing angle estimates made with the vision system to manual measurements. This is achieved by capturing images of a single teat in a range of angulations and comparing values with those obtained using the Microscribe. Measurements taken with the Microscribe and angle estimates made using the angle algorithm must be converted to the same reference frame before direct comparisons can be made. This section describes how both sets of measurements are translated to the World Reference Frame (re-defined in this chapter) and from the World Reference Frame to the Test-Rig Reference Frame. This coordinate system is seen in Fig 5.1.

5.1 - Test setup

A single teat is located at the centre of the camera view in the centre of the target region. In Fig 5.1 the teat is located at approximately (0,0,-200), the camera is located at (0,-680,-200). Two points along the teat centre line are captured using the Microscribe for each teat orientation.

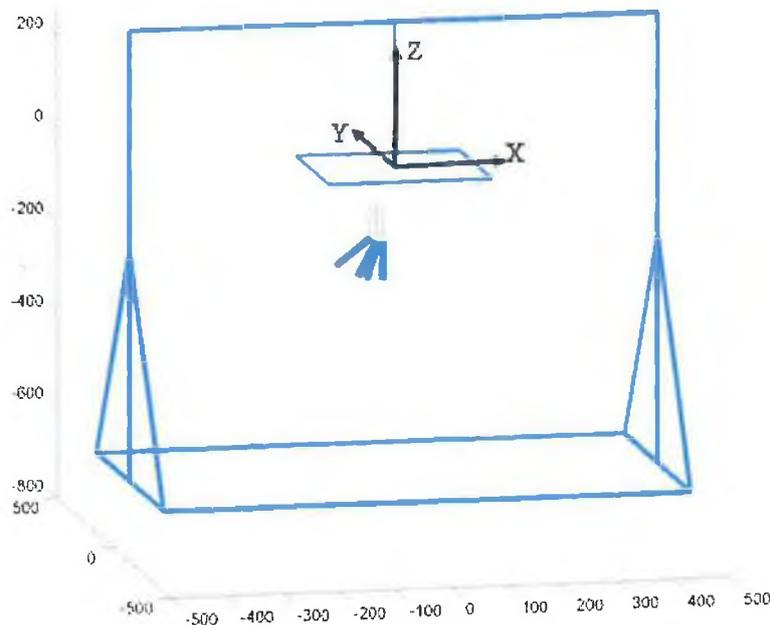


Fig 5.1 – Isometric view of test rig, 4 teat angulations in the XZ-plane, Test Rig Reference Plane shown

Fig 5.1 illustrates four angulations of the teat in the XZ plane; these are better seen in Fig 5.2.



Fig 5.2 – Teat angulations in the XY Plane

These four angulations, ranging from approximately 0° to 45° in increments of 15° , are repeated for different angulations in the YZ plane, seven in total. Fig 5.3 illustrates the range of angulations in 3D. Fig 5.4 is a cross-section of all the teat orientations and locations in testing. There are 28 test positions in total.

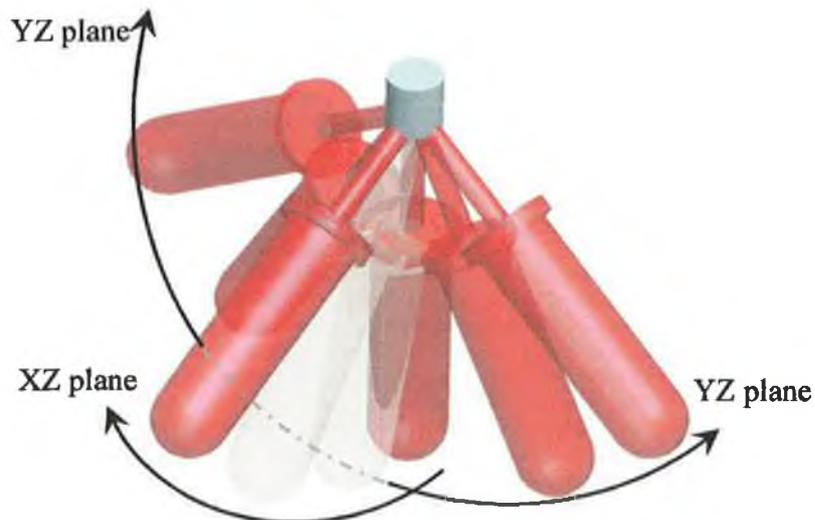


Fig 5.3 – Teat Angulations in XY and YZ planes

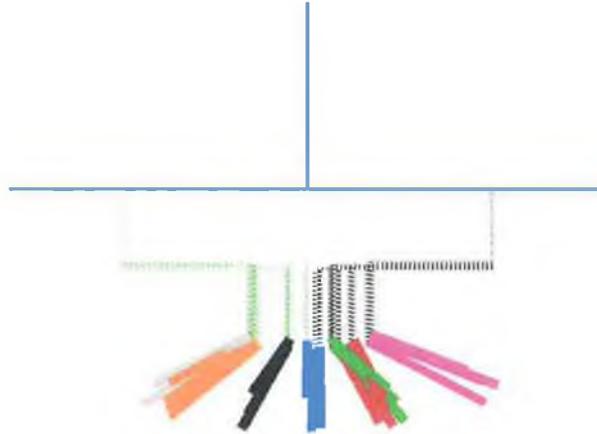


Fig 5.4 – Cross section of teat positions and angles for angle accuracy tests in YZ-Plane

The volume defined by all the teat angulations in the tests makes up half a hemisphere. All the angles in the XZ-plane are positive teat angled to the left). A negative angle in the YZ-plane corresponds to a teat angled towards the cameras. A positive angle corresponds to a teat that is angled away. There is no need to perform the tests on the opposite half of the half-hemisphere; this would be a repeat of the test angles mirrored through the YZ plane.

5.2 - Reference Plane Transformations

In Appendix B.2.2 the transformations are calculated that are required to convert the manual angle measurements of the Microscribe and the angle estimates of the vision system to the same reference frame. The transformation ${}^{\text{Microscribe}}T_{\text{Rig}}$ (Equation 5.1) converts the coordinate outputs of the Microscribe into coordinates of the Test Rig reference frame seen in Fig 5.1. ${}^{\text{Camera}}T_{\text{Rig}}$ (Equation 5.2) transforms the coordinates triangulated by the vision system to the same Test Rig reference frame.

$${}^{\text{Microscribe}}T_{\text{Rig}} = {}^{\text{Microscribe}}T_{\text{World}} {}^{\text{World}}T_{3\text{Teat}} {}^{3\text{Teat}}T_{\text{Rig}} \quad (5.1)$$

$${}^{\text{Camera}}T_{\text{Rig}} = {}^{\text{Camera}}T_{\text{World}} {}^{\text{World}}T_{3\text{Teat}} {}^{3\text{Teat}}T_{\text{Rig}} \quad (5.2)$$

The component transformations of Equations 5.1 and 5.2 are defined and evaluated in Appendix B.2.2. ${}^{\text{Microscribe}}T_{\text{Rig}}$ and ${}^{\text{Camera}}T_{\text{Rig}}$ are also evaluated, given here as Equations 5.3 and 5.4 respectively.

$${}^{\text{Microscribe}}T_{\text{Rig}} = \begin{bmatrix} 0.7024 & -0.7070 & 0.0519 & -6.0112 \\ -0.0585 & 0.0153 & 1.0002 & 11.9959 \\ -0.7080 & -0.7056 & -0.0306 & -109.9329 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5.3)$$

$${}^{\text{Camera}}T_{\text{Rig}} = \begin{bmatrix} 0.9846 & -0.0022 & 0.1594 & -114.9859 \\ -0.1588 & 0.1154 & 0.9826 & -644.3310 \\ -0.0206 & -0.9939 & 0.1134 & -260.7888 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5.4)$$

These two transformations are used to make direct comparisons between angles measured using the Microscribe and using the vision system.

5.3 - Teat Angle Test Results

The teat angle extraction algorithm was run on twenty eight stereo image pairs. These images contained views of a single teat; the teat angle varying in each image in accordance with Section 5.1.

5.3.1 – Effect of Input Variance on Performance

The output of the IceRobotics system was simulated in the collection of all results in this chapter. In Chapter 3.1 it was shown that the location of the teat end in both the left and right images is calculated from the teat coordinates outputted by the IceRobotics system. For the purpose of angle extraction the teat locations in the images have been simulated and passed as an input to the angle extraction algorithm. It was deemed reasonable that a location calculated from the IceRobotics system should reside within an area of 21 by 21 pixels situated at the teat end, as seen in Fig 5.5.

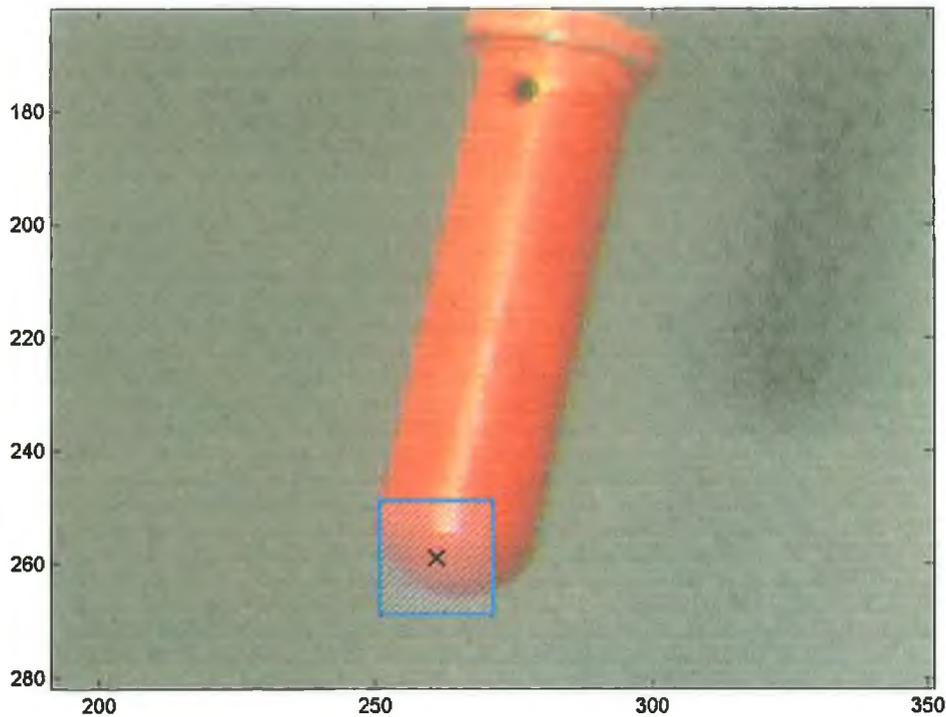


Fig 5.5 – Region in which IceRobotics System output is expected

The performance of the angle extraction algorithm varies depending on the estimation the teat end coordinates. In Fig 5.5 the black 'X' represents a location in the left image that, if it was passed to the algorithm, would allow it to perform optimally, where optimal performance is defined as correctly identifying the sides of the teat and producing teat angle estimates with the highest possible level of accuracy. The 21 by 21 pixel region is centred at this location. The angle extraction algorithm has been iterated with each of the 441 possible pixel locations within this region for three of the tests: Test#2, Test#11 and Test#23. The choice of these is discussed in Chapter 6.2.1.a. The results of these iterations are seen in Fig 5.7, Fig 5.8 and Fig 5.9. In these figures the plot on the left corresponds to Theta_YZ measurements, the plot on the right to Theta_XZ measurements. Each square of the colour grid represents an iteration of the algorithm with the pixel location of that square passed as the input. The colour of the square indicates the performance, where the associated absolute errors in the respective angle estimates correspond to the legend in Fig 5.6.

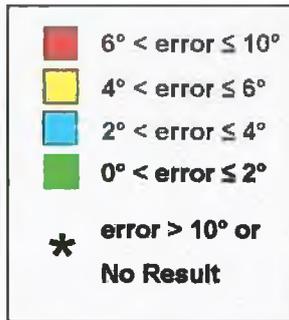


Fig 5.6 – Legend: Colours indicated absolute error associated with pixels in Figs 5.7 to 5.9

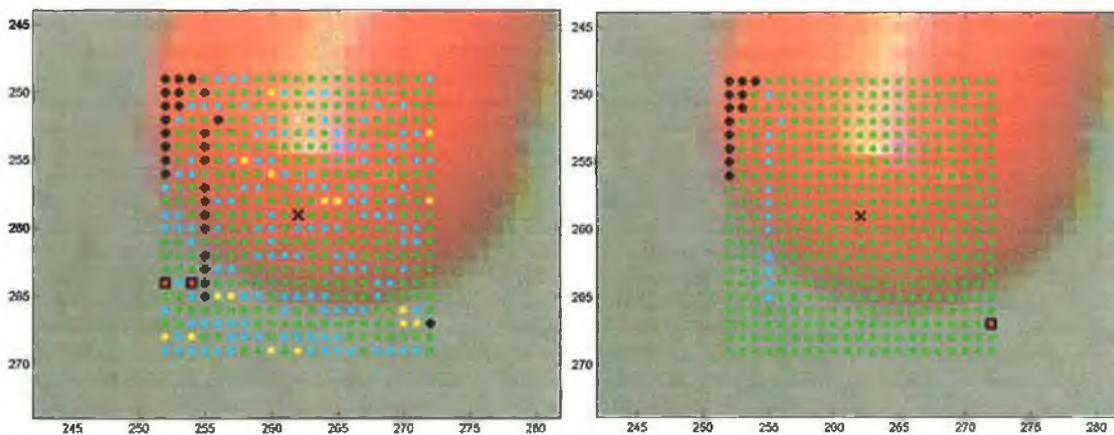


Fig 5.7 – Test#2 accuracies: Theta_YZ on left, Theta_XZ on right

In Fig 5.7 it is seen from the number of green and blue squares in the left image that the majority of the iterations produced Theta_YZ estimates with an error $\leq \pm 4^\circ$. The vast majority of the Theta_XZ estimates have an error $\leq \pm 2^\circ$.

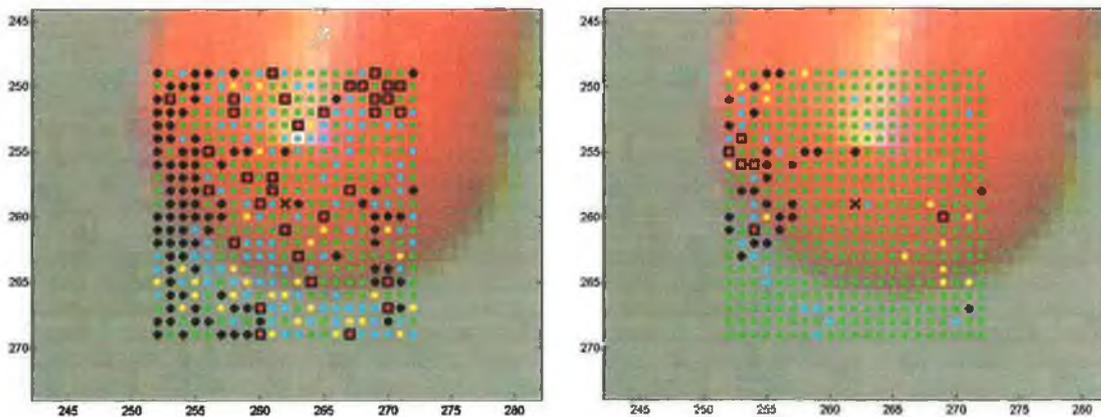


Fig 5.8 – Test#11 accuracies: Theta_YZ on left, Theta_XZ on right

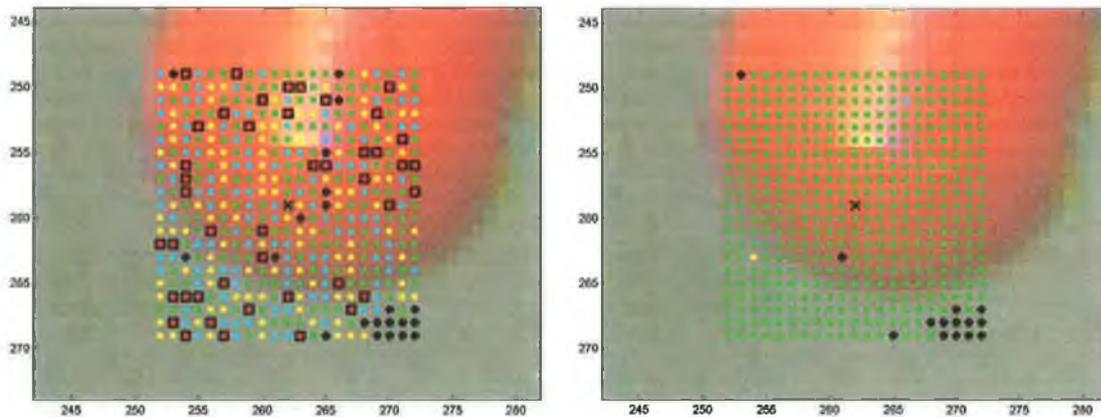


Fig 5.9 – Test#23 accuracies: Theta_YZ on left, Theta_XZ on right

In Fig 5.8 there are an increased number of yellow and red squares in the left image as well as a larger number of black asterisks. The majority of the Theta_XZ estimates (green squares) still have an error $\leq \pm 2^\circ$ but there is an increased number of black asterisks and blue and yellow squares. A systematic spread of the error range is not observed in the Theta_YZ estimates of Test#11 (left of Fig 5.8), nor in the Theta_YZ estimates of Test#23 (left of Fig 5.9) in which there is a significant population of each square colour and of asterisks. Out of the three groups of tests, Test#11 has the largest errors in the Theta_XZ estimates. It is seen in Fig 5.8 (right) that compared to Test#2 (right Fig 5.7) and Test#23 (right Fig 5.9) there is a large number of black asterisks corresponding to either an error greater than $\pm 10^\circ$ or to a non-result. Over the three tests there is a systematic spread observed in the error range in the Theta_XZ estimates (right side of Figs. 11 – 13). It is seen that as the passed input location moves away from the optimal location the performance degrades.

5.3.2 - Optimal Performance

The output of the algorithm for each of the twenty eight tests is shown in Table 5.1. These results are based on the optimal performance of the algorithm, i.e. the teat location in the image passed as an input to the angle extraction algorithm would result in a green square in the previous section. For this to occur the algorithm correctly chooses the lines corresponding to the sides of the teat from the selection created by the Hough line detection stage, and, the said selection includes lines that are an accurate approximation

for both sides of the test. The 'Time' column in Table 5.1 refers to the amount of time in seconds that it took for the algorithm to complete its processes once executed. For these tests all image outputs for user display were removed from the algorithm. The tests were run in the Matlab environment on a Windows XP PC with an Intel Pentium 4 2GHz processor and 256MB of RAM.

Test #	Croppoint				Microscribe		Vision System		Difference		Time (seconds)
	u left	v left	u right	v right	theta YZ	theta XZ	theta YZ	theta XZ	theta YZ	theta XZ	
1	292	262	322	260	1.41	-0.02	-3.46	-0.62	-4.87	-0.60	2.91
2	262	259	291	257	-0.18	-13.76	-0.38	-13.73	-0.20	0.03	2.39
3	246	253	274	250	-0.66	-23.68	-1.54	-23.84	-0.88	-0.16	2.11
4	208	230	241	225	0.31	-46.43	1.12	-46.95	0.81	-0.52	1.88
5	316	255	335	253	-18.39	0.69	-20.91	0.12	-2.52	-0.57	2.17
6	277	251	301	251	-14.80	-17.79	-14.86	-18.81	-0.06	-1.02	2.11
7	243	236	267	234	-14.84	-35.30	-17.38	-35.20	-2.54	0.10	2.05
6	226	223	250	219	-14.96	-47.08	-13.03	-47.57	1.93	-0.49	2.22
9	303	249	323	247	-25.23	-0.70	-28.13	-1.77	-2.90	-1.07	2.22
10	267	244	288	242	-25.40	-21.58	-26.88	-21.77	-1.48	-0.19	1.92
11	235	228	256	226	-27.88	-38.20	-28.45	-39.10	-0.57	-0.90	2.02
12	219	214	245	211	-29.86	-49.77	-30.17	-51.02	-0.31	-1.25	2.11
13	302	233	311	229	-43.39	-3.58	-44.12	-4.65	-0.73	-1.07	2.42
14	261	212	270	206	-52.44	-32.59	-53.38	-32.94	-0.94	-0.35	2.27
15	239	205	251	200	-51.66	-43.49	-53.73	-43.96	-2.07	-0.47	2.17
16	212	193	230	189	-52.55	-57.53	-52.89	-59.86	-0.34	-2.33	2.45
17	298	265	337	264	15.64	1.70	14.21	1.41	-1.43	-0.29	2.73
18	267	262	307	261	14.93	-14.29	13.89	-14.13	-1.04	0.16	2.53
19	243	248	281	247	15.67	-30.90	16.86	-31.01	1.19	-0.11	1.72
20	218	229	255	225	18.73	-48.80	20.87	-49.09	2.14	-0.29	1.81
21	292	254	337	255	31.10	-0.93	32.81	-0.87	1.71	0.06	2.14
22	263	245	309	241	32.25	-21.45	31.67	-21.08	-0.58	0.37	2.00
23	247	243	290	243	34.58	-32.91	38.71	-34.12	4.13	-1.21	1.88
24	229	220	274	216	40.93	-50.87	41.73	-52.24	0.80	-1.37	2.02
25	300	247	348	246	40.86	-1.45	-	-	-	-	2.34
26	258	243	311	243	39.74	-23.13	38.03	-22.82	-1.71	0.31	2.09
27	232	227	280	224	44.42	-45.38	46.91	-45.80	2.49	-0.42	1.98
28	225	223	261	222	43.93	-50.42	41.65	-50.48	-2.28	-0.06	1.97

Sum of Absolute Errors

42.64	15.75
-------	-------

Average Cycle Time

2.17

Average of Absolute Errors

1.58	0.58
------	------

Table 5.1 – Results of accuracy tests: optimal algorithm performance

The results seen in Table 5.1 show the angle extraction algorithm to be highly accurate. The maximum error of the angles in either plane is -4.87° (Test #1) while the larger average error is 1.58° . The best results were produced by Test #2, an error of -0.2° and

0.03° in the theta YZ plane and theta XZ plane respectively. The results of Test #25 were discounted because the algorithm failed to correctly identify the sides of the teat.

5.3.2.a - Test #2: Best Results

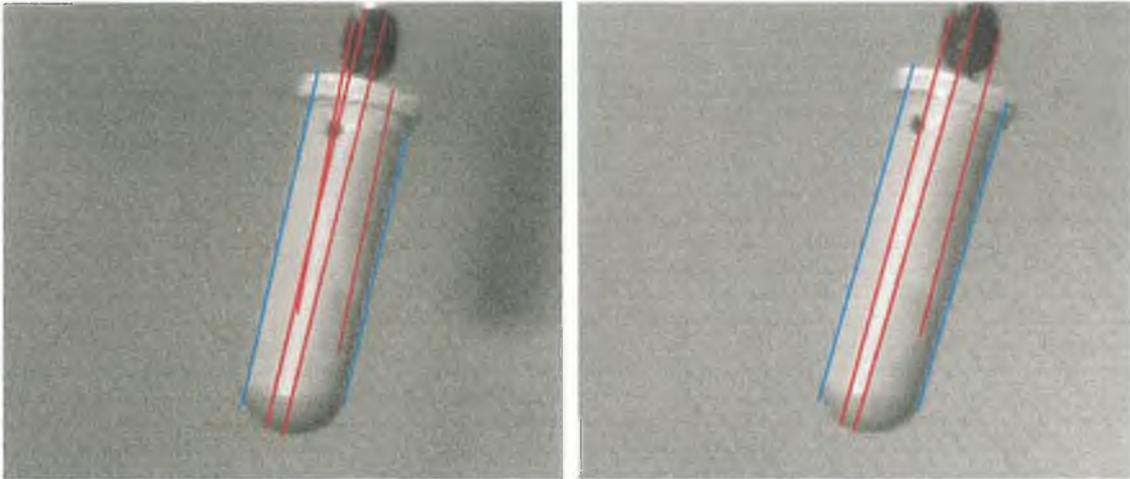


Fig 5.10 – Left and right images of lines found in Test #2

As can be seen in Fig 5.10 the chosen lines (shown in blue) are an excellent approximation to the sides of the teat

5.3.2.b - Test #1: Worst Results

Fig 5.11 shows the chosen lines for Test #1. As can be seen the lines (in blue) appear to be good approximations to the actual sides of the teat. However, both edges of the teat in the left image are not where they appear to be in the greyscale image. In the colour image of Fig 5.12 it is seen that the approximations are in fact not that good.

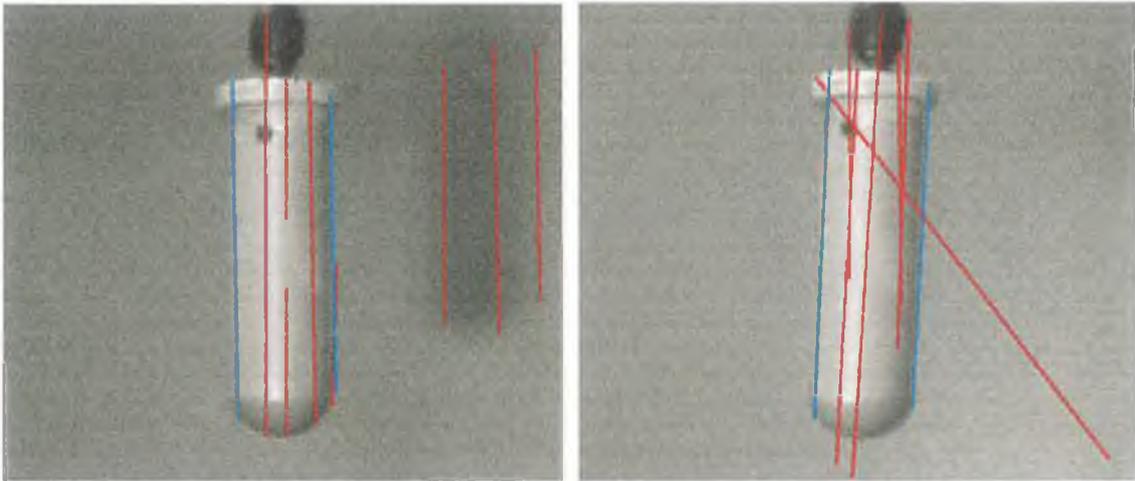


Fig 5.11 – Left and right images of lines found in Test #1

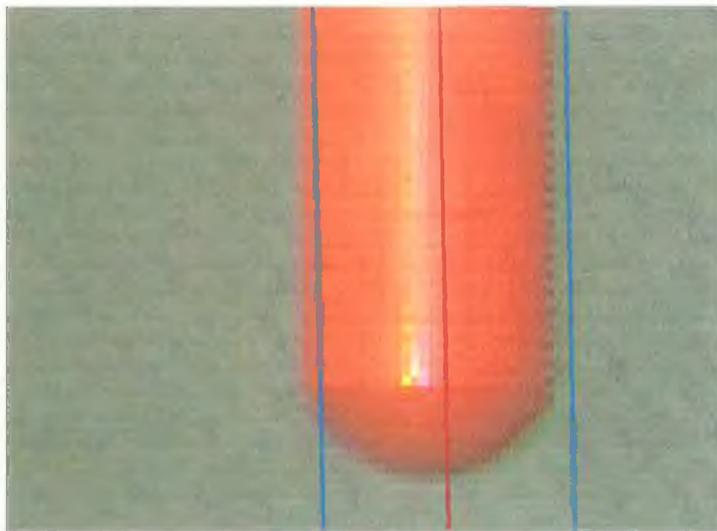


Fig 5.12 – Left RGB view of teat sides in Test #1

This is the source of the error in the first test sample. Repeating the test and manually selecting points on the sides of the teat in the RGB image produced the teats side lines found in Fig 5.13. This manually assisted test yielded angle measurements of 0.85° in the Theta YZ plane and -0.83° in the Theta XZ plane, an absolute error of 0.53° and 0.81° respectively. This is a reduction in the absolute error of 4.34° in the Theta YZ plane (and a slight increase (0.23°) in the Theta XZ plane).

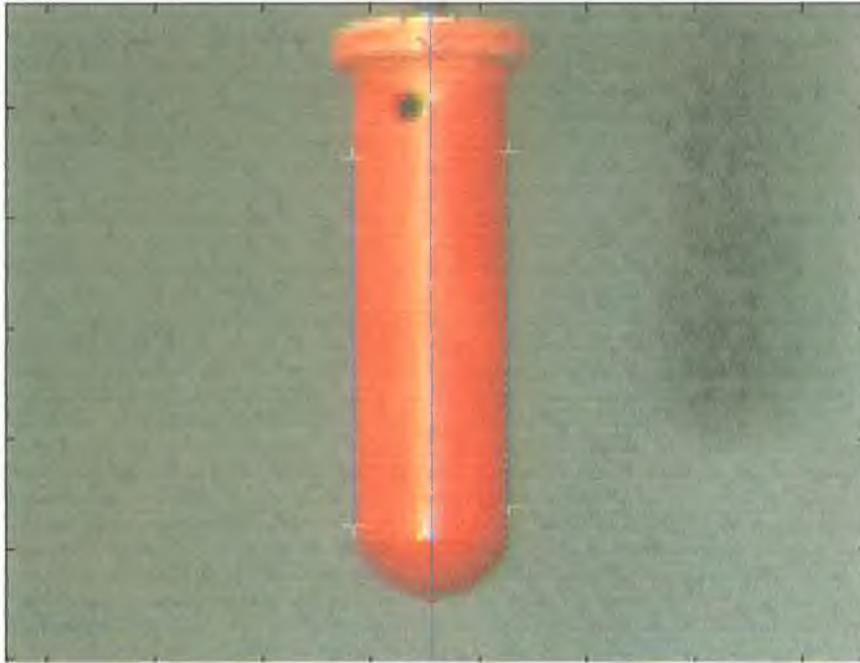


Fig 5.13 – Side lines manually selected in left view of Test #1

5.3.2.c - Test #23: Teat Side Approximation Error

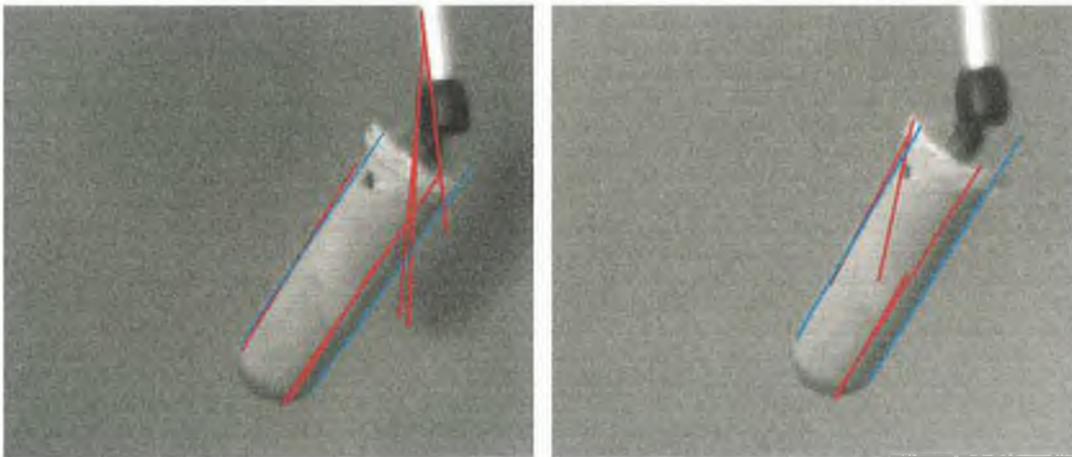


Fig 5.14 – Left and right images of lines found in Test #23

In Fig 5.14 the leftmost chosen lines in each image do not coincide exactly with the side of the teat. The lines chosen are the most suitable available in the selection provided by the Hough Line detection stage of the angle algorithm, but, it can be seen that a line

bisecting the chosen line and the next most suitable line in each image would be a better approximation to the sides of the teat.

Increasing the resolution of the Hough Line Detector enables the algorithm to better approximate the sides of the teats. The test results of Table 5.1 were found using a 'dtheta' value of 0.5° as a parameter for the Hough transformation.

Test #23 was repeated with an angle resolution of 0.05° , the images from the test are seen in Fig 5.15. This repeat of the test yielded angle measurements of 31.64° in the Theta YZ plane and -33.14° in the Theta XZ plane, an absolute error of 2.94° and 0.23° respectively. This is a reduction in the absolute error of 1.19° in the Theta YZ plane and 0.98° in the Theta XZ plane. The cycle time increased to 97.297 seconds.

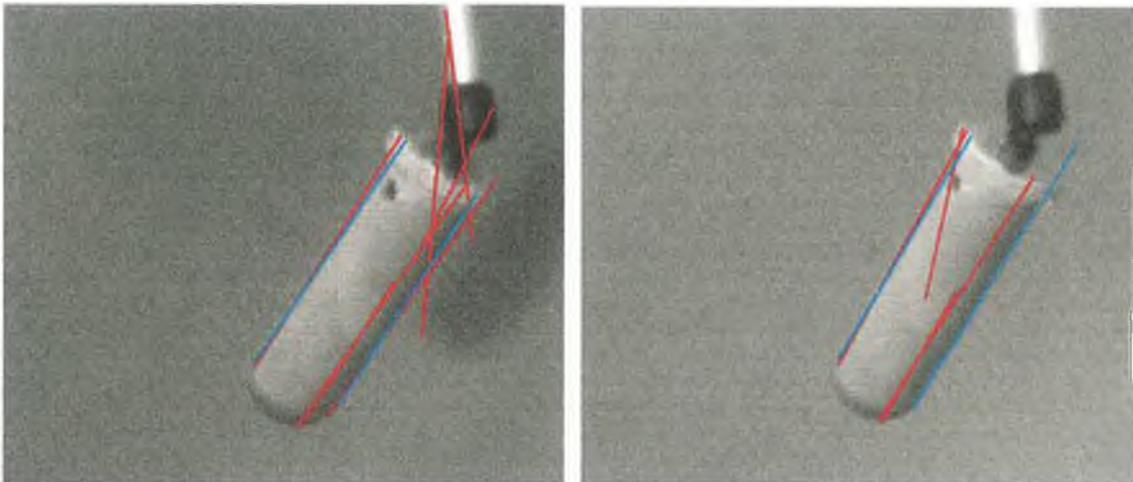


Fig 5.15– Repeat of Test #23 with increased resolution in Hough line detector

5.3.2.d - Test 25: Failed Detection

Fig 5.16 shows the selection of lines and the chosen pair for both the left and the right images. It can be seen that, as in Test #23, the line selection is not good. The chosen lines (blue) do not correspond to the side of the teat but there are choices available that do.

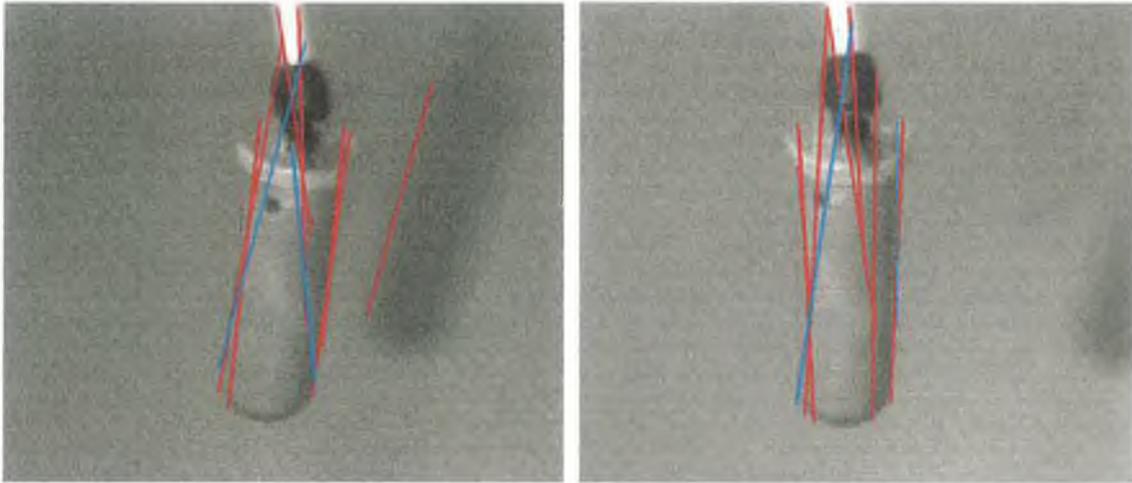


Fig 5.16 – Left and right images of lines chosen in Test #25

5.3.3 - Non-Optimal Performance

In Section 5.3.1 it is seen that the algorithm does not always perform optimally and it depends on the location of the starting coordinates passed to the algorithm relative to where the teat end is located in the images. This section presents examples of the different causes of the non-optimal performance is the test sample studies of Section 5.3.1.

5.3.3.a - Non-Optimal Examples: Best line approximations not chosen

Fig 5.17 contains the lines chosen in the left image of Test #2 when the location input is shifted two pixels to the right and one pixel upwards from the location that produced the optimal results seen in Table 5.1. The line chosen to represent the left side of the teat is not a good approximation. There is another line available in the selection that is clearly seen to be a better approximation to the teat side.

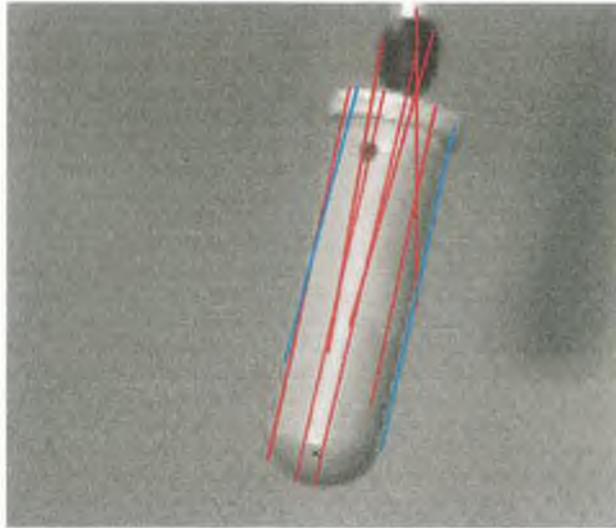


Fig 5.17 – Lines chosen in left image of Test #2 during non-optimal performance

This iteration of the algorithm produced a Theta_YZ estimate of 4.42° and a Theta_XZ estimate of -14.05° . The absolute errors are 4.24° and 0.29° respectively. The error associated with the Theta_YZ value is 21.2 times greater than that of the optimal performance result in Table 5.1. This error produces the corresponding yellow box in Fig 5.7.

5.3.3.b - Non-Optimal Examples: Best line approximations not available

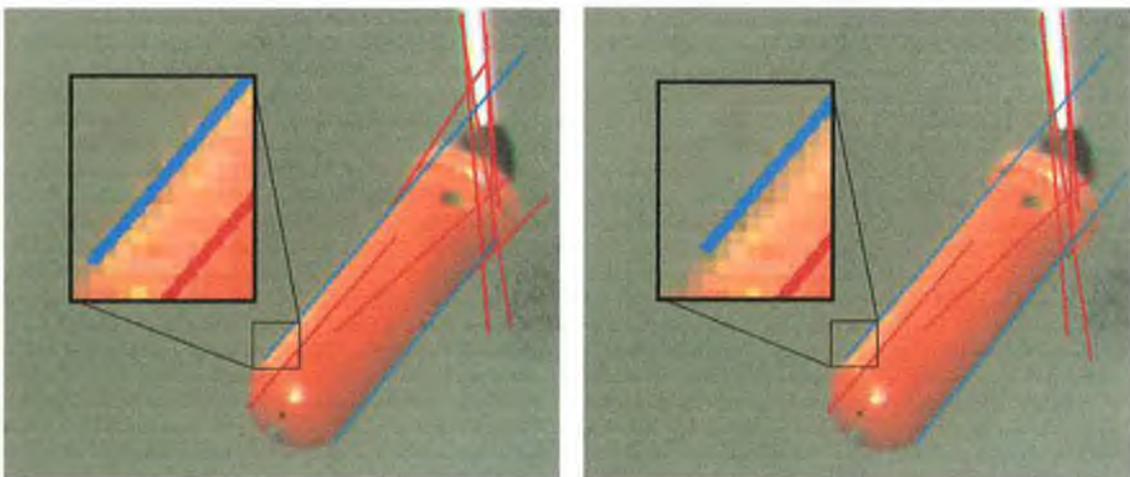


Fig 5.18 – Lines Chosen in left images of Test #11, optimal on left, 1 pixel up & 1 pixel left on right

The left image in Fig 5.18 shows the lines chosen by the algorithm in the left image during the optimal performance that produced the result in Table 5.1. The image on the right corresponds to the algorithm being re-iterated with the passed input being one pixel upwards and one pixel to the left. From the zoomed windows in Fig 5.18 it is seen that the line representing the left side of the teat in the non-optimal test is not as good as that of the optimal test, but, the chosen line is the best approximation available for the algorithms selection. The non-optimal estimation is -21.46° for Theta_YZ and -39.62 for Theta_XZ. The absolute errors are 6.42° and 1.41° respectively. The error associated with the Theta_YZ value is 11.3 times greater than that of the optimal performance result in Table 5.1; the Theta_YZ error is 1.6 times greater. It is the Theta_YZ error that produces the corresponding red box in Fig 5.8.

5.3.3.c - Non-Optimal Examples: Failed Detection

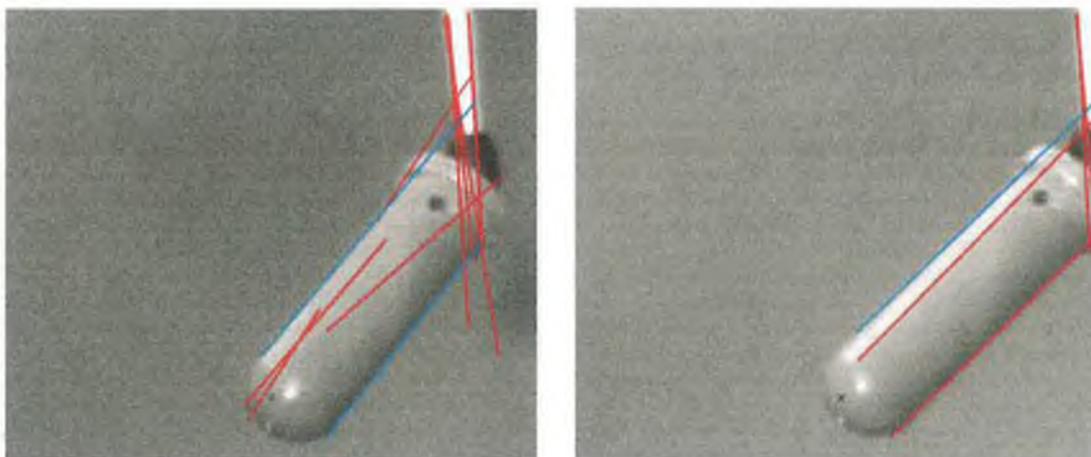


Fig 5.19 – Lines chosen in left(left) and right(right) images of Test#11 during non-optimal performance

Fig 5.19 shows the lines chosen by the algorithm in the left and right images of Test#11 during non optimal performance. The teat location input has been shifted up by four pixels, resulting in a failed detection and the corresponding black asterisk in Fig 5.8. It is seen in Fig 5.19 that the algorithm has failed to identify a line to approximate the right side of the teat in the right image.

Chapter 6 – Discussion

6.1 - IceRobotics Teat Tracker Review

The IceRobotics TeatTracker vision system could be incorporated into a rotary based AMS only if significant advances are made in its development. The system is capable of simultaneously identifying four teats in a scene and calculating their 3D locations, in approximate real-time, but, there are issues regarding the robustness of the teat identification process and regarding both the precision and the accuracy of the location measurements. There is a working volume within which the highest levels of accuracy are attained. In Chapter 3, Section 3.2.3.d, Fig 3.49 and Fig 3.52 illustrate the region, to which accurate results are confined. The region is 80mm deep and 150mm wide

6.1.1 – Robustness of Teat Identification Process

The ability of the IceRobotics TeatTracker to identify teats is explored in Chapter 3, in Section 3.2 for nine particular circumstances. The performance is judged on the ability to successfully identify a target teat. A successful identification occurs when the system correctly identifies corresponding image points in the left and right images of the stereo pair belonging to the same teat target. Section 3.2.2 contains the results of the performance in testing with the different criteria for each of the nine circumstances. It was found that in general the system is capable of producing desirable results in all of the defined setup conditions, but it cannot reproduce them reliably. The ideal scenario in Fig 3.12 is the benchmark on which all other scenarios are based. The scene contains four well-illuminated teats in a standard udder formation, located within the target region of the vision system. All teats are clearly defined in the stereo images and none of the teats are obscured in either view. The background scene is neutral; there are no teat-like objects or artefacts in the images due to the background. In this first scenario the system successfully identifies all of the target teats. The second scenario involves all the teats being partially obscured, in both camera views, either allowing the tops or the bottoms of the teats to be seen by the cameras. It is seen that the vision system must have a view of

the teat end (the bottom) in order for it to identify a teat. The third scenario investigates the systems ability to identify a teat that is partially occluded in either one of the camera views or in both camera views. It is seen that the system is still capable of identifying partially occluded teats when the bottom of the teat can be seen in both the left and right camera views. In the fourth scenario an object which approximates the shape of a teat is introduced to the background of the scene, outside of the target region. It is seen that the system falsely identifies such an object as a teat. Corresponding points of the non-teat object in the stereo images will result in a disparity that is too small for a teat within the target region (the object is outside of the target region, its depth in the image is too great). Thus, in falsely identifying the object the system makes false correspondences between the non-teat object in one image, and another, different, teat shaped object in the other image of the stereo pair. The resulting disparity of this false correspondence results in a triangulated teat that resides inside the target region. In the example given in Fig 3.20 the false correspondence is made between two different non-teat objects outside of the target region, the resulting triangulated point is within the target region. The fifth scenario has a non teat shaped object within the target region. The results of this test demonstrate the shortcoming of the IceRobotics system. The object is falsely identified as a teat. There is no clear reason as to why the system chose the object that doesn't resemble a teat in favour of an unidentified target teat in the centre of the target region being well illuminated and clearly defined in the camera images. The system performs well in identifying the closely bunched teats in the sixth scenario. The occlusions are not an issue provided the bottom of the teats can be seen by both cameras. The outcome of the seventh scenario has bearing on the teat angle extraction algorithm presented in later chapters. The scenario investigates the ability of the system to identify a teat that is not pointed straight down. The system is seen to perform equally well in identifying angled teats as it does with non angled teats. Teats angled both towards and away from the cameras, and angled to the left and the right in the images are successfully identified. This result is crucial as the basis of extending the systems functionality to provide teat angle measurements in that the system is able to first locate the teat in the images. In scenario eight the teats are moving within the target region. The system can track targets in

approximate real time, providing an updated location for an identified teat every $\frac{1}{10}$ of a second. The stereo pairs of images are captured instantaneously, and for each instance the system identifies the teats from the static images captured. The final scenario involves objects moving in the background scene. The system is prone to misidentifications in this case due to the fact that if there are regular movements of multiple objects in the background scene, it is likely the objects will interact in such a way as to appear to be a teat in the camera images. The output of the system is seen to be unstable and unpredictable. Even in ideal conditions (scenario one) a minor change in the position or in the orientation of the teats can lead to the system failing to identify a teat that was being successfully tracked prior to the changes. Changes in the position or in the intensity of the light source are seen to have the same effect on the output stability. There is indecision in the coordinate outputs of the system even when all targets have been correctly identified in a static scene with constant illumination. All three component values of the coordinate triples are seen to fluctuate, usually within a range of $\pm 1\text{mm}$, with the biggest fluctuations being seen for the Z-component. The overall assessment of the systems ability to successfully identify target teats is that it is capable of identifying teats in various and demanding scenarios, but, the system is very prone to making false identifications.

6.1.2 – Accuracy of Teat Location Estimates

The accuracy of the system was tested by taking measurements of a single teat moved to different locations within a 300mm by 275mm horizontal plane. These measurements were compared to known accurate measurements for the teat locations. The 3D surface plots clearly illustrate the error of the systems measurements throughout the test plane. The region of best accuracy is achieved in the central area, as the teat location moves away from this area the error increases. The average error for the entire test region is 11.79mm. The largest error is 33.32mm and the smallest error is 0.69mm. At the outset of this project it was proposed that a teat location measurement with a certainty of $\pm 5\text{mm}$ or less would be sufficient in autonomously attaching a milking cup to the teat. To attain an average error within this requirement the test region must be confined to an area of

150mm wide (X plane of camera coordinate frame, as in Fig 3.52) by 80mm deep (Z plane). The average error for the ten test samples within this region is 4.14mm. The region is indicated by the blue hatched area overlaid on the standard teat formation in Fig 6.1. It is seen in Fig 6.1 that only the minimum expected teat formation dimensions fit inside the 150mm by 80mm area of highest accuracy. It is therefore not possible for the system to simultaneously provide four accurate location measurements for teats in the average standard udder formation. Either the front teats or the rear teats will be outside of the region of an accuracy of ± 5 mm.

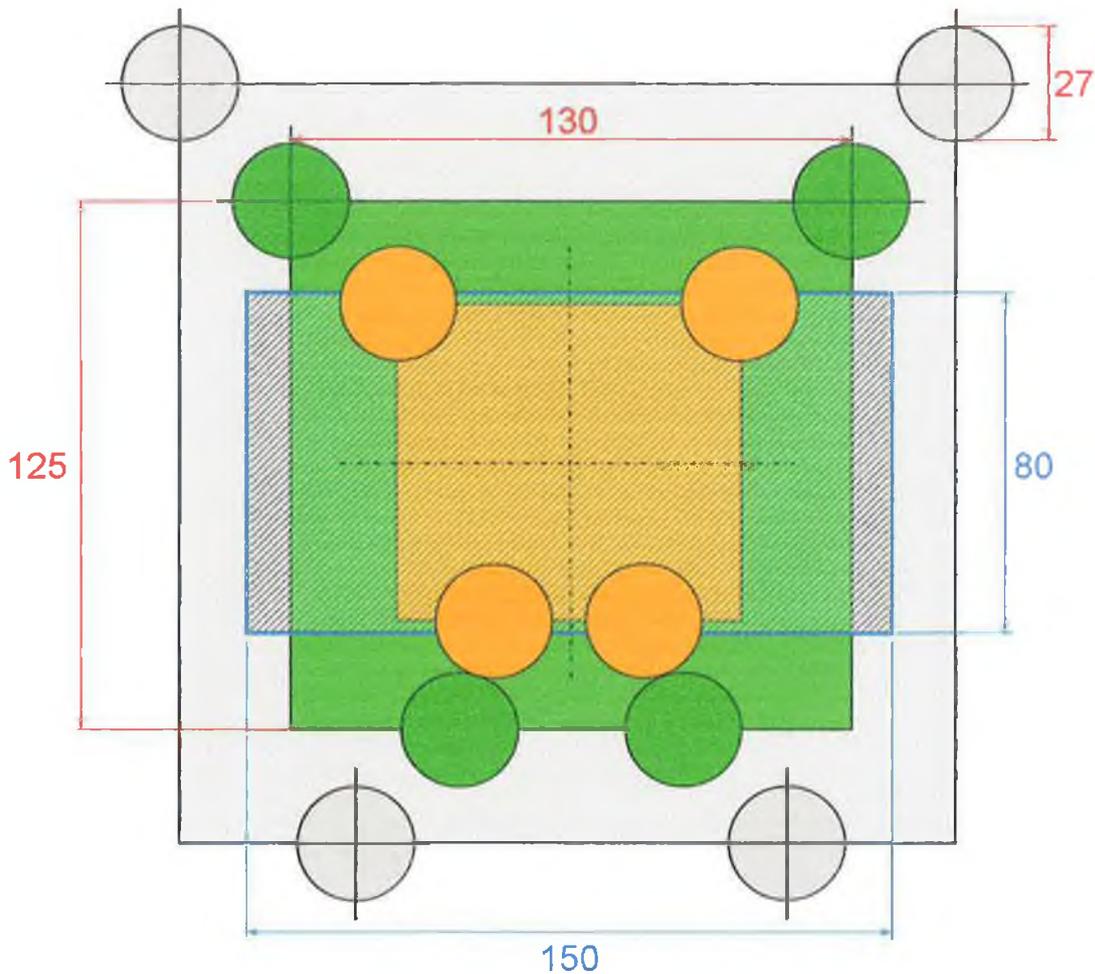


Fig 6.1 – Standard teat locations, region of best accuracy of vision system overlaid in blue hatch

If the system was modified so that the ± 5 mm accuracy area was increased to encompass the maximum expected teat formation dimensions, it would still not be sufficiently large.

The cows positioning in the stall is constrained by the dimensions of the stall and as such there is a region in which the udder of the cow is expected to be. Slight deviations from one cow to the next mean that the centre of the udder will not always be in the same location. A certain amount of leeway is required so that the cameras do not need to be repositioned relative to the udder in order for all four teats to be within the region of accuracy.

6.2 - Angle Extraction Algorithm Review

The angle extraction algorithm is still in the early stages of development. The basis of the algorithm is approximating the location the line of symmetry of the teat in each image of a stereo image pair and triangulating 3D points from these lines. The resulting 3D vector describes the teat angulations. Testing has produced promising results, showing the basis of the algorithm to be sound and capable of producing highly accurate results. However further development of the algorithm is required to improve the reliability in finding the lines that approximate the sides of the teat in the images. The algorithm must also be modified to accommodate non-cylindrical teats and these modifications must be validated by further testing. At that point testing on animals should commence as well as the integration of the algorithm into a real time multiple teat tracking system.

6.2.1 - Accuracy of Angle Measurements

In Chapter 5 the results of the optimal performance of the angle extraction algorithm in 28 tests are given in Table 5.1. These results show that when the algorithm identifies the teat sides correctly it is capable of producing highly accurate results with an error well inside the $\pm 5^\circ$. Of the 27 out of 28 successful test samples the maximum error encountered was 4.87° in the Theta YZ plane (teat angled towards (negative angle) and away (positive angle) from the cameras). The minimum error was in the Theta XY plane (teat angled to left (positive angle) and to right (negative angle) in camera images). It is expected that angle measurements in the Theta XY plane will be more accurate than those in the Theta YZ plane because changes in the angle in the Theta XZ plane do not result in a change of depth. Error is introduced by the stereo triangulation algorithm when

estimating depth. The position of the teat in the image plane is a direct measurement of the angle in the Theta XY plane. This measurement is highly accurate (down to pixel level) but not without error (blurring in images, edges exist at a sub pixel level). A slight error encountered in Theta XY plane is compounded in the Theta YZ plane because the depth calculation is based on the disparity in the Theta XY plane. The test results verify this, the average absolute error in the Theta YZ plane is 2.72 times greater than that of the Theta XZ plane (1.58° versus 0.58°). The two samples that have the largest associated error are Test #1 and Test #23. From Fig 5.12 it is seen that the lines representing the sides of the teats in the left image of Test #1 are not a good approximation of the sides of the teat. Manually adjusting the positioning of these lines, so that they coincide with the teat side extremities (as seen in Fig 5.13), yielded an overall reduction of 4.11° in the total error of the two reference planes (Theta XZ and Theta YZ). The error encountered in Test #23 is due to the algorithm being unable to provide accurate approximations to the teat sides. It is seen in Fig 5.14 that the approximations in the left image are close to the sides but they do not coincide exactly with the edges. This leads to the relatively poor results, having an overall absolute error total of 5.34° . This test was repeated with the 'dtheta' parameter of the Hough transform function being changed from 0.5° to 0.05° . The 'theta' parameter specifies the resolution of the Hough transform as discussed in Chapter 2.2.2. The line detector searches for straight lines rotated through 180° at intervals of 'dtheta' degrees. The higher resolution line search improved the approximations of the teat sides (as seen in Fig 5.15) and reduced the error of the result by a total of 2.17° in absolute terms. Increasing the angular resolution by a factor of 10 increased the overall cycle time of the algorithm by a factor of 51.75 (time at dtheta = 0.5° : 1.88s; time at dtheta = 0.05° : 97.297s). The cycle time for the increased resolution is 44.8 times the average cycle time for the 28 tests.

6.2.2 - Performance in Detection of the Teat Sides

6.2.2.a – Non Robust Detection

In Chapter 5, Section 5.3.1, the fragile nature of the algorithm is exposed. Fig 5.7, Fig 5.8 and Fig 5.9 illustrate that optimal performance of the algorithm relies on the position

coordinate that is passed as an input. Tests #2, #11 and #23 have been selected for this particular study because Test #2 and Test #23 produced the best and worst results in testing respectively, Test #11 was chosen as it is used as a worked example in Chapter 4. Paying particular attention to the Theta_YZ estimates of Test #11 (left of Fig 5.8) it is seen that the angle extraction algorithm is not robust in its current form. There is a non-systematic spread in the errors produced in the Theta_YZ estimates iterating the algorithm with the different input locations. This can be said for all three of the studies, Test#2, Test #11 and Test #23. Changes in the input as small as one pixel can cause the algorithm performance to change from optimal to non optimal. This is demonstrated in Section 5.3.3 of Chapter 5. Due to changes in the input, the algorithm is seen to a) choose a poor line to represent the side of a teat when there is a better line approximation available for selection (Fig 5.17), b) fail to generate a selection of lines containing accurate approximations of all teat sides (Fig 5.18) and c) fail to detect both sides of the teat in each image (Fig 5.19). The small input changes refer to changes in the pixel coordinates representing the teat end in the left and right images of the stereo pair. It is the IceRobotics system that is intended to pass the teat end location to the angle extraction algorithm, however, in testing the positions of the teat end in the images was manually passed to the algorithm. This eliminates the issue of miss-identifications made by the IceRobotics system. Changes in the location of the point given to represent the end of the teat can directly affect the performance of the algorithm in two ways: 1) Altering the contents of the scene and 2) Offsetting the reference point used by the line filters,

6.2.2.b – Altering the Scene

One of the first steps in the algorithm is to crop the images to a region of interest around the teat based on the given teat end coordinate. The Hough transformation is then applied to the cropped image in the process of detecting straight lines. The Hough line detector outputs possible straight lines in the order of likelihood that they are in fact a straight line. The transform is applied to the entire scene (in this case all pixels in the cropped image) and thus the contents of the cropped image can affect the priority of any given line. As the cropped region moves in the image its pixel contents will change and so may the

priority of the detected lines. Fig 6.2 and Fig 6.3 contains the lines detected in the left image of Test #2 for two different input coordinates: (262,259) and (264,257).

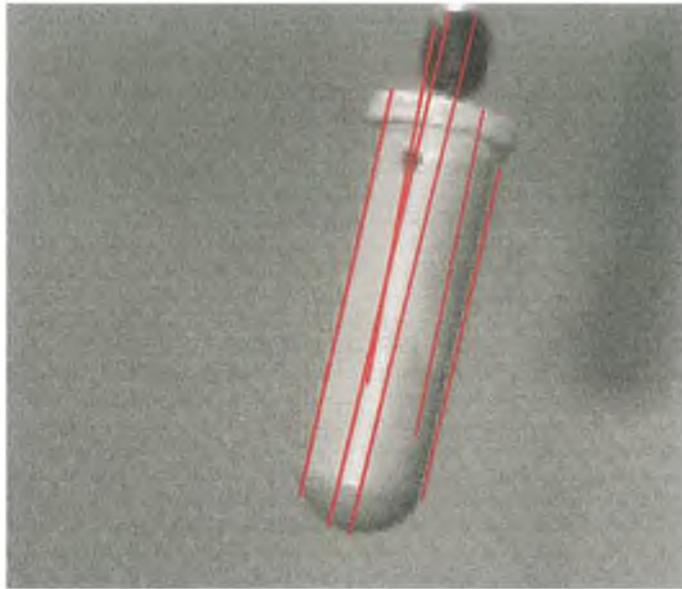


Fig 6.2 – Lines in left view, Test #2, input = (262,259)

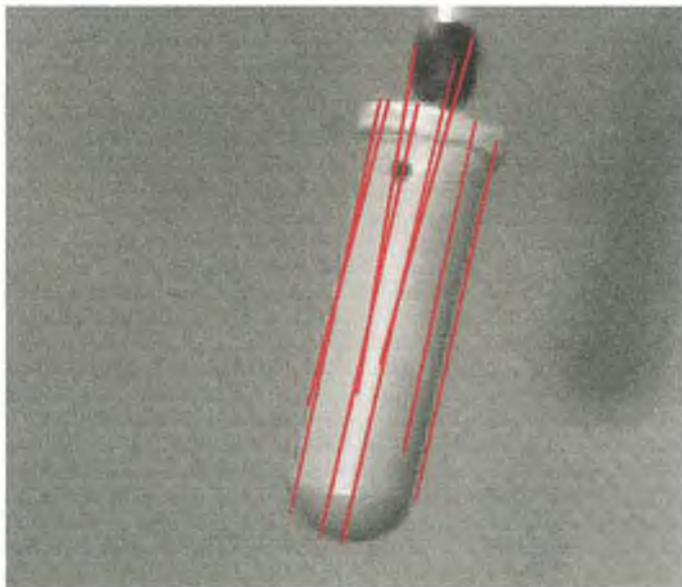


Fig 6.3 – Lines in left view, Test #2, input = (264,257)

On close inspection of the previous two figures the subtle differences in the output of the Hough line detector can be observed. The line selection has changed with the addition of

two extra lines in the image. The left hand edge approximation has degraded with the selection of a suboptimal line. This example illustrates how a small change in the input to the algorithm can dramatically affect the performance in terms of locating the sides of the teats: the choice of approximating lines may become more complex (as on left of teat) or the approximation value may change (as on right side of teat).

6.2.2.c – Offsetting Linefilter Reference Point

Small input changes affect the choice of the lines from the Hough line detector selection. As described in Chapter 4.2.6.a, some of the functions employed to filter out the lines not of interest rely on hard-coded pixel distances from the point in the image where the algorithm is told the teat end is located. In section 4.2.6.b the sequence of line filters used in testing was:

1. `filter_farfromcrop`
2. `filter_leastpointdistance`
3. `filter_maxnormpairleftright`

The first two of the above filters both rely on the location of the crop point (the inputted teat end coordinate in the image). The first filter removes lines whose perpendicular distance from the crop point is too great (in testing, for the results produced in Chapter 5 the maximum distance was set to 30 pixels). The second filter eliminates lines not having at least one end point within a certain absolute distance hard-coded in the program. In producing all the documented results in Chapter 5 this hard-coded value was set to 35 pixels. Fig 6.4 demonstrates the effect of this value on the process of choosing the teat side lines. The image shown corresponds to the left camera image of the teat in Test#2 of a non optimal test result. The green circle has a radius of 35 pixels; its centre is located at the crop point in the image, denoted by a black 'x'. The black circle has its centre located at the same point but has a radius of 25 pixels. If the hard-coded value of the least point distance filter (2. `filter_leastpointdistance`) was set to 25 pixels then only lines having an endpoint within the area of the smaller black circle would qualify as possible teat side candidates. As can be seen the line incorrectly chosen as the best approximation available

to the left side of the teat does not have an end point within the black circle, but does within the larger, 35 pixel radius, green circle.

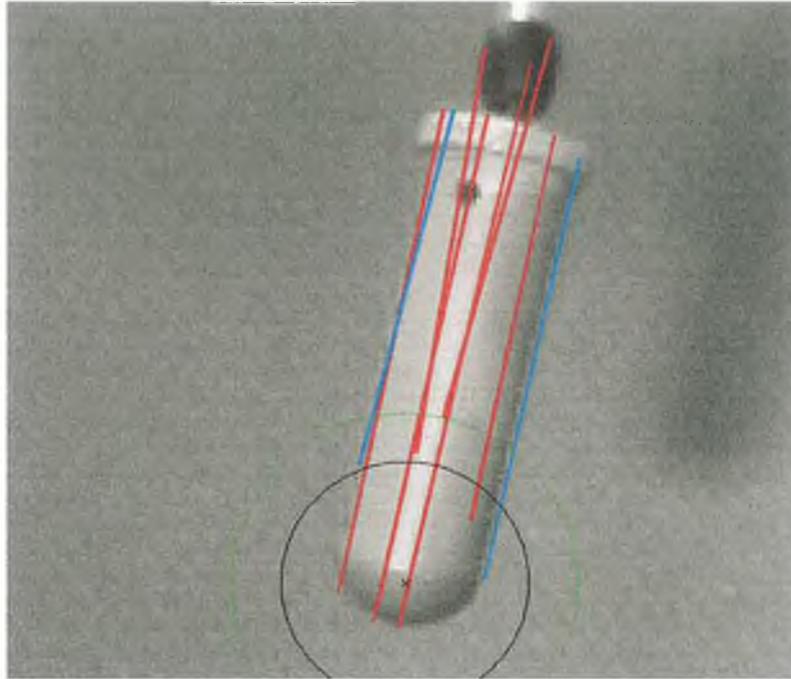


Fig 6.4 – Circles of radius 25 & 35 pixels, centre at croppoint, left image of teat in non optimal iteration

In this case reducing the least point distance parameter would improve the performance of the algorithm and the accuracy of the angle approximation (as seen in Test #2 of the optimal results). However, this may not be the case for all orientations of the teat. The value is set to the seemingly large (relative to the location of teat ends seen in Fig 6.4) value of 35 pixels to ensure the reliable completion of the algorithm throughout the range of tests. Non-completion occurs when the appropriate teat side lines are incorrectly eliminated; in this case the algorithm might only choose one or zero lines to represent the sides of the teat. This means that the central axis cannot be calculated and in turn the teat angles cannot be estimated. An emphasis was placed on avoiding this occurrence in choosing the sequence of filters to eliminate line choices. The sequence of the three filters used in testing is very simple. Using this sequence the algorithm reliably finds two lines (though not necessarily the best lines) in each image from which a central axis can be calculated. The full collection of filters developed was not exhaustively tested. Further

development of the filter sequence may lead to improved decisions in choosing the side lines in future testing.

6.2.2.d – Systematic spread of Error

It has already been stated that there is no systematic spread in the errors encountered in the Theta_YZ estimates for different location inputs. This however is not the case for the Theta_XZ estimates. Looking at the right hand sides of Figs 5.7 - 5.9 definite groupings are observed. In Test #2 (right of Fig 5.7) there is a single column of blue squares situated seven pixels to the left of the centre point. The upper left corner of the region contains a large population of asterisks, as does the lower right corner in the Test #23 Theta_XZ results (right of Fig 5.9). The left side of the region for Test #11 (right of Fig 5.8) has a large population of non optimal results. These observations confirm that there is a systematic spread in the associated error, and it is a function of the distance and location relative to the optimal centre pixel. It is seen that this function varies from one test to the next.

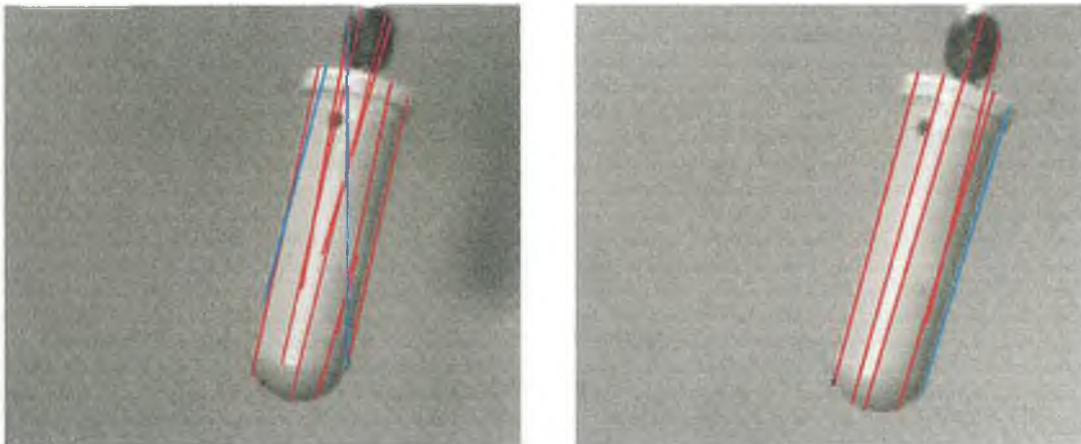


Fig 6.5 - Test #2 ,Left: Lines for column of blue squares, Right: Lines for Asterisks

Fig 6.5 illustrates the effects of the input changes that lead to the column of blue squares (left) and the black asterisks (right) in the Theta_XZ estimates for Test #2 (right of Fig 5.7). In the left image of Fig 6.6 the input location (black 'x') is located on the lower left edge of the teat. Iterating the algorithm with this input has caused it to produce a vertical

line in the image. This line is not produced in the other pixel positions of the column that do not correspond to blue squares (Fig 5.7). In the right image of Fig 6.5 the algorithm has not selected a line to represent the left side of the teat. The reason for this is that the appropriate line is eliminated. Because the perpendicular distance between it and the crop point is too small.

6.2.2.e – Minimal Error in Theta XZ estimates

It is clear from the iteration tests in Section 5.3.1, Chapter 5, that the algorithm is significantly more robust in its estimates of the Theta_XZ angles than the Theta_YZ angles. This is because they don't rely on depth being recovered from the scene- the Theta_XZ plane is approximately parallel to the image planes of the cameras. A side effect of the Theta_XZ plane being approximately parallel to the image planes is that the algorithm does not need to correctly choose the lines to represent the sides of the teat in order to produce accurate estimates for Theta_XZ, provided the chosen line has the same slope as the correct line.

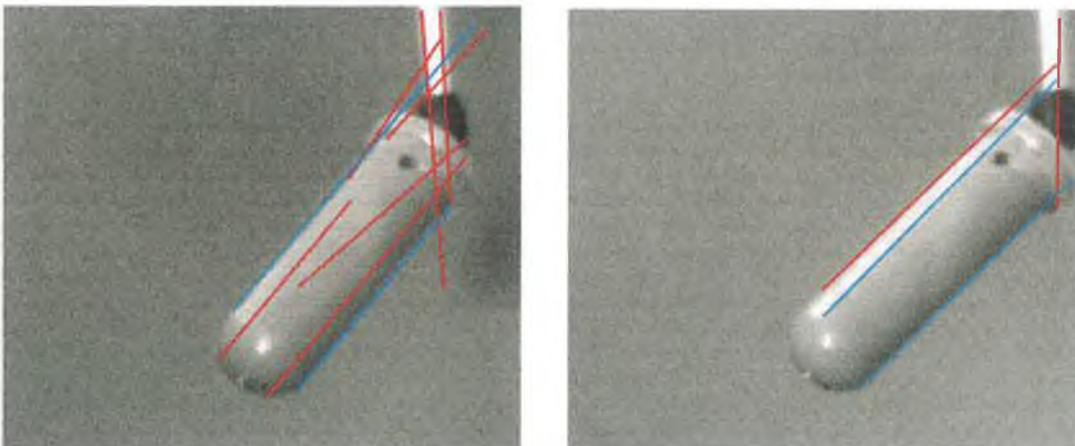


Fig 6.6 – Lines chosen in left and right images of Test #11, input = 4px right, 7px down



Fig 6.7 – Centerlines in images of Test #11, input = 4px right, 7px down

The images in Fig 6.6 contain the line choices made by the algorithm when the input location is shifted 4 pixels right and 7 pixels down. It is seen that the line chosen to represent the left side of the teat in the right image is not actually on the edge but it is parallel to it. Thus the centerline produced in the right of Fig 6.7 is not collinear with the central axis of the teat but it is parallel to it. The error of this estimation is 6.23° in the Theta_YZ estimate but only 1.43° in the Theta_XZ estimate.

6.2.3 - Further Development of Algorithm

It has been seen that the level of accuracy attained in the angle estimations depends greatly on the ability of the algorithm to approximate the sides of the teat. It is this facet of the algorithm that requires refinement. Without straying from the structure of the algorithm presented in Chapter 4 the next logical step would be to develop sequences and combinations of filters used in choosing the lines to represent the sides of the teat. The decision process could be iterated, allowing for a coarse detection using one set of filter sequences and then refining the search region and applying another set of filter sequences. The first set of filters would produce robust general estimates of the sides of the teats (similar to the current incarnation); the second set of filters could apply a stricter rule base once the general location of the teat sides is known. The effect of increasing the angular resolution of the Hough line detector is discussed in Section 6.2.1. An iterative process could similarly be used to allow a more refined search while keeping processing time to a minimum. For a coarse search a larger 'dtheta' value could be used, reducing the number of possible discreet angulations for a detected line. Once the regions of

interest known to contain the teats sides have been identified the Hough line detector could be re-applied to these regions using a smaller value of 'dtheta'. The effect of increasing the angle resolution on the processing time will be reduced as there will be less possible orientations to search through in the regions of interest. Pre-processing of the images prior to the algorithm would improve the detection of the teat sides and hence improve the accuracy of the angle estimations. As seen in Chapter 5 many of the errors encountered were due to the poor definition of the teat edge in the image. An example of the possibilities of image pre-processing is seen in Section 4.2.2, Fig 4.6. In this example the red channel of the RGB image was extracted. It is seen in Fig 4.6 that there is a significant improvement in the definition of the teat edge due to it being red in colour. This is just an example of the possibilities, further investigation is required.

Chapter 7 – Conclusions

7.1 - Teat Identification and Location

A stereo vision system, purpose built for the task of tracking cow teats, was acquired from IceRobotics. The system was assessed in terms of its ability to successfully identify target teats in the working region, and, in the accuracy of the teat position estimates it provides. Nine specific scenarios demonstrate that in all of them the system is capable of identifying teats but that the system is prone to making false identifications. False identifications are made of non teat shaped objects within the target region and of objects resembling teat proportions in the background of the scene but outside of the target region. Both of these scenarios are severely problematic as they are likely to occur in practise because of the presence of the robotic manipulator moving in and out of the working region. When moving within the target region it is likely that the system will falsely identify as a teat some part of the manipulator or the cups that it is applying. When moving within the background (behind the cow) a section of the manipulator that loosely resembles the teat (in the image of the cameras) may be matched to a different target, creating a disparity that produces a position estimate within the target region. The system is not robust in identifying teats. This needs to be resolved should the system be considered for continues incorporation into the AMS. The accuracy of the system is within $\pm 5\text{mm}$ within a region 150mm wide by 80mm deep. This error is acceptable for automated attachment but the region of accuracy is not large enough that all four teats can be reliably identified simultaneously. If the cameras are to remain stationary then it is necessary that the region of accuracy be enlarged. The reproducibility of the position estimates made by the system is insufficient. Testing has shown the output of the system to vary depending on lighting conditions. The stability of the system output is also shown to vary when conditions are stable. These problems must be overcome if the system is to be incorporated into the final AMS.

7.2 – Angulations of Teats

The IceRobotics system provides no information regarding the teat orientation. This information is required in simultaneously attaching four milking cups. An algorithm has been developed to utilise the hardware of the IceRobotics system and its teat position outputs to provide estimates of teat angulations. The performance of the algorithm has been assessed in terms of its capability to estimate the angles when the central axes in the stereo images have been correctly identified, and in terms of its ability to correctly identify the central axes. When the centre lines are correctly identified the algorithm produces excellent results, all tests in this optimal scenario resulted in angle estimate with an associated absolute error of less than $\pm 5^\circ$. The average absolute error of the angle in the plane parallel to the image plane is $\pm 0.58^\circ$. In the plane perpendicular to the image plane it is $\pm 1.58^\circ$. The angle estimate in the plane parallel to the image plane is expected to be more accurate as it does not rely on depth recovery from the stereo images. In its current incarnation the algorithm is not robust in its identification of the central axes. The algorithm relies on determining lines in the images that approximate the sides of the teat. It is seen that the algorithm is capable of making accurate approximations but this depends on the teat location input that is passed to the algorithm. A change of one pixel in the input location can cause the algorithm to make less accurate approximations to the teat sides. This leads to a less accurate approximation to the central axes which in turn leads to increased error in the outputs. The source of this non-robust performance is primarily the stage in the algorithm where possible lines to represent the teat sides, generated by the Hough line detector, are eliminated until there are two lines remaining. The line filters that have been employed rely heavily on the position input passed to the algorithm and are thus prone to errors when the input changes.

7.3 – Future Work and Recommendations

The IceRobotics system requires development if it is to be suitable for incorporation into the AMS. A hardware upgrade, including cameras of greater resolution and improved optics, is suggested to improve the accuracy of the position estimates and to increase the region of accuracy. Further development of the systems internal algorithms is required to

ensure the correct identification of teats. Also, output smoothing could be applied to attempt to prevent any outputs of the system that do not correspond to teats from being passed to the robotic teat cup applicator. Thermal imaging could be applied to determine regions in the images that do not correspond to the udder region of the cow. This region has a greater external temperature than other regions on the surface of the cow or the robotic manipulator. The temperature of an object identified as a teat could be measured to ensure that it could in fact be a teat. Active illumination should be considered as a means of reducing the complexity of finding stereo point correspondences. A laser grid generator could illuminate points on the teats and the udder that could easily be identified in both images of the stereo pair. Dense disparity maps would allow the surface of the udder, including the teats, to be mapped out in 3D. Active illumination should improve the identification of the teats and the accuracy of the teat position estimates.

The angle extraction algorithm is not robust in determining lines that approximate the sides of a teat. Further development of the line selection stage of the algorithm and the incorporation of the unused line filters that do not rely on the image croppoint should greatly enhance the robustness of the algorithm. The use of different parameters for both the edge detector and the line detector should be investigated with the aim of improving consistency in detecting edges and lines. Pre-processing of images will help eliminate the effects of inconsistent illumination.

The principle of determining the teat angulations based on the location of the central axis of the teat in both images of a stereo pair has been validated by the test results. The algorithm should be adapted to apply this principle to teats of more natural dimensions. In estimating the centreline, the current incarnation of the algorithm approximates the sides of a teat with two straight lines and finds the centreline as the bisector of these lines. The problem of selecting two straight lines to represent the sides of the teat is greatly simplified in the case of cylindrical teats because the teat sides appear as straight edges in the image. It may be necessary to calculate a line of best fit in the case of curved teat sides. Further problems are expected to be encountered with teats that are not symmetrical about a central axis. In this case it is difficult to definitively describe the

angle at which the teat is orientated: Should irregularities be ignored and approximate the teat with an axisymmetric shape? Or, should the entire volume including the irregularities be included in finding a single axis which best represents the orientation of the teat? These are questions that need to be answered in conjunction with testing on actual cow teats. It is suggested that modifying the algorithm to accommodate naturally shaped teats be the next stage in the algorithm development. To avoid redundant effort, it may be sensible not to resolve the issue of robustness until the algorithm is closer to its final embodiment.

Appendix A – Additional Mathematics

A.1 – Useful Conversions

A.1.1 – Fundamental Matrix from Projective Camera Matrices

The Fundamental matrix can be found for a pair of individually calibrated cameras, i.e. there are two cameras each with a known projective camera matrix that transforms points in the World Coordinate Frame to the image plane of the camera. The World Coordinate Frame is the same for both cameras. In this case F can be found directly, expressed in terms of 4x4 determinants composed from rows of the camera projective matrices P and P' [48]. If the camera matrices are defined as:

$$P = \begin{bmatrix} \mathbf{a}^1 \\ \mathbf{a}^2 \\ \mathbf{a}^3 \end{bmatrix} \quad (\text{A.1})$$

$$P' = \begin{bmatrix} \mathbf{b}^1 \\ \mathbf{b}^2 \\ \mathbf{b}^3 \end{bmatrix} \quad (\text{A.2})$$

Then the elements of F are defined as:

$$F_{ji} = (-1)^{i+j} \det \begin{bmatrix} \sim \mathbf{a}^i \\ \sim \mathbf{b}^j \end{bmatrix} \quad (\text{A.3})$$

Where: $\sim \mathbf{a}^i$ is the matrix remaining when the row \mathbf{a}^i is removed from P

$\sim \mathbf{b}^j$ is the matrix remaining when the row \mathbf{b}^j is removed from P'

$$F = \begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix}$$

For example:

$$F_{11} = (-1)^2 \det \begin{bmatrix} \mathbf{a}^2 \\ \mathbf{a}^3 \\ \mathbf{b}^2 \\ \mathbf{b}^3 \end{bmatrix} \quad (\text{A.4})$$

A.1.2 – Decomposing the Camera Matrices

From the following previously defined equations:

Equation 2.14: $P = K[R | \mathbf{t}]$

Equation 2.15 $\mathbf{t} = -R \tilde{C}$

It is seen that:

$$P = K \left[R | -R \tilde{C} \right] = \left[M | -M \tilde{C} \right] \quad (\text{A.5})$$

where:

$$M = KR \quad (\text{A.6})$$

The values of both K and R can be found from a known projective matrix P by decomposing M into the product of an upper triangular matrix and an orthogonal matrix. This decomposition is known as RQ-decomposition [49]. The ambiguity in the decomposition is removed by requiring that K have positive diagonal entries.

A.2 – Basic Geometry

A.2.1 - The Locus of Two Intersecting Lines

A locus of two intersecting lines is one of two groups of points that are the average of the two intersecting lines. These groups of points form a straight line through the point of intersection of the original two lines and bisect the angle between them. This is seen in Fig A.1.

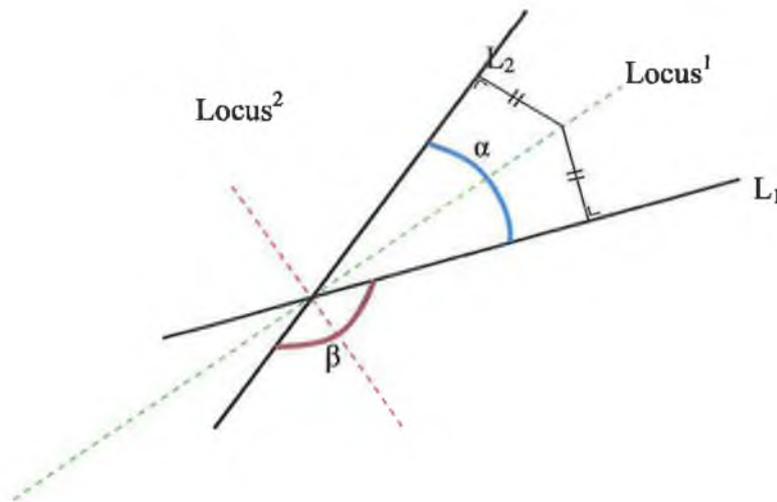


Fig A.1 – Two intersecting lines and their loci

Each point on a locus is the average of pair of points located on L_1 and L_2 . The perpendicular distance from a point to one line is equal to the perpendicular distance to the other. One method of determining the equations of the loci is to use the absolute distance formula by equating the distance of an arbitrary point on the locus to one of the lines with the distance from the same point to the other line. This will provide two possible solutions to the equation of the locus, as is to be expected since there are two possible loci (see Fig A.1). However, this is not the method used throughout this work.

A.2.1.a - Angle between Two Lines

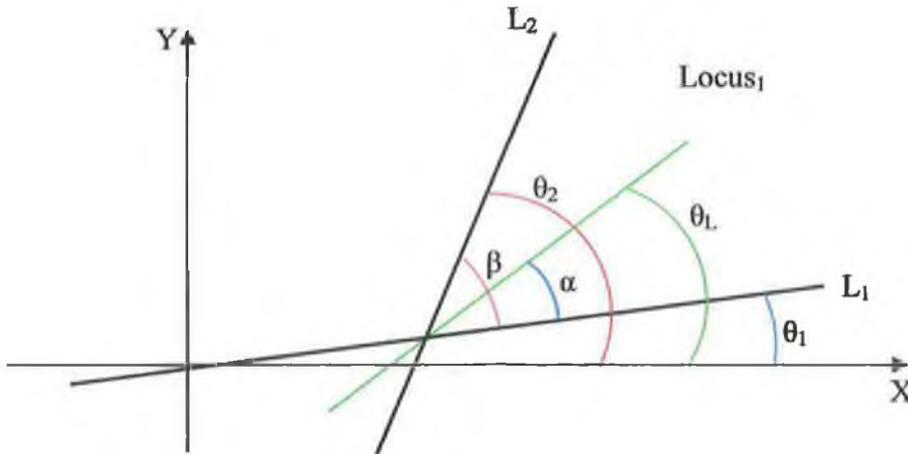


Fig A.2 – Angles of lines with X intercept

From Fig x.2 M_1 , the slope of L_1 , is equal to $\tan \theta_1$ and M_2 , the slope of L_2 , is equal to $\tan \theta_2$. The slope of the locus, M_L , is equal to $\tan (\alpha + \theta_1)$. Half of angle β , the angle between L_1 and L_2 , is equal to α . β is found using the following formula:

$$\tan \beta = \frac{M_2 - M_1}{1 - M_1 M_2} \quad (\text{A.7})$$

Therefore,

$$\alpha = \frac{1}{2} \tan^{-1} \left(\frac{M_2 - M_1}{1 - M_1 M_2} \right) \quad (\text{A.8})$$

and

$$\tan \alpha = \tan \left[\frac{1}{2} \tan^{-1} \left(\frac{M_2 - M_1}{1 - M_1 M_2} \right) \right] \quad (\text{A.9})$$

Once α has been determined the slope of the angle bisector can be calculated:

$$M_L = \tan(\alpha + \theta_1) \quad (\text{A.10})$$

This expression can be expanded to:

$$M_L = \frac{\tan \alpha + \tan \theta_1}{1 - \tan \alpha \tan \theta_1} \quad (\text{A.11})$$

Substituting in previous values it is seen that:

$$M_L = \frac{\tan \alpha + M_1}{1 - \tan \alpha M_1} \quad (\text{A.12})$$

The slope of the angle bisector of interest is now known. In order to determine its line equation a point on the line must be found; this allows the use of the point slope formula:

$$y - y_1 = m(x - x_1) \quad (\text{A.13})$$

The point of intersection of the original two lines must lie upon the new line (the bisector), once this point is found the equation of the line can be evaluated.

A.2.1.b - Point of Intersection of Two Lines

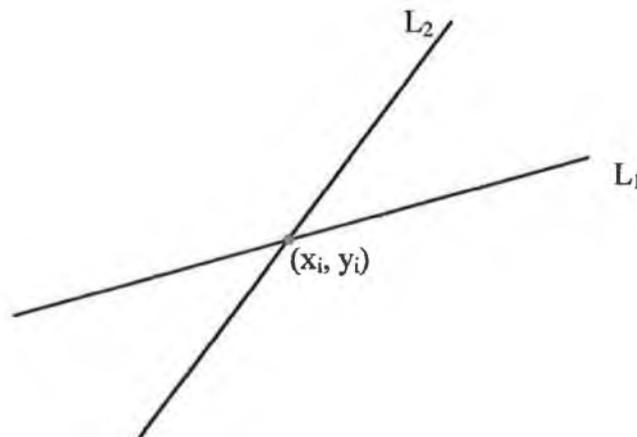


Fig A.3 – Point of Intersection of Two Lines

The point (x_i, y_i) is the point of intersection of the lines L_1 and L_2 . The point lies on each of the lines and therefore the values of ' x_i ' and ' y_i ' must satisfy the following equations:

(Equation of L₁) $a_1x_i + b_1y_i + c_1 = 0$ (A.14)

(Equation of L₂) $a_2x_i + b_2y_i + c_2 = 0$ (A.15)

The constants a_1, b_1, c_1 and a_2, b_2, c_2 are known (the line equations are evaluated). There are only two unknowns (' x_i ' and ' y_i ') and there are two equations. The values of the unknowns are determined by solving this set of simultaneous equations. The following method is one of the ways in which they can be solved (note: this method directly applies only if neither of the lines is vertical). Make ' x_i ' the subject of the second equation:

$$x_i = \frac{-b_2y_i - c_2}{a_2} \quad (\text{A.16})$$

Substitute ' x_i ' value into Equation A.14:

$$a_1 \frac{-b_2y_i - c_2}{a_2} + b_1y_i + c_1 = 0 \quad (\text{A.17})$$

Solve for ' y_i ':

$$y_i = \frac{a_1c_2 - a_2c_1}{a_2b_1 - a_1b_2} \quad (\text{A.18})$$

Solve for ' x_i ':

$$x_i = \frac{-b_2 \frac{a_1c_2 - a_2c_1}{a_2b_1 - a_1b_2} - c_2}{a_2} \quad (\text{A.19})$$

In the event that one of the lines is vertical it will not have a 'b' coefficient of zero. In this case the 'x' component is evaluated directly and can be easily substituted into the other equation to find the 'y' component of the point of intersection.

A.2.1.c - Equation of Bisector

Once a point on the Bisector and the slope of Bisector is known, the equation of this line can be determined using the point slope formula (Equation A.13) giving the general solution, in the form $ax + by + c = 0$, to the bisector equation as:

$$M_L x - y + (y_i - M_L x_i) = 0 \quad (\text{A.20})$$

Substituting in the previously found values (Equation A.18 & Equation A.19) gives:

$$M_L x - y + \left[\left(\frac{a_1 c_2 - a_2 c_1}{a_2 b_1 - a_1 b_2} \right) - M_L \left(\frac{-b_2 \frac{a_1 c_2 - a_2 c_1}{a_2 b_1 - a_1 b_2} - c_2}{a_2} \right) \right] = 0 \quad (\text{A.21})$$

Where:

$$M_1 = \frac{-a_1}{b_1} \quad M_2 = \frac{-a_2}{b_2} \quad M_L = \frac{\tan \alpha + M_1}{1 - \tan \alpha M_1} \quad \tan \alpha = \tan \left[\frac{1}{2} \tan^{-1} \left(\frac{M_2 - M_1}{1 - M_1 M_2} \right) \right]$$

Care must be taken when using this solution if the resulting bisector is either horizontal or vertical as the slope will be zero or infinite respectively. Each of these events results in either the X term or the Y term in the line equation to have a coefficient of zero, this must be accommodated.

A.2.2 - The Normal Representation of a Line:

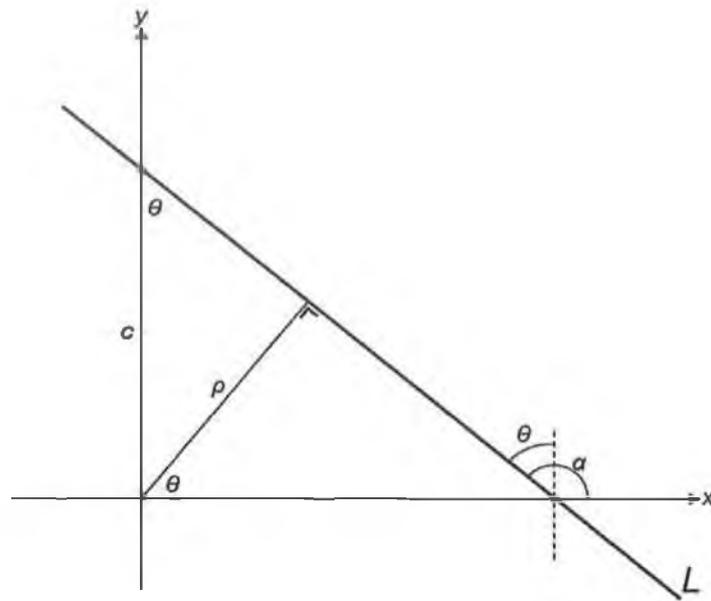


Fig x.x – The line L and it's Normal representation parameters: θ and ρ

In Fig A.4 the line L has a slope m , intercepts the x axis at x_i and intercepts the y axis at y_i . The equation of the line L is:

$$y = mx + c \quad (\text{A.22})$$

The slope of the line is:

$$m = \tan \alpha \quad (\text{A.23})$$

The y intercept is equal to c which is:

$$c = \frac{\rho}{\sin \theta} \quad (\text{A.24})$$

Equation A.22 can now be rewritten:

$$y = x \tan \alpha + \frac{\rho}{\sin \theta} = x \frac{\sin \alpha}{\cos \alpha} + \frac{\rho}{\sin \theta} \quad (\text{A.25})$$

Since

$$\alpha = (\theta + 90^\circ) \quad (\text{A.26})$$

it follows that:

$$\cos \alpha = \cos \theta \cos 90^\circ - \sin \theta \sin 90^\circ = -\sin \theta \quad (\text{A.27})$$

and

$$\sin \alpha = \sin \theta \cos 90^\circ + \cos \theta \sin 90^\circ = \cos \theta \quad (\text{A.28})$$

Therefore substituting back into Equation A.25:

$$y = x \frac{\cos \theta}{-\sin \theta} + \frac{\rho}{\sin \theta} \quad (\text{A.29})$$

From this the Normal Representation of a line follows:

$$x \cos \theta + y \sin \theta = \rho \quad (\text{A.30})$$

Appendix B – Apparatus Setup

B.1 – Calibration of Cameras

B.1.1 – Calibration of Ice Robotics System

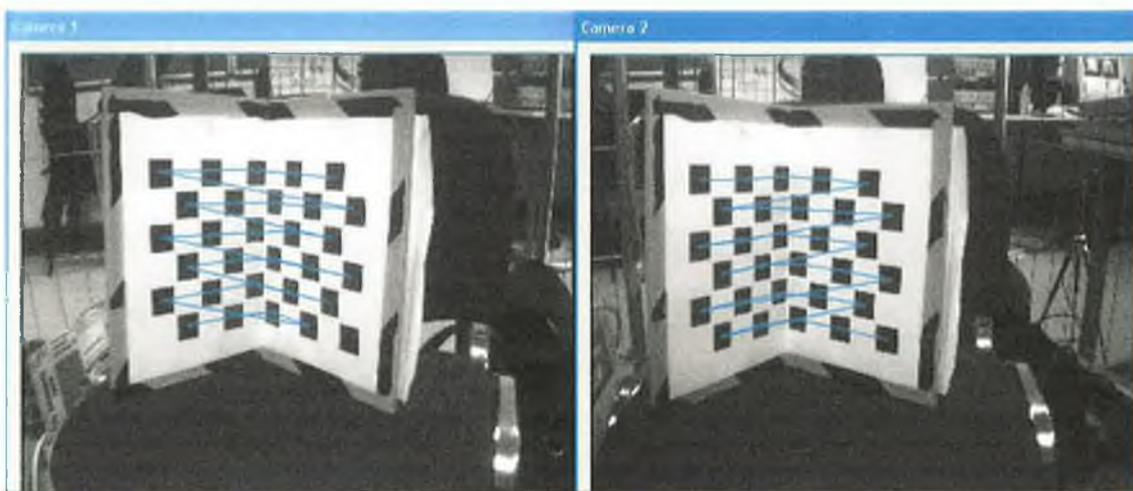


Fig B.1 – View from cameras during Ice Robotics Calibration Routine

The Ice Robotics System is calibrated using the calibration pattern seen in Fig B.1. It consists of two orthogonal planes, each having fifteen black squares in a chessboard pattern. The calibration pattern is positioned approximately 0.7m from the midpoint of the baseline of the stereo-rig. The focus is adjusted to produce crisp edges in the images of the calibration pattern. The scene is illuminated and the calibration routine is run. Information regarding the calibration algorithm is not available. During the calibration routine there are lines and dots overlaid on the video output. The routine is seen to locate the centroid of each square and join them with a line, as seen in Fig B.1. There is a set of 6 parallel lines on one plane with a corresponding set of 6 parallel lines on the other plane. The principal that in projective geometry parallel lines intersect at infinity may be used. The calibration board is constructed so that the two planes are orthogonal to each other. The angles between corresponding lines from the two planes, along with information regarding the relative positions and the dimensions of the squares, may be

used in recovering metric data, allowing measurements of the scene to be made by the system.

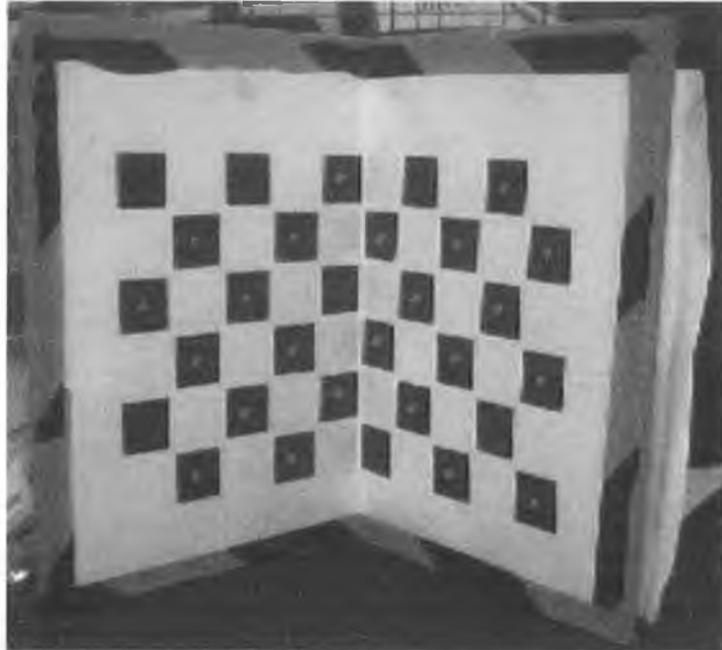


Fig B.2 – Red dots overlaid on centroid of squares

Fig B.2 is an output frame during the calibration sequence prior to displaying the lines that join the square centroids. At the instance captured in this image not all of the squares have been marked with a red dot. Once the calibration routine has successfully identified sufficiently many squares at a single instant the calibration is complete and the camera projective matrices for the left and right cameras are outputted.

$$P_{left} = \begin{bmatrix} 658.212 & -63.735 & -327.995 & -201420.312 \\ 130.866 & -700.850 & 160.261 & -107417.136 \\ 0.771 & -0.080 & 0.632 & -675.832 \end{bmatrix} \quad \text{(B.1)}$$

$$P_{left} = \begin{bmatrix} 716.741 & -60.486 & -136.414 & -214300.826 \\ 117.985 & -686.843 & 179.583 & -99745.522 \\ 0.613 & -0.120 & 0.781 & -636.156 \end{bmatrix} \quad \text{(B.2)}$$

The Fundamental matrix is constructed from P_{left} and P_{right} , each element of F is the determinant of a 4x4 matrix made up of a combination of two rows from each projective matrix as described in Appendix A.1.1.

The Fundamental matrix is:

$$F_{icc} = \begin{bmatrix} -1119338.229 & -8351817.605 & 4623811745.102 \\ 22080246.666 & -3842270.527 & -46853623547.006 \\ -5436382304.916 & 45624867005.477 & -219165010882.963 \end{bmatrix} \quad (\text{B.3})$$

B.1.2 – Caltech Camera Calibration Toolbox for Matlab

The California Institute of Technology have a freely distributable calibration toolbox, which integrates with Matlab, available for download along with detailed instructions on the calibration process [50]. The toolbox uses iterative non-linear methods to accurately account for lens distortion in the estimation of the camera matrices. World points are acquired to sub-pixel accuracy from a set of images containing a chess board calibration pattern. A stereo rig can be calibrated using the toolbox as described in the online tutorial '*Fifth calibration example - Calibrating a stereo system, stereo image rectification and 3D stereo triangulation*' [51]. Once calibrated, the toolbox provides functions for triangulating 3D World points from point correspondences in stereo pairs of images.

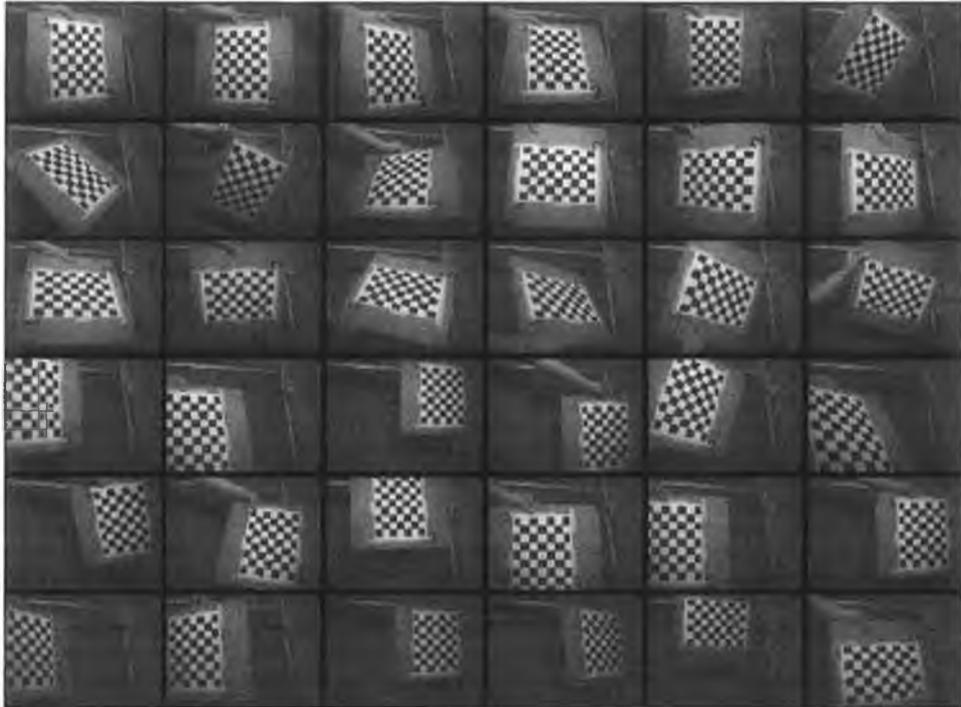


Fig B.3 – Mosaic of the different calibration pattern orientation from left cameras viewpoint

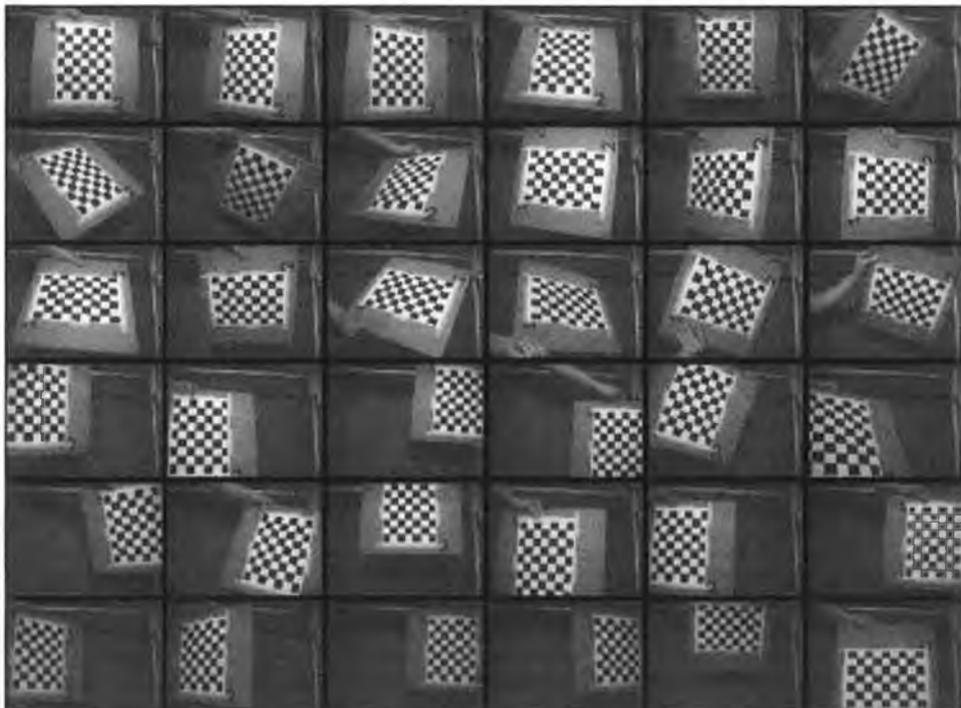


Fig B.4 – Mosaic of the different calibration pattern orientation from right cameras viewpoint

Fig B.3 and Fig B.4 contain the left and right calibration images respectively. There are 36 stereo pairs of images. Each pair contains the left and the right camera view of the calibration pattern. Both the location and the orientation of the pattern move from each stereo pair to the next. The dimensions of the squares in the pattern are entered in the software. This information is used in estimating the intrinsic parameters of the camera. With manual guidance from a user, the tool is able to identify the location and orientation of the pattern in the images. The user clicks on the outer corners of the pattern and initial estimates of all the square corners are made. The user confirms that the estimates are in close proximity to the actual corner locations and the software refines its estimates.

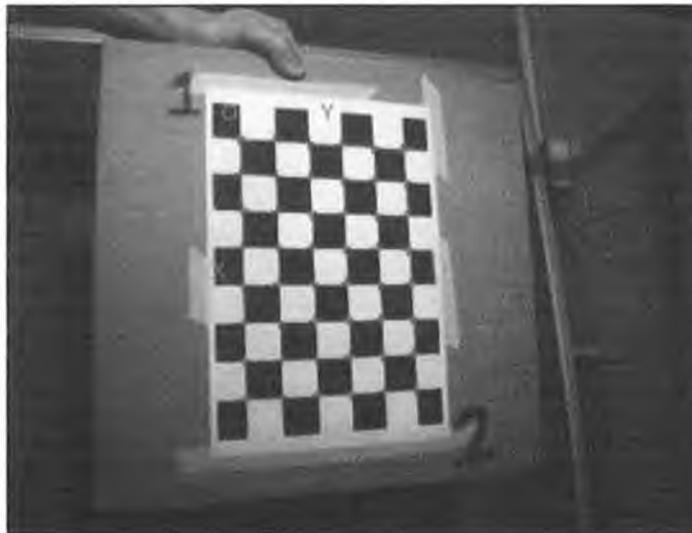


Fig B.5 – Initial estimate of corners indicated by red '+'

The red squares indicate the software's initial estimate as to the location of the corners. The initial estimate is then refined to sub-pixel accuracy by the software as seen in Fig B.6.

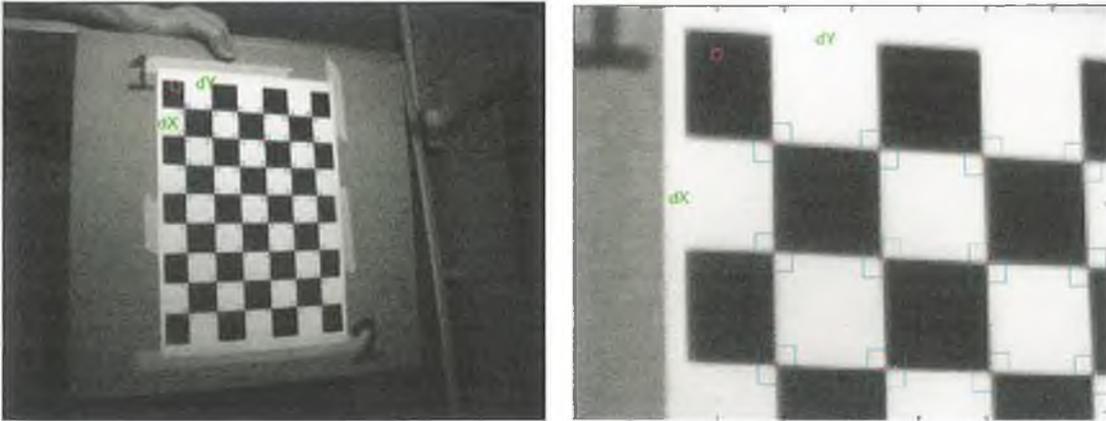


Fig B.6 – Extracted corners, zoomed in view on right

Once the corners have been extracted the projective camera matrix can be solved as in Chapter 2.1.3.a. Fig B.7 illustrates the re-projection of the corner points once the camera has been calibrated. The 36 different locations of the calibration pattern relative to the left camera are shown.

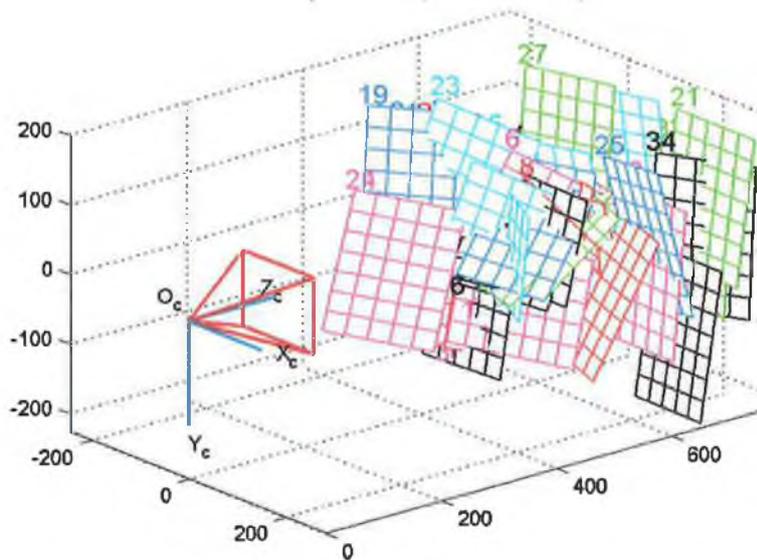


Fig B.7 – Location of calibration pattern in each image with respect to left camera

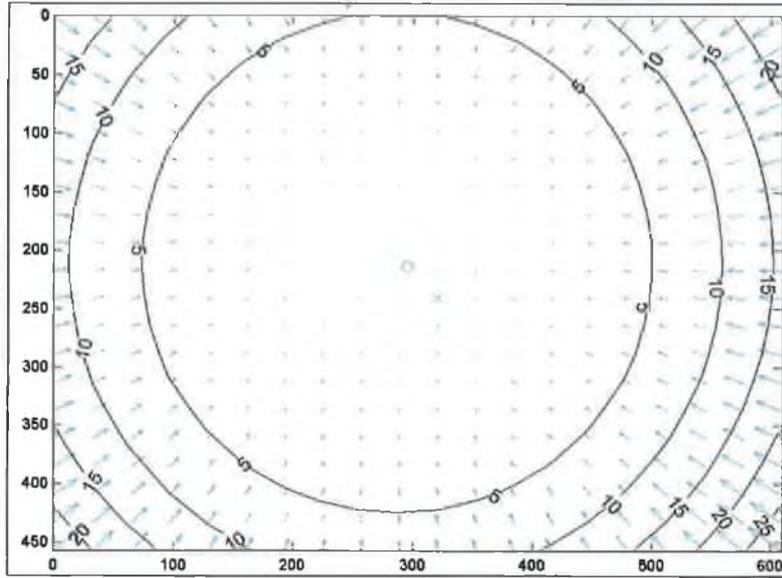


Fig B.8 - Tangential and Radial Distortion Model for left camera

Fig B.8 illustrates the distortion model of the left camera. The arrows show the effect of the distortion. They point from where an image point would lie if there was no distortion to where it is located in the image. It is clear that there is greater distortion further from the image centre. The software makes use of this distortion model by calculating the undistorted image coordinates of the calibration pattern corners and using these points in the calibration estimation.

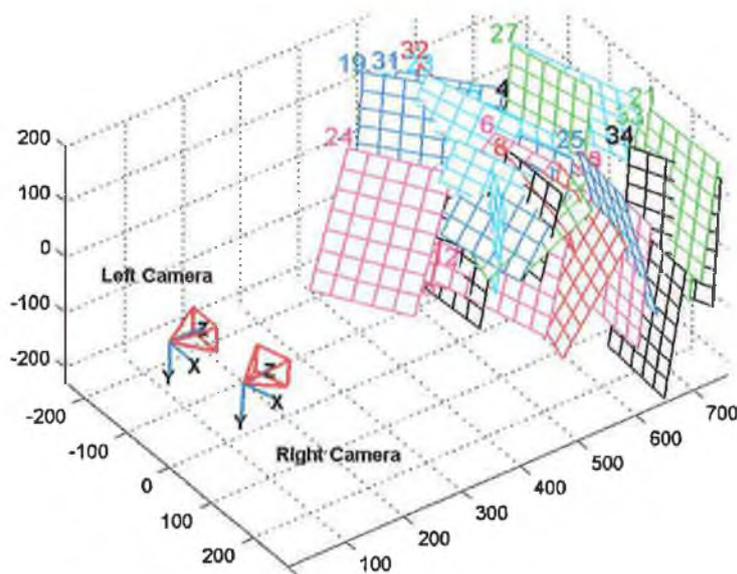


Fig B.9 - Relative positions of cameras and calibration patterns

Once both the right and left cameras have been calibrated individually the results are combined. A rotation vector and translation vector describing the pose of the right camera relative to the left camera is calculated. Fig B.9 illustrates the relative positions of the two cameras and all 36 calibration pattern locations. The output of the calibration tool is given in Table B.1.

Output of Stereo Calibration:

```

Stereo calibration parameters after optimization:

Intrinsic parameters of left camera:

Focal Length:          fc_left = [ 661.97767   662.01837 ] ± [ 2.15814   2.18461 ]
Principal point:      cc_left = [ 300.71582   210.18354 ] ± [ 3.35891   3.27236 ]
Skew:                 alpha_c_left = [0] ± [0]   => angle of pixel axes = 90 degrees
Distortion:           kc_left = [ -0.25653   0.18253   -0.00053   -0.00133   0.00000 ] ±
                        [ 0.00756   0.02593   0.00058   0.00062   0.00000 ]

Intrinsic parameters of right camera:

Focal Length:          fc_right = [ 659.07899   659.28223 ] ± [ 2.37180   2.38052 ]
Principal point:      cc_right = [ 303.78958   287.84193 ] ± [ 2.92129   2.89519 ]
Skew:                 alpha_c_right = [0] ± [0]   => angle of pixel axes = 90 degrees
Distortion:           kc_right = [ -0.24719   0.12789   0.00015   0.00086   0.00000 ] ±
                        [ 0.00565   0.01622   0.00048   0.00048   0.00000 ]

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

Rotation vector:      om = [ 0.11126   0.26092   0.03934 ] ± [ 0.00265   0.00460   0.00109 ]
Translation vector:   T = [-139.43580 -8.38632   3.48797] ± [0.44265 0.297011.40608]

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

```

Table B.1 – Output of Stereo Calibration

$$\text{Rotation vector: } \mathbf{om} = [0.11126 \ 0.26092 \ 0.03934] \quad (\text{units in pixels}) \quad (\text{B.4})$$

$$\text{Translation vector: } \mathbf{T} = [-139.43580 \ -8.38632 \ 3.48797] \quad (\text{units in pixels}) \quad (\text{B.5})$$

The stereo calibration gives the camera matrices:

$$K_{left} = \begin{bmatrix} 661.9777 & 0 & 300.7158 \\ 0 & 662.0184 & 210.1835 \\ 0 & 0 & 1 \end{bmatrix} \quad (\text{B.6})$$

$$K_{right} = \begin{bmatrix} 659.079 & 0 & 303.7896 \\ 0 & 659.2822 & 287.8419 \\ 0 & 0 & 1 \end{bmatrix} \quad (\text{B.7})$$

Placing the World Origin at the left camera centre defines the projective camera matrix of the left camera as:

$$P_{left} = \begin{bmatrix} 661.9777 & 0 & 300.7158 & 0 \\ 0 & 662.0184 & 210.1835 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (\text{B.8})$$

The rotation vector om (Equation B.4) is converted to a rotation matrix (Equation B.9) using the function `rodrigues(om)`. This matrix, combined with the translation vector (Equation B.5), transform the world origin to the camera coordinate frame of the right camera. From here it transformed to the image plane by K_{right} (Equation B.7).

$$R = \text{rodrigues}(om) = \begin{bmatrix} 0.9654 & -0.0244 & 0.2595 \\ 0.0532 & 0.9931 & -0.1047 \\ -0.2552 & 0.1148 & 0.96 \end{bmatrix} \quad (\text{B.9})$$

$$\begin{aligned}
{}^{Left}T_{Right} &= \begin{bmatrix} 1 & 0 & 0 & -139.4358 \\ 0 & 1 & 0 & -8.3863 \\ 0 & 0 & 1 & 3.488 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.9654 & -0.0244 & 0.2595 & 0 \\ 0.0532 & 0.9931 & -0.1047 & 0 \\ -0.2552 & 0.1148 & 0.96 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 0.9654 & -0.0244 & 0.2595 & -139.4358 \\ 0.0532 & 0.9931 & -0.1047 & -8.3863 \\ -0.2552 & 0.1148 & 0.96 & 3.488 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (B.10)
\end{aligned}$$

$$P_{right_full} = \begin{bmatrix} 659.079 & 0 & 303.7896 & 0 \\ 0 & 659.2822 & 287.8419 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} {}^{Left}T_{Right} \quad (B.11)$$

$$P_{right} = \begin{bmatrix} 558.766 & 18.817 & 462.709 & -90839.6 \\ 38.369 & 687.78 & 207.346 & -4524.97 \\ 0.2552 & 0.1148 & 0.96 & 3.488 \end{bmatrix} \quad (B.12)$$

The fundamental matrix is calculated from the left and right projection matrices as described in Appendix A.1.1.

$$F_{Caltech} = \begin{bmatrix} -853056.039 & 1932067.639 & 2071177498.848 \\ 14056407.563 & -6949595.79 & -41692664865.94 \\ -3981261165.233 & 41302462359.322 & -147065393260.38 \end{bmatrix} \quad (B.13)$$

B.1.3 - Finding F directly using Normalized 8-point Algorithm

The ‘8-point Algorithm’, briefly described in Chapter 2.1.3.b, is implemented in a Matlab program by P. D. Kovesi freely available online [52]. The name of the function is `fundmatrix.m`. It accepts a set of point correspondences ($x_i \leftrightarrow x_i'$) as arguments and returns the Fundamental matrix using the ‘Normalized 8-point Algorithm’. Image points

are normalised to improve accuracy. This involves transforming the points in the 2D image plane so that their centroid is located at the origin. This reduces the effects of errors in the image points. Such errors may be introduced by noise. Using the point correspondence found in the images of the previous calibration using the Caltech Toolbox (the corners of the squares) the Fundamental Matrix is calculated as:

$$F_{\text{eight}} = \begin{bmatrix} -0.00000012350826 & 0.00000101988685 & 0.00021910295819 \\ 0.000000172262705 & -0.00000024934524 & -0.00772803916520 \\ -0.00053564039248 & 0.00717295899661 & 0.03093373230607 \end{bmatrix} \quad (\text{B.14})$$

B.2 – Reference Frame Transformations

B.2.1 – IceRobotics Coordinate Outputs

The transformation ${}^{Ice}T_{World}$ converts the position outputs of the IceRobotics vision system into coordinates with respect to the World reference frame. The World reference plane is defined by the Three-Teat Pattern seen in Fig B.10. The location of the origin and the axis directions are indicated.

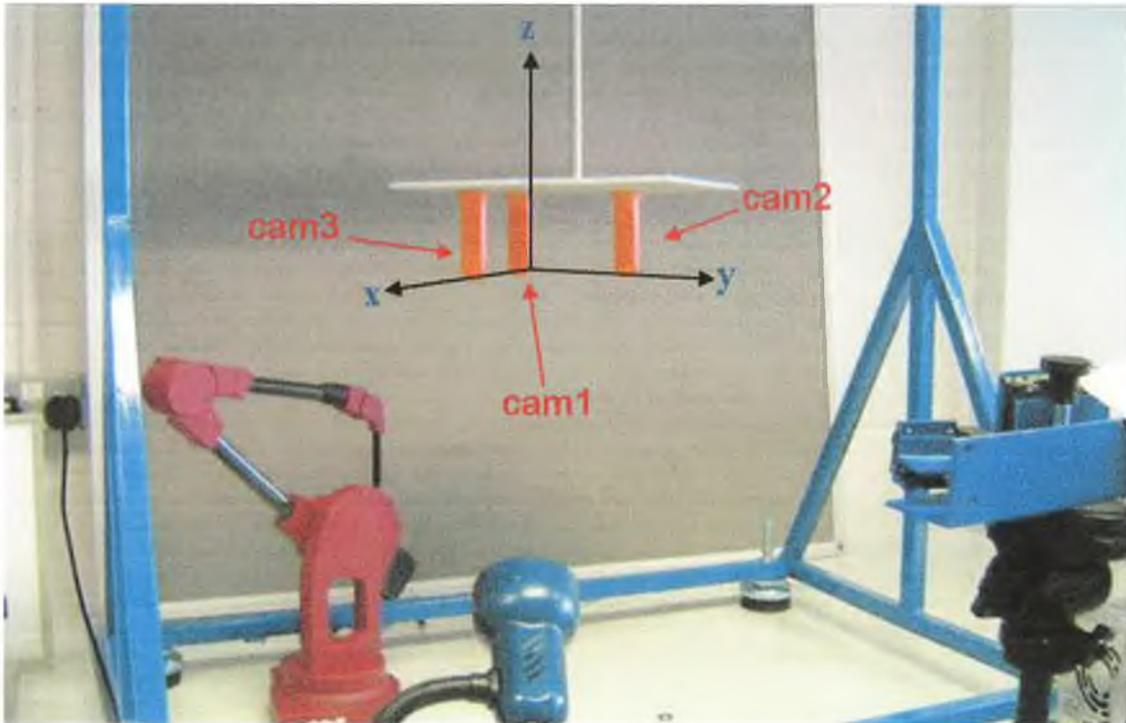


Fig B.10 – 3 Teat Reference Frame

The coordinate values of the three teats, labelled cam1, cam2 and cam3, are shown in Table B.1. Their coordinates (units in millimetres) are given with respect to both the World reference frame and the Camera reference frame. The teat end positions are offset by 12mm in the direction of the positive Z-axis. The coordinates provided by the IceRobotics system for each teat are offset upwards along the teat axis by half the width of the teat ($28/2 = 14\text{mm}$). The location of the World origin is the point in teat end hole to which the tip of the Microscribe lactates. The depth of the hole is 2mm. Therefore the

coordinates cam1, cam2 and cam3 are offset from the world origin by +12mm in the Z-direction.

	Coordinate in World Reference Plane			Coordinate in Camera Reference Plane		
	X	Y	Z	X	Y	Z
Cam1	0	0	12	-43.028	6.458	-647.130
Cam2	0	132	12	-128.938	13.657	-544.858
Cam3	132	0	12	58.573	14.296	-565.228

Table B.2 – Coordinate values of 3 teats with respect to World and Camera reference frames

The transformation matrix ${}^{Ice}T_{World}$ is built as described in Method 3.1 of Chapter 3.2.3.b.

$${}^{Camera}T_{World} = \begin{bmatrix} X_x & Y_x & Z_x & Cam1_x \\ X_y & Y_y & Z_y & Cam1_y \\ X_z & Y_z & Z_z & Cam1_z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \quad (B.15)$$

$${}^{Camera}T_{World} = \begin{bmatrix} 0.7665 & 0.0609 & -0.6396 & 446.4793 \\ -0.6293 & 0.0549 & -0.7754 & 474.3580 \\ -0.0121 & 0.9968 & 0.0804 & -58.9594 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (B.16)$$

The IceRobotics systems two views of the Three-Teat pattern are seen in Fig B.11.

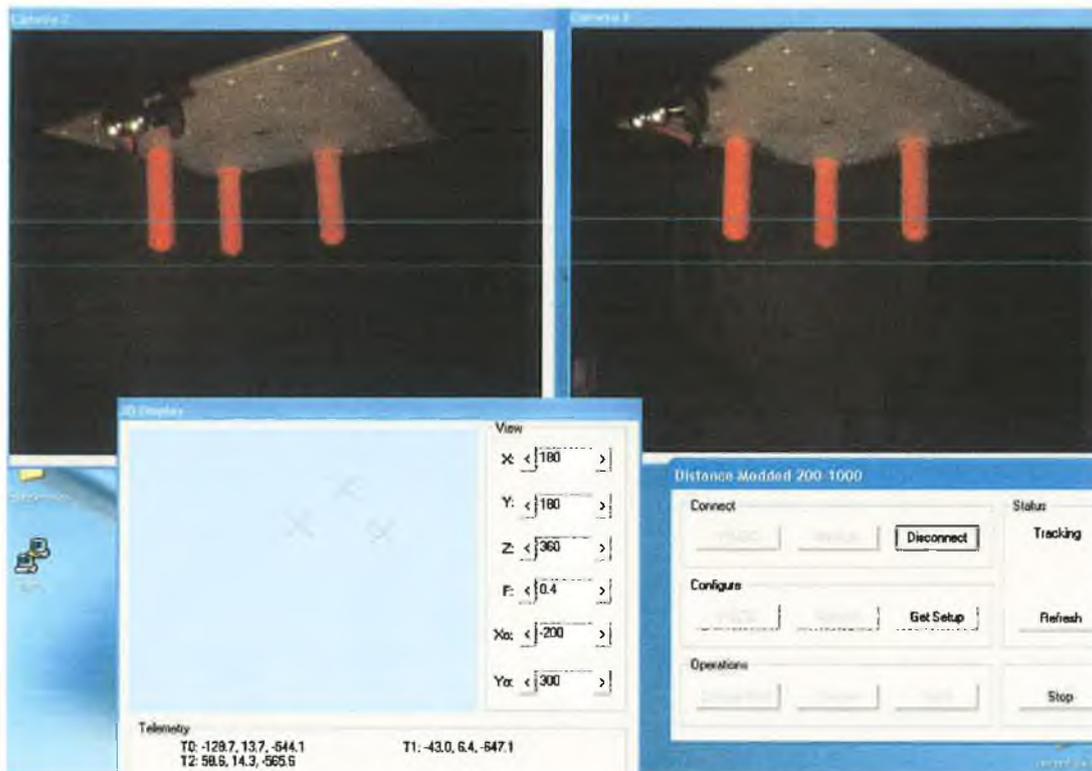


Fig B.11 – IceRobotics output: views of Three-Teat pattern

B.2.2 – Angle Extraction Algorithm

A new World reference frame is defined for the performance tests of the angle extraction algorithm. This coordinate frame is distinct from the World reference frame of Appendix B.2.1.

B.2.2.a - World Reference Frame

The World Reference Frame is defined by the 3 planar points marked on the Reference Frame Board pictured in Fig B.12 and Fig B.13. The top point, labelled 'O' is the origin of the coordinate frame. The direction of the X axis is defined by the vector from the origin to point 'X' on the board. The direction of the Y axis is defined by the vector from the origin to the point 'Z'.



Fig B.12 - Left & Right View of 'Reference Frame Board' representing World Reference Frame

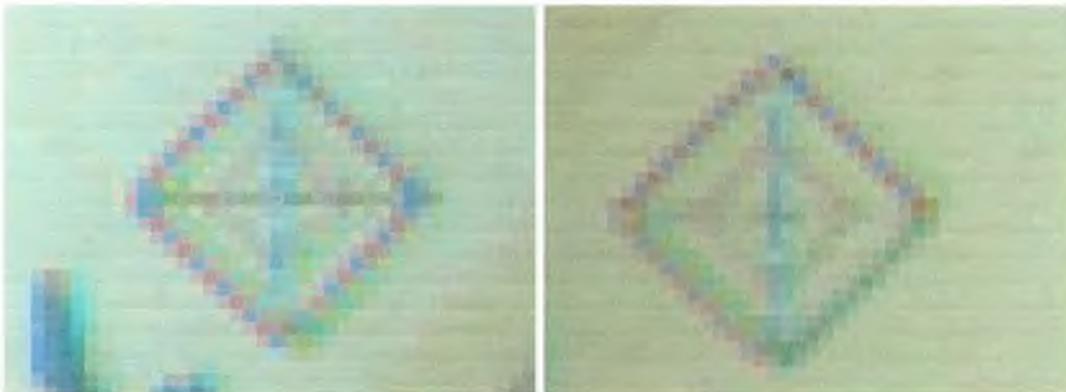


Fig B.13 - Zoomed View of 'Y' target in Left & Right View

B.2.2.b - Microscribe Reference Frame

The Microscribe coordinate frame is setup such that it coincides with the World Reference Frame. This is achieved using the Microscribe calibration software. Fig B.14 illustrates the choosing of the 'Y' target of the Reference Frame Board.

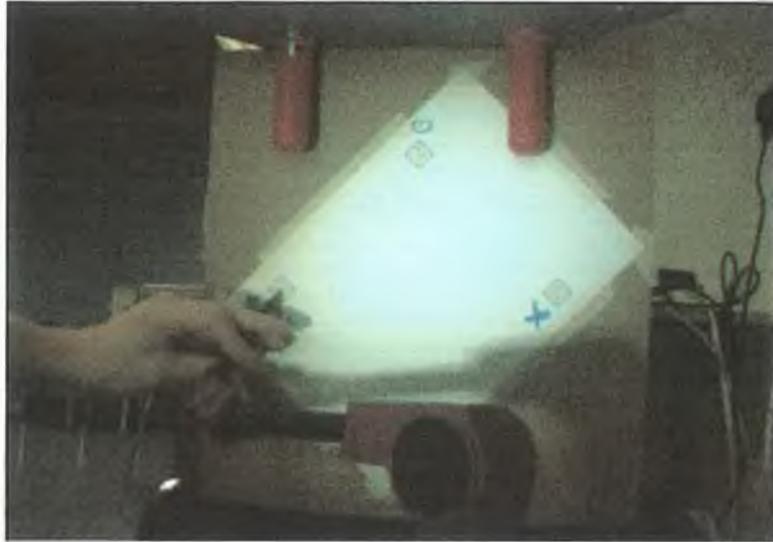


Fig B.14 - Setting Coordinate Frame of Microscribe

B.2.2.c - Test Rig Reference Frame

This reference frame aligns with the visual reference frame provided by the structure of the Test Rig. Fig B.15 illustrates the orientation of the visual reference frame in relation to the test rig. The axes align with the straight edges of the rig. Fig B.16 demonstrates how three teats arranged in a right angle are used to determine a transformation from the Microscribe reference frame (same as World frame) to the Test Rig reference frame.

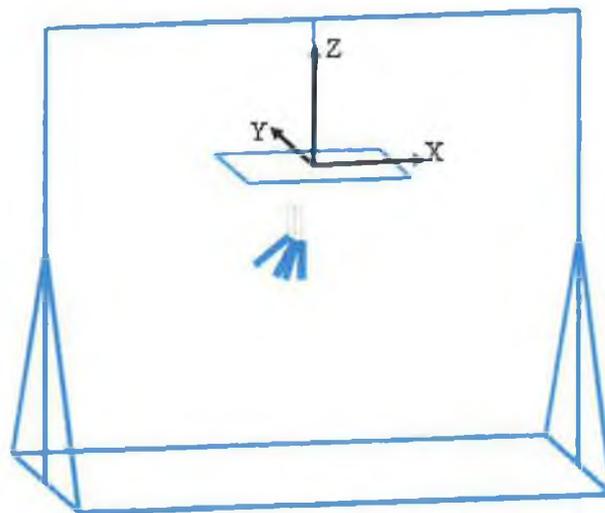


Fig B.15 - Test Rig Reference Frame

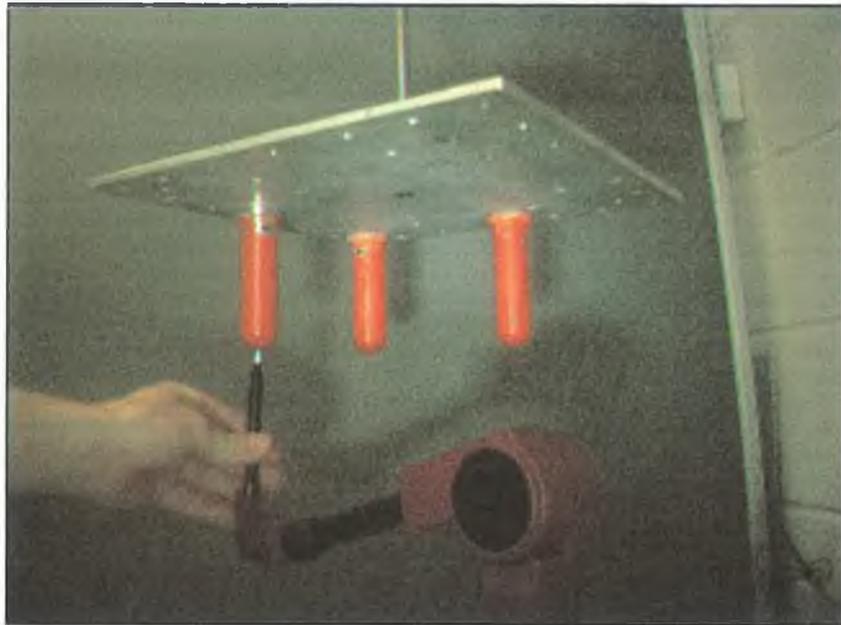


Fig B.16 - Establishing Coordinate Frame of Test Rig using 3 Teat Pattern

B.2.2.d - Camera Coordinate Frame

The coordinates given out by the camera are found using stereo triangulation. The reference frame of these coordinates is defined as the origin of the chessboard calibration pattern in the first pair of stereo images in a series of stereo images. In order to relate measurements found using the camera to known physical locations the relationship between the camera coordinate frame and the world coordinate frame is established. Fig B.13 is a zoomed view of the target points representing the World Coordinate Frame. The 3D location of these points is triangulated using the cameras and the stereo triangulation function of the Caltech Matlab tool [53]. From these coordinates a transformation can be calculated to map points from the camera frame to the world frame.

B.2.2.e - Coordinate Frame Conversions

Camera → World / Microscribe → 3 Teat Pattern → Test Rig Frame

	Camera Reference Frame			World Reference Frame		
	X	Y	Z	X	Y	Z
X Target	117.256	-110.24	0.945	160.5	0	0
Y Target	37.8232	39.2406	-74.555	0	160.5	0
Origin Target	672.784	648.018	676.825	0	0	0

Table B.3 – Coordinates of Reference Frame Board Targets with respect to both coordinate systems

	World Reference Frame		
	X	Y	Z
Right Teat (X)	171.092	27.5434	-16.244
Left Teat (Y)	38.9613	160.533	-25.647
Centre Teat(Z)	99.4397	95.5966	72.3418

Table B.4 – Coordinates of Three Teat pattern with respect to the World Reference Frame

Table B3 and Table B4 contain the coordinate information necessary to construct the transformation matrices of Equations B.18 and B.19. They are constructed using Method 3.1 described in Chapter 3.2.3.b. Equation B.17 is constructed as a rotation of 45° about the Z-axis and a simple translation to align with the visual reference frame of the test rig.

3-Teat Pattern to Test Rig Reference Frame:

$${}_{3\text{Teat}}T_{\text{Rig}} = \begin{bmatrix} 0.7071 & -0.7071 & 0 & 56.5685 \\ -0.7071 & -0.7071 & 0 & 56.5685 \\ 0 & 0 & 1 & -250 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (\text{B.17})$$

World / Microscribe to 3-Teat Pattern:

$${}_{\text{World}}T_{3\text{Teat}} = \begin{bmatrix} 0.5380 & -0.5108 & -0.6706 & 43.8355 \\ -0.4553 & 0.4891 & -0.7440 & 52.3367 \\ 0.7080 & 0.7056 & 0.0306 & -140.0671 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (\text{B.18})$$

Camera to World:

$$\text{Camera } T_{\text{World}} = \begin{bmatrix} 0.7182 & 0.6954 & -0.0252 & 68.1967 \\ -0.6869 & 0.7045 & -0.1782 & 173.7885 \\ -0.1062 & 0.1453 & 0.9837 & -654.8465 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (\text{B.19})$$

By combining the previous transformations the following two transformations are found:

Microscribe/World to Rig:

$$\text{World } T_{\text{Rig}} = \begin{bmatrix} 0.7024 & -0.7070 & 0.0519 & -6.0112 \\ -0.0585 & 0.0153 & 1.0002 & 11.9959 \\ -0.7080 & -0.7056 & -0.0306 & -109.9329 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (\text{B.20})$$

Camera to Rig:

$$\text{Camera } T_{\text{Rig}} = \begin{bmatrix} 0.9846 & -0.0022 & 0.1594 & -114.9859 \\ -0.1588 & 0.1154 & 0.9826 & -644.3310 \\ -0.0206 & -0.9939 & 0.1134 & -260.7888 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (\text{B.21})$$

These two transformations are used to make direct comparisons between angles measured using the Microscribe and using the angle extraction algorithm. The comparisons are made in the Visual Reference Frame defined by the physical orientation of the test rig.

Appendix C – Software Code

C.1 – Angle Extraction Algorithm

Function Name	Called By	Inputs	Outputs	Comments
auto_testangle		im_left im_right calibdata transformdata clickpoints	angle_returns	main angle extraction program
pack_cropconvert	auto_testangle	im_left im_right cropsize clickpoints	crop_left crop_right cropdata cropsize clickpoints	convert images to greyscale and crop to region of interest
findlines	auto_testangle	edge_left edge_right crop_left crop_right	lines_left lines_right	find lines in images
pack_hough	findlines	f dtheta drho	h theta rho	Hough Transform Function: Gonzalez, R. C., Woods, R. E. and Eddins, S. L. (2004). "Image Segmentation", Digital Image Processing using Matlab, Pearson Prentice Hall, New Jersey.
pack_houghpeaks	findlines	h numpeaks threshold rho0	r c hnew	Find peaks in Hough Transform: Gonzalez, R. C., Woods, R. E. and Eddins, S. L. (2004). "Image Segmentation", Digital Image Processing using Matlab, Pearson Prentice Hall, New Jersey.
pack_houghlines	findlines	f theta rho r cc filgap minlength	lines	Match Lines to edge pixels: Gonzalez, R. C., Woods, R. E. and Eddins, S. L. (2004). "Image Segmentation", Digital Image Processing using Matlab, Pearson Prentice Hall, New Jersey.
pack_linefilter	auto_testangle	lines_left lines_right cropdata cropsize	returnline_left returnline_right	Find lines to represent the sides of the test
setsideofcropmatrix	pack_linefilter	lines_data localcroppoint x_value_at_ycrop	sideofcropmatrix	subroutine - which side of the test is the line located?
equationcal	pack_linefilter	lines_data localcroppoint	line_length midpoint midpoint_distance lines_equations slope x_value_at_ycrop	subroutine - calculate information regarding the lines
perpendistances	pack_linefilter	lines_data midpoint lines_equations localcroppoint slope	normal_distance max_norm_distance normal_crop_distance	subroutine - calculate the perpendicular distances of all lines to the croppoint
filter_min_midpoint	pack_linefilter	inoutmatrix lines_data midpoint_distance mean_midpoint_distance	inoutmatrix	linefilter - eliminate lines with midpoint to croppoint distances that are too small
filter_normfromcrop	pack_linefilter	inoutmatrix lines_data normal_crop_distance input_maxdistance	inoutmatrix	linefilter - eliminate lines with too small a perpendicular distance from the crop point
filter_farfromcrop	pack_linefilter	inoutmatrix lines_data normal_crop_distance	inoutmatrix	linefilter - eliminate lines with too large a perpendicular distance from the crop point
filter_maxnormpair	pack_linefilter	inoutmatrix lines_data normal_distance	inoutmatrix	linefilter - chooses two lines with greatest perpendicular distance from each other

filter_maxnormpairleftright sideofcropmatrix	pack_linefilter	inoutmatrix lines_data normal_distance normal_crop_distance	inoutmatrix	linefilter - choses two lines with greatest perpendicular distance from each other but they must be on opposite sides of the teat
filter_avg_slope	pack_linefilter	inoutmatrix lines_data lines_equations mean_slope	inoutmatrix	linefilter - eliminate lines having a slope that differs too greatly from the average slope of the lines
filter_avg_length	pack_linefilter	inoutmatrix lines_data line_length mean_length	inoutmatrix	linefilter - eliminate lines having a length that
filter_leastpointdistance	pack_linefilter	inoutmatrix lines_data croppoint Input_maxdistance	inoutmatrix	linefilter - eliminate lines that do not have an endpoint within a certain distance of the croppoint
pack_tranlatelines	auto_teatangle	left_linepair right_linepair cropdata	left_linepair_T right_linepair_T	convert line equations from positions in cropped images to positions in original images
pack_epipolarcentreline	auto_teatangle	leftline rightline image_left image_right cropdata F	left_bisectline right_bisectline lowpoint_uv_left highpoint_uv_right	determining the centreline of the teat in stereo pair and choose point correspondences using epipolar geometry
calculate_angle	auto_teatangle	lowpoint_uv_left highpoint_uv_right om T fc_left fc_right cc_left cc_right kc_left kc_right alpha_c_left alpha_c_right cam_T_scribe worldTrig	theta_YZ theta_XZ	calculate the teat angles
stereo_triangulation	calculate_angle	XL XR	xL xR om T fc_left cc_left kc_left alpha_c_left fc_right cc_right kc_right alpha c right	Caltech Calibration Toolbox For Matlab - (c) Jean-Yves Bouguet - Intel Corporation - April 9th, 2003

C.2 – Utilities

Function/Script Name	Called By	Inputs	Outputs	Comments
decompose	-	P	R q c_ EXT	Factorise a Projective camera matrix
coordframe.m	-	-	-	script to setup 3D test rig display
pointplot.m	Microscribe	-	-	Script called upon an coordinate input from Microscribe
teatplot.m	pointplot.m	-	-	display teat using microsibe coordinates
angulate.m	pointplot.m	-	-	calcalate teat angle
displayangle.m	angulate.m	-	-	script to display in 3D the teat orientation

All source code for the functions listed here, and for all other applications developed in the course of the project, is available upon request. Contact the Department of Mechanical and Manufacturing Engineering, Dublin City University for details.

References

¹ O'Brien, B., et al, 2001, "*Profiling the working year on Irish dairy farms - identification of some work areas towards improvement in efficiency*", Proceedings of Irish Grassland Association Dairy Conference, Cork, September 11-12, 2001. Pages 112-124. [*Labour studies show the milking process to account for 37% of total dairy labour over 12 month period*].

² O'Callaghan, E., et al, "*Milking Facilities and Labour*",
<http://www.teagasc.ie/publications/2001/ndc/ndc-callaghan.htm> [last visited: 2/06/06].

³ Mottram, T. (1997), Requirements for teat inspection and cleaning in automatic milking systems, "*Computers and electronics in agriculture*", Issue 17 (1997), 63 – 77.

⁴ Research Institute for Animal Husbandry:
<http://www.automaticmilking.nl/projectinformation/General.asp> [last visited: 2/06/06].

⁵ Official DeLaval Website,
http://www.delaval.com/Products/Automatic-Milking-Robotic-milking/DeLaval_VMS/default.htm [last visited: 2/06/06].

⁶ Image: <http://www.delaval.com/NR/rdonlyres/AA98F3E4-715F-4355-B175-DC3C4C3CAD0D/0/VMS.jpg> [link last checked 2/06/06].

⁷ Internal Report, Review Document, 2006, "*Teat Location for a Rotary Carousel Automatic Milking System*", Department of Mechanical and Manufacturing Engineering, Dublin City University

⁸ RMS (Robotic Milking Solutions) official website, <http://www.roboticmilking.com> [last visited: 4/06/06].

⁹ Lely - official website, <http://www.lely.com> [last visited: 4/06/06].

¹⁰ SAK – official website, <http://www.sak.dk> [last visited: 4/06/06].

¹¹ Fullwood – official website, <http://www.fullwood.com/merlin.php> [last visited: 4/06/06].

¹² Image: http://www.delaval.com/NR/rdonlyres/4295E840-4045-4127-9F47-517EAAC40904/0/Rotary_PER_web.jpg [link last checked 2/06/06].

¹³ Image used with permission. White James, (2006), Masters Thesis, Design of a robotic manipulator for automatic application of milking cups, Department of Mechanical and Manufacturing Engineering, Dublin City University.

¹⁴ Kuczaj, M et al. (2000). Relations Between Milk Performance and Udder Dimensions of Black-White Cows imported from Holland, "*Electronic Journal of Polish Agricultural Universities, Animal Husbandry*", Volume 3, Issue 2.

¹⁵ Internal Report, 2006, Existing Technologies and Patent Reviews of AMS, Department of Mechanical and Manufacturing Engineering, Dublin City University.

¹⁶ Lars, A et al. (1998) International Patent, W098/47348, Apparatus and method for recognising and determining the position of a part of an animal.

¹⁷ Clarke, T.A. & Fryer, J.F. 1998. The development of camera calibration methods and models, *Photogrammetric Record*, 16(91): pp 51-66.

¹⁸ Brown, D.C. (1966). Decentering distortion of lenses, "*Photogrammetric Engineering*", 32(3): pp444-462.

¹⁹ Tsai, Rodger Y. 1987. Versatile camera calibration technique for high-accuracy 3D machine vision metrology using off the shelf TV cameras and lenses, "*IEEE Journal of Robotics and Automation*", vol. 3, issue 4, Aug 1987, p323-344.

²⁰ Weng, J., Cohen, P. and Hemiou, M. (1992). Camera Calibration with Distortion Models and Accuracy Evaluation, "*IEEE Transactions on Pattern Analysis and Machine Vision Intelligence*", Vol. 14, No. 10, Oct. 1992, pp. 965-980.

²¹ Hartley. R. (1997). In Defense of the Eight-Point Algorithm, "*IEEE Transactions on Pattern Analysis and Machine Intelligence*", Vol. 19, No.6, June.

²² Hartley, R and Gupta, R. (1993). Computing Matched-Epipolar Projections, "*Proceedings of IEEE Conference in Computer Vision and Pattern Recognition*", 1993 pp 549-555.

²³ Zhang et al. (1995), A Robust Technique for Matching Two Uncalibrated Images through the Recovery of the Unknown Epipolar Geometry, "*Artificial Intelligence*", Journal, Vol. 78, Oct 1995, pp87-119.

-
- ²⁴ Woetzel, J. And Koch, R. (2004) .Real-time multi-stereo depth estimation on GPU with approximate discontinuity handling, "*IEE Conference Publication, 1st European Conference on Visual Media Production*", CVMP 2004, pp 245-254.
- ²⁵ Hartley, R. and Zisserman, A. (2003), "*Multiple View Geometry in computer vision*", Cambridge University Press, Cambridge.
- ²⁶ Faugeras, O. (1993), "*Three Dimensional Computer Vision A Geometric Viewpoint*", Massachusetts Institute of Technology, Massachusetts.
- ²⁷ Caltech *Camera Calibration Toolbox for Matlab* by Jean-Yves Bougue ,
http://www.vision.caltech.edu/bouguetj/calib_doc/index.html (checked 1st May 2006)
- ²⁸ Hartley, R. and Zisserman, A. (2003). "Camera Models". *Multiple View Geometry*, (2nd ed.), Cambridge University Press, Cambridge, pg 154.
- ²⁹ Hartley, R. and Zisserman, A. (2003). "Epipolar Geometry and the Fundamental Matrix". *Multiple View Geometry*, (2nd ed.), Cambridge University Press, Cambridge, pg 243.
- ³⁰ Hartley, R. and Zisserman, A. (2003). "Epipolar Geometry and the Fundamental Matrix". *Multiple View Geometry*, (2nd ed.), Cambridge University Press, Cambridge, pg 263.
- ³¹ Hartley, R. and Zisserman, A. (2003). "Computation of the Camera Matrix P". *Multiple View Geometry*, (2nd ed.), Cambridge University Press, Cambridge, pg 182.
- ³² Atkinson, K.E. (1989). "*An Introduction to Numerical Analysis*", (2nd ed.), Wiley and Sons, New York.
- ³³ Gonzalez, R. C., Woods, R. E. and Eddins, S. L. (2004). "Image Segmentation", *Digital Image Processing using Matlab*, Pearson Prentice Hall, New Jersey, pp 378 – 392.
- ³⁴ Canny, J. (1986). "A Computational Approach for Edge Detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 8, Issue 6, pp. 679 – 698.

-
- ³⁵ Parker, J. R. (1997). "ADVANCED EDGE DETECTION TECHNIQUES: The Canny and the Shen-Castan Methods", *Algorithms for image processing and computer vision*, John Wiley & Sons, Inc., New York, pp23.
- ³⁶ Gonzalez, R. C., Woods, R. E. and Eddins, S. L. (2004). "Image Segmentation", *Digital Image Processing using Matlab*, Pearson Prentice Hall, New Jersey, pg 389.
- ³⁷ Gonzalez, R. C. and Woods, R. E. (2002). "Image Enhancement in the Frequency Domain", *Digital Image Processing*, (2nd ed.), Pearson Prentice Hall, New Jersey, pp175 – 180.
- ³⁸ Gonzalez, R. C. and Woods, R. E. (2002). "Image Enhancement in the Frequency Domain", *Digital Image Processing*, (2nd ed.), Pearson Prentice Hall, New Jersey, pg 118.
- ³⁹ Gonzalez, R. C., Woods, R. E. and Eddins, S. L. (2004). "Image Restoration", *Digital Image Processing using Matlab*, Pearson Prentice Hall, New Jersey, pg 146.
- ⁴⁰ Hough, P.V.C. (1962). "Methods and Means for Recognizing Complex Patterns", U.S. Patent 3069654.
- ⁴¹ Gonzalez, R. C., Woods, R. E. and Eddins, S. L. (2004). "Image Segmentation", *Digital Image Processing using Matlab*, Pearson Prentice Hall, New Jersey, pg 393.
- ⁴² IceRobotics Official Website, <http://www.icerobotics.co.uk/company.html> , checked 14th May 06.
- ⁴³ Immersion Corporation Website, <http://www.immersion.com/>, checked 27th May 06.
- ⁴⁴ Hartley, R. and Zisserman, A. (2003). "Camera Models". *Multiple View Geometry*, (2nd ed.), Cambridge University Press, Cambridge, pp 158 – 161.
- ⁴⁵ Matlab Image Processing Toolbox, MathWorks
- ⁴⁶ Gonzalez, R et al (2004), "Image Segmentation", *Digital Image Processing using Matlab*, Pearson Prentice Hall, New Jersey, pp 396 – 404.
- ⁴⁷ Caltech Camera Calibration Toolbox for Matlab by Jean-Yves Bouguet
http://www.vision.caltech.edu/bouguetj/calib_doc/index.html (checked 1st May 2006)

⁴⁸ Hartley, R. and Zisserman, A. (2003). "N-Linearities and Multiple View Tensors". *Multiple View Geometry*, (2nd ed.), Cambridge University Press, Cambridge, pp 411-413.

⁴⁹ Heath, M.T. (2002), "Orthogonalization Methods", *SCIENTIFIC COMPUTING An Introductory Survey*, The McGraw-Hill Companies, New York, pp 129.

⁵⁰ Caltech *Camera Calibration Toolbox for Matlab* by Jean-Yves Bougue
http://www.vision.caltech.edu/bouguetj/calib_doc/index.html (checked 1st May 2006)

⁵¹ Caltech *Camera Calibration Toolbox for Matlab* by Jean-Yves Bougue, 'Fifth calibration example - Calibrating a stereo system, stereo image rectification and 3D stereo triangulation',
http://www.vision.caltech.edu/bouguetj/calib_doc/htmls/example5.html (checked 11th May 2006)

⁵² P. D. Kovesi. MATLAB and Octave Functions for Computer Vision and Image Processing. School of Computer Science & Software Engineering, The University of Western Australia. Available from:
<http://www.csse.uwa.edu.au/~pk/research/matlabfns/> (checked 12th May 2006)

⁵³ Caltech *Camera Calibration Toolbox for Matlab* by Jean-Yves Bougue
http://www.vision.caltech.edu/bouguetj/calib_doc/index.html (checked 1st May 2006)