

1-1-1992

Design and optimization of three-dimensional scanning laser range finder

Robert Ben McFarland
University of Nevada, Las Vegas

Follow this and additional works at: <https://digitalscholarship.unlv.edu/rtds>

Repository Citation

McFarland, Robert Ben, "Design and optimization of three-dimensional scanning laser range finder" (1992). *UNLV Retrospective Theses & Dissertations*. 260.
<http://dx.doi.org/10.25669/kguw-rkog>

This Thesis is protected by copyright and/or related rights. It has been brought to you by Digital Scholarship@UNLV with permission from the rights-holder(s). You are free to use this Thesis in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you need to obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/or on the work itself.

This Thesis has been accepted for inclusion in UNLV Retrospective Theses & Dissertations by an authorized administrator of Digital Scholarship@UNLV. For more information, please contact digitalscholarship@unlv.edu.

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

U·M·I

University Microfilms International
A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
313/761-4700 800/521-0600

1870

1871

1872

Order Number 1352554

Design and optimization of 3-D scanning laser range finder

McFarland, Robert Ben, III, M.S.

University of Nevada, Las Vegas, 1993

U·M·I
300 N. Zeeb Rd.
Ann Arbor, MI 48106

**DESIGN AND OPTIMIZATION
OF 3-D SCANNING
LASER RANGE
FINDER**

by

Robert Ben McFarland III

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of
Master of Science
in
Mechanical Engineering

Department of Mechanical Engineering

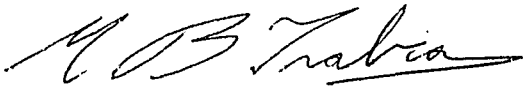
University of Nevada, Las Vegas

January 1993

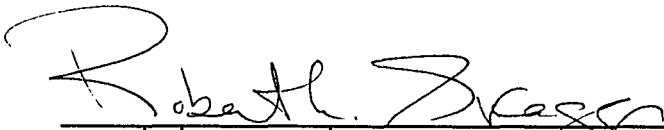
The thesis of Robert B. McFarland for the degree of Masters of Science in Mechanical Engineering is approved.



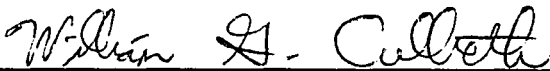
Chairperson, **Woosoon Yim, Ph.D**



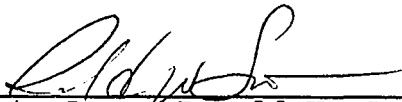
Examining Committee Member, **Mohamed B.E. Trabia, Ph.D**



Examining Committee Member, **Robert Skaggs, Ph.D**



Graduate Faculty Representative, **William G. Culbreth, Ph.D**



Graduate Dean, **Ronald W. Smith, Ph.D**

University of Nevada, Las Vegas

January 1993

ABSTRACT

There are a multitude of different types of range finders being as diverse as the number of applications. A large portion of range finders are based on the principles of triangulation. One of the drawbacks with triangulation principles is the large amount of needless data generated to solve for the range. The purpose of this project is to develop a technique for minimizing the number of iterations needed to determine the XY&Z position for a laser rangefinding system. Using a Charged Coupled Device (CCD) camera to view a robotic work area a user can select an object on a monitor. A LASER is used to project a beam into the work area and the camera in conjunction with image processing software and hardware is used to locate the LASER spot. The purpose of the research in this paper is to devise techniques to minimize the number of points needed to determine XY&Z location without sacrificing accuracy. The techniques developed to reduce the number of iterations worked very well. They were extensively tested in the laboratory to determine the number of iterations needed for a solution and the accuracy of that solution. However, the error for calculating the XYZ position was adversely affected by system errors. These system errors were comprised of the errors associated with positioning motors and with the camera modeling. Changing the configuration could minimize these errors and dramatically improve the accuracy.

TABLE OF CONTENTS

ABSTRACT	iii
LIST OF FIGURES	iiix
LIST OF TABLES	x
LIST OF NOMENCLATURE	xi
1. INTRODUCTION	1
2. DESCRIPTION OF EXPERIMENT	8
2.1 Equipment.....	9
2.1.1 Computer.....	9
2.1.2 Camera/Image Processing.....	10
2.1.3 LASER Projector.....	12
2.2 Limitations and Assumptions.....	15

3. MATHEMATICAL BACKGROUND 18

3.1 Coordinate Systems.....19

3.2 Transformation Matrix.....20

3.3 2-Dimensional Transformation Matrices.....22

 3.3.1 Translation.....22

 3.3.2 Rotation.....24

 3.3.3 2-D Projections.....26

3.4 3-Dimensional Transformation Matrices.....28

 3.4.1 Translation.....32

 3.4.2 Rotations.....32

 3.4.3 About 'X' Axis.....33

 3.4.4 About 'Y' Axis.....33

 3.4.5 About 'Z' Axis.....33

 3.4.6 Perspective.....34

4. CAMERA CALIBRATION 35

4.1 Description of Camera Set-up.....35

4.2 Equations for Camera Placement.....39

4.3 Camera Calibration.....42

 4.3.1 Method of Minimization.....42

 4.3.2 Camera System and Equations.....43

4.4 Data Sets.....47

4.5 Results.....49

4.6 Analysis of Results.....50

5.	X-Y LASER PROJECTOR	53
5.1	Description of Stepper Motor Set-up.....	54
5.2	Equations of Stepper Motors.....	55
5.3	Calibration.....	65
5.3.1	Projector Calibration.....	66
5.3.2	Indexer Motor Calibration.....	68
6.	TRIANGULATION OF NON-INTERSECTING LINES	71
6.1	XYZ Location of Non-Intersecting Lines.....	73
6.2	Development of the Equations.....	74
7.	DETECTION PRINCIPLES	78
7.1	Definition of Regions.....	79
7.2	Optimization.....	80
7.4	Alternate Optimization Method.....	86
8.	RESULTS/ANALYSIS	89
8.1	Collection of Data.....	89
8.2	Methods.....	92
8.3	Analysis of Methods.....	97
9.	CONCLUSION	104

APPENDIX

106

BIBLIOGRAPHY

135

LIST OF FIGURES

FIGURE 1.1	DIRECT RANGE DATA CHART	2
FIGURE 1.2	INDIRECT RANGE DATA CHART	3
FIGURE 2.1	SYSTEM SETUP	8
FIGURE 2.2	CAMERA/POLE SETUP	11
FIGURE 2.3	STEPPER MOTORS SETUP	13
FIGURE 2.4	LASER/COLLAR SETUP	15
FIGURE 2.5	PROJECTOR SETUP	16
FIGURE 3.1a	2-D COORDINATE SYSTEM	19
FIGURE 3.1b	3-D RIGHT HAND COORDINATE SYSTEM	19
FIGURE 3.2	CAMERA COORDINATE SYSTEM	20
FIGURE 3.3	TRANSLATION EXAMPLE	23
FIGURE 3.4	ROTATION EXAMPLE	25
FIGURE 3.5	2-D LINEAR ARRAY	26
FIGURE 3.6	PROJECTION EXAMPLE 1	28
FIGURE 3.7	PROJECTION EXAMPLE 2	29
FIGURE 3.8	PROJECTION EXAMPLE 3	30
FIGURE 3.9	PROJECTION EXAMPLE 4	31
FIGURE 4.1	CAMERA/POLE SETUP	36
FIGURE 4.2	TRANSLATION/ROTATION I	37
FIGURE 4.3	TRANSLATION/ROTATION II	38

FIGURE 5.1	WCS DEFINITION	54
FIGURE 5.2a	WCS DEFINITION	55
FIGURE 5.2b	WCS DEFINITION	55
FIGURE 5.2c	WCS DEFINITION	55
FIGURE 5.3	ROTATION MIRROR 1	57
FIGURE 5.4	ROTATION MIRROR 1	57
FIGURE 5.5	MIRRORS 1 AND 2	59
FIGURE 5.6	REFLECTION OFF OF MIRRORS	61
FIGURE 5.7	CALIBRATION 1	67
FIGURE 5.8	CALIBRATION 2	67
FIGURE 5.9	STEPPER MOTOR ERROR PLOT	69
FIGURE 6.1	DEFINITION OF LINES	71
FIGURE 6.2	DEFINITION OF CROSSING LINES	73
FIGURE 6.3	DEFINITION OF POINTS	74
FIGURE 7.1	DEFINITION OF REGIONS	77
FIGURE 7.2 - 7.7	EXAMPLE 1	80 - 83
FIGURE 7.8	DEFINITION OF SELECTED PIXEL	85
FIGURE 7.9	TRACK OF LASER SPOT ACROSS IMAGE PLANE	86
FIGURE 8.1	TARGET SETUP	89

LIST OF TABLES

TABLE 4.1	DEFINITION OF TRANSFORMATIONS	37
TABLE 4.2	MEASUREMENTS AND TOLERANCES	47
TABLE 4.3	RESULTS OF OPTIMIZATIONS	49
TABLE 4.4	ERROR	51
TABLE 8.1	SAMPLE DATA (ONE POINT)	90
TABLE 8.2	DIFFERENCES (ONE POINT)	92
TABLE 8.3	OPTIMAL LOCATIONS (ONE POINT)	94
TABLE 8.4	ERROR (ONE POINT)	95
TABLE 8.5	2-D ERROR BY RANGE	98
TABLE 8.6	3-D ERROR BY RANGE	99
TABLE 8.7	2-D ERROR OVERALL	100
TABLE 8.8	3-D ERROR OVERALL	100

LIST OF NOMENCLATURE

XYZ (inches)	Data set of points X, Y, & Z in world coordinates. These points are collected to be used as an input in computing xycp.
xpm & ypm (inches)	Data set of points containing the x and y location of the pixel on the image plane in image plane coordinates. These points are collected as an input in conjunction with XYZ.
xpc & ypc (inches)	Data set of points containing calculated x and y values for projections of the XYZ points on to the image plane. This is a calculated data to be used later as an input for error analysis.
xpe & ype (pixels)	Data set of the x and y errors between xypm and xycp. Initially this data set is in inches but is converted to pixels. This data is used as the basis for all error analysis.
magpe	Magnitude of error calculated from xpe and ype.
CEP (pixels)	Circle Error Probable.
WCS	World Coordinate System
CCS	Camera Coordinate System

CHAPTER 1

INTRODUCTION

The purpose of range finders is to determine a distance from a reference point to an object within a sensor's field of view. There are a wide variety range finders that can be tailored in several configurations to meet the requirements of the operational environment. Figure 1.1 [16,17] is a classification flow chart of "Three Dimensional Range finders-Direct Range Data." The range finder developed and used in this project falls within this main classification of range finders. Figure 1.2 [16] shows the other main classification of range finders that get their information through indirect methods.

Indirect range finders do not emit any type of energy (LASER, infrared, ultrasonic, etc.) toward the object. They gather their range data by techniques such as focusing [2], known geometry, Moire fringe patterns, etc. [17]. Indirect methods need to have some prior knowledge of what is being measured or be provided in general range data of a surface or an area. They are limited on their ability to provide specific range data.

Conversely, direct range finders use some form of energy to

PLEASE NOTE

**Page(s) not included with original material
and unavailable from author or university.
Filmed as received.**

2 and 3

University Microfilms International

illuminate the target. They can be subdivided into 2 main groups; 1) Time-of-Flight and 2) Triangulation [13]. Time-of-Flight sensors measure the time it takes to project either a LASER or ultrasound and receive it back. Ultrasonic detectors need a normally oriented surface to provide a reflection. [9] There are also problems associated with focusing the beam to minimize the dispersion and improve directivity. When focusing and distance become important then LASERs are used as the projector. These can be precisely projected over a long distance. Drawbacks of LASER time-of-flight range finders can include slow measurement to process the received image. The LASER beam is also subject to being reflected off of mirror-like surfaces and not returned to the receiver.[16,17]

Triangulation methods have become a fundamental method of determining range [17]. Triangulation can be broken down into Passive methods where two cameras view a scene in stereo and active methods where a single camera views the scene and a LASER is projected into the field of view. These methods have the drawback of missing data points where the LASER can be projected into shadow areas that the camera cannot see or one camera sees something that the other cannot[16]. While this can be corrected by reducing the baseline between the camera and LASER, this increases the uncertainty of the final measurement.

The range finding system for this project is an active system. Using a LASER to project the beam, the LASER spot illuminates a selected object. The projection of the beam is manipulated by a pair of stepper motors. A conventional black and white television camera is used as a sensor to record the location of the LASER spot on a two dimensional image plane. From the motor's positions and feedback from the camera the XY&Z location of the LASER spot is calculated using principles of triangulation. Similar systems used an analog spot detector to quickly locate the LASER spot[6].

The advantage of using a camera to locate the LASER spot is to allow a user to directly view what the computer sees. However, this is offset by additional image processing needed to convert the image into a form that the computer understands. While putting this system together it became evident that an effective method is needed to minimize the number of iterations needed to determine a range. An iteration is defined as converting the image to a binary format that the computer understands and processes the image to see if a spot exists. By locating the spot on the image plane, the equation of a line from the camera to the spot can be determined and the equation of the projection of the LASER beam is obtained from the scanner. Using the equations of these two lines the XY&Z location of the LASER spot can be calculated by triangulation for the XY&Z location of the LASER

spot. As developed, there are two methods that can be carried out to determine range information.

The first is a trivial case where the LASER is manipulated by the operator to place the projection of the beam so that the spot illuminates a particular object of interest. The system then locates and triangulates on the LASER spot to calculate the XY&Z position. This is trivial because there is one and only one location on the image plane representing the LASER spot, thus only one iteration is required to determine the XY&Z location. Depending on the operating environment, this can be a tedious task where the operator is provides the feedback in placing the LASER beam on the object. It requires the user to be in the immediate area of the range finder preventing the system from being remotely operated.

In comparison the nontrivial case allows the user to select an object displayed on the television monitor. In selecting the object an equation of a line is defined. The range finding system then manipulates the LASER beam such that it is projected through various points along this line. Once the LASER spot is detected on the selected object, the XY&Z location can be determined. This method brings up the problem that in theory there are an infinite number of points along this line that the LASER can be projected through. In reality it means a large number of iterations needed to solve for the

XY&Z location. The calculation time is so large in fact that it may not be practical to build such a system.

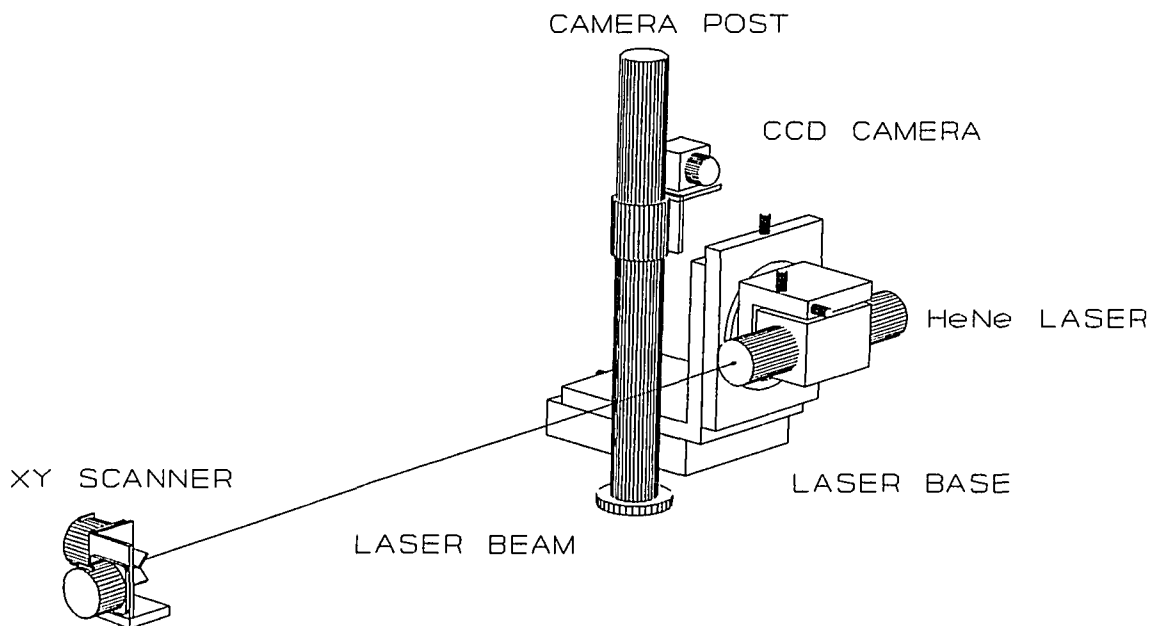
One of the objectives of this thesis is to develop an efficient method to implement the nontrivial case for range detection. A range system as described in this project could be used to locate objects in a robotic work area, remotely pilot a vehicle, or even mathematically construct an object 's structure geometry in the computer. This system was designed to be implemented in a robotic work cell and determine ranges from approximately 10 to 22 feet.

The principles of triangulation are rather straightforward, yet the mathematics involved in a three-dimensional transformation are complicated and can be very confusing. Chapter 3 introduces coordinate transformations and how they are used to describe the location of objects in two or three-dimensional space. This chapter forms the basis for all the math used to describe the placement of the camera in chapter 4 and control of the LASER beam projection in chapter 5. Chapters 4 & 5 also describe how these subsystems are setup and calibrated. Chapter 6 brings the previous two chapters together describing the triangulation for non-intersecting lines. The method of optimization used to minimize the number of iterations is described in chapter 7. Chapter 8 reports the statistical performance of the system.

CHAPTER 2

CONFIGURATION OF SYSTEM

Figure 2.1 shows the setup of the equipment used for the experiment. The equipment can be divided into three categories: 1) The computer (not shown), 2) The X-Y LASER projector, and 3) The camera/image acquisition and processor. This chapter describes the equipment, specifications (where important), physical set-up, implementation, and their interrelation with each other.



**SYSTEM SETUP
FIGURE 2.1**

2.1 Equipment

The purpose of the X-Y LASER projector is to project a beam into 3-D space such that its spot falls within the camera's field of view. The CCD camera is used to view the work area and locate the LASER spot pending it is within its field of view. Finally the computer is used to control the projection of the LASER and process the data received from the camera. This is a brief overview of how these major components interrelate.

2.1.1 Computer

The computer gathers input, controls the equipment, and process the data. It is made up of the hardware, software, and supports the peripheral equipment.

The computer used to support this project was a Gateway 2000, 16 MHz 286 AT. It ran the ranging software, supported the image processing hardware, and communicated with the indexing motors controlling the projection of the LASER beam.

By its nature image processing requires a lot of computing power. Having a more powerful computer (ie 25 MHz 386's or 33 MHz 486's) which processes more information at faster rates could be necessary depending on the application, especially

where minimizing time is essential. For this project it would have been desirable to have a faster computer but not necessary. While better computers were available to support this project, minimizing the time to determine the range was not the main objective. Instead the focus was on developing techniques which minimized the number of iterations needed to determine the range.

2.1.2 Camera/Image Processing

A MVP-AT (Matrox Electronic Systems Ltd.) image processing package is used to perform the image processing routines.

This hardware/software package consisted of a:

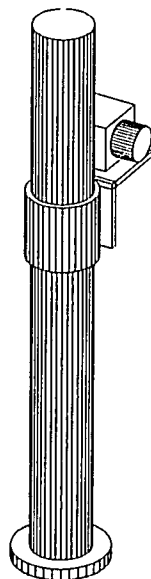
- Image Processing and Acquisition Hardware installed in the computer,
- Image Processing Software Libraries written in C (compiled in the ranging program),
- Charged Coupled Device (CCD) television camera, and
- Video Monitor used in addition to the normal computer monitor.

Image acquisition and processing software takes an image from the camera as input, processes it, and determine some desired information. As mentioned earlier, image processing can be very expensive as far as computing time is concerned. In our case we have an image consisting of 512 by 480 discrete pixels with each pixel capable of representing one of 256 levels of the gray scale. This creates a single image that is approximately 250 kilobytes in size. Image processing routines effectively manipulate the image to provide the

desired information. These image processing routines are executed when called by the program.

The project uses image processing routines to determine if a LASER spot exists and if it does the location of the spot on the image plane. Once the representation of a spot is identified on the camera's image plane, the XY&Z position of that spot can then be determined. The location of the LASER spot can be compared with the selected pixel. Once the location of the spot is coincident with the selected pixel on the image plane the range will be calculated for the desired object.

A CCD black and white TV camera is used to view the work space. Figure 2.2 shows the camera mounted on the pole.



CAMERA/POLE SETUP
FIGURE 2.2

There are two reasons for using a TV camera: 1) The user sees the same work environment that the computer does and 2) As an input device to locate the LASER spot. Using a TV camera requires considerably more computation to find the location of the LASER spot on the image plane as compared to spot finder. First the image must be examined to see whether there is anything that could be considered a LASER spot. Then any locations found must be processed to see if they are indeed the spot. If both of those conditions are met then the coordinates of the LASER spot on the image plane are determined. The disadvantages of the added processing are offset by the advantage of the user viewing the scene through the same camera.

The image processing card supports an additional video monitor displaying the input from the TV camera. The user can manipulate a cursor on this monitor to select objects. The regular monitor is supported by the video card in the computer. It displays alpha numeric output calculated by the ranging program.

2.1.3 LASER Projector

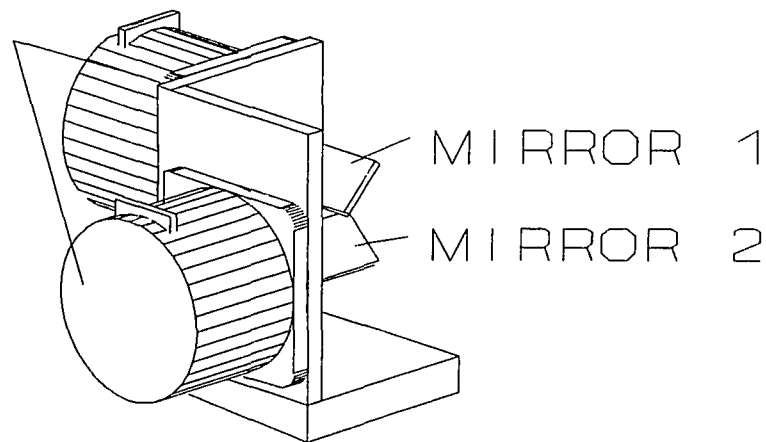
The LASER projector controls the projection of the laser into 3-dimensional space. It is the most physically complicated subsystem and is made up of:

- a 5 mw Helium-Neon LASER,

- two Stepper Motors,
- two Controllers for the Stepper Motors,
- two specially coated mirrors, and
- the mounting equipment.

It can be divided into 1) the projector and 2) the LASER.

STEPPER
MOTORS



STEPPER MOTOR SETUP
FIGURE 2.3

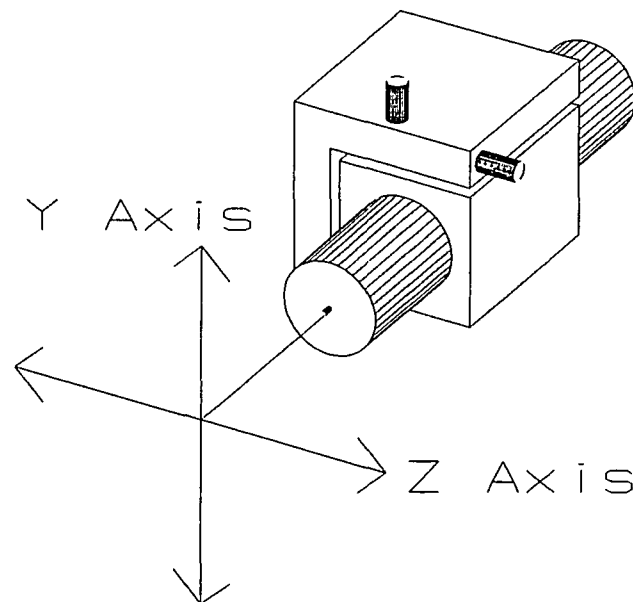
Figure 2.3 shows the setup of the projector. It is used to control the beam and project it into 3-D space. Each stepper motor has 200 discrete positions per 360 degrees of revolution. Between each position it can be divided into 1/125th resolution. This gives a capability of 25,000 steps per revolution. An example of the position resolution is that incrementing an indexer one step would move the LASER spot 1/16th of an inch at a distance of 20 feet. To put this magnitude into perspective, the LASER spot at this distance is approximately 3/16ths of an inch. One motor controls the deflection of the LASER in the X and the other in the Y axis.

The computer does not directly communicate with the stepper motors. Each motor gets its power and commands from their own control box. The control box takes position input commands from the computer through the serial port. It then converts the position information into the proper amount of pulses needed to accurately rotate the motor to the desired position. These control boxes are initialized, by the program, with various parameters controlling the motors such as resolution, acceleration, deceleration, speed, etc.

Attached to each motor spindle is a mirror used to reflect the LASER beam. The first mirror reflects the beam on to the second mirror which in turn projects the beam into 3-D space. Because the reflection is critical specially coated front surface mirrors are used. Both mirrors have the reflective coating on the outside of the glass as compared to a typical mirror whose coating is on the inside of the glass. Having the coating on the outside allows the reflection to take place without passing through the glass thus eliminating an index of refraction error. Since the coating is not protected by the glass, these mirrors need to be handled with care to keep them from being scratched.

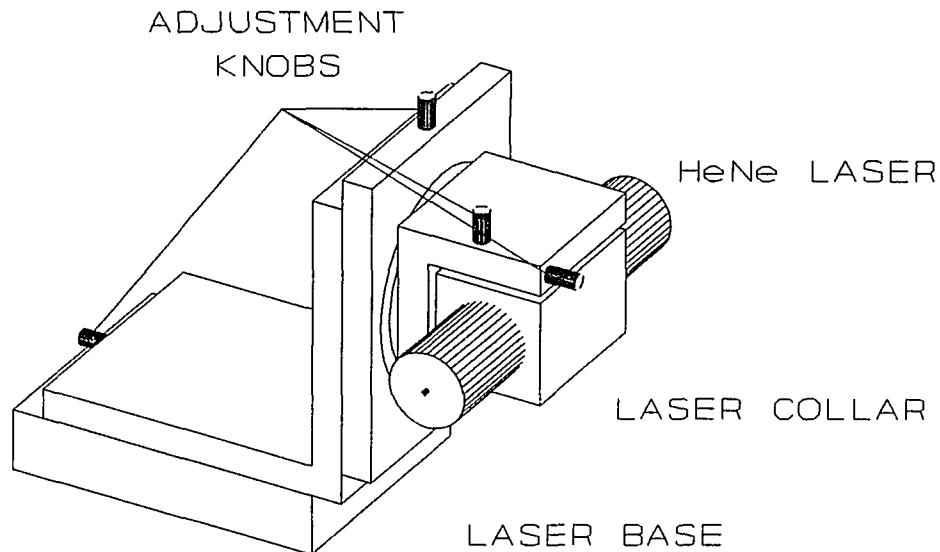
All of this equipment is used to control the projection of the LASER beam. A five milliwatt Helium-Neon LASER was used as the projector. It produces a very bright beam, intense red

beam that can be projected across a long distance, with little dispersion, to illuminate an object with a LASER spot. When viewed by the camera the spot shows up on the image plane as the most intensely lit pixels. This is generally true regardless of the color or texture of the material the spot is on.



LASER/COLLAR SETUP
FIGURE 2.4

The LASER is mounted on a stand that allows very precise adjustments. Figure 2.4 shows the LASER affixed in its collar. The collar allows the LASER to be pivoted about the Y and Z axes. This enables the LASER to be aligned squarely with the projector. In figure 2.5 shows the LASER and collar attached to the base. The base allows the LASER to be



**PROJECTOR SETUP
FIGURE 2.5**

translated along the Y and Z axis. This enables the LASER to be positioned with the projector. Once the LASER is calibrated with it's environment, it doesn't need to be readjusted.

2.2 Limitations and Assumptions

For this project to work as described, there are certain limitations within the framework that this system is implemented. Likewise, in order for this to work as intended there are certain assumptions that were made and adhered to during the experimentation.

With most range finders of this type have been used to

determine ranges of one to several inches with an error of $1/32$ to $1/4$ of an inch. In this project we wanted to determine ranges to objects that are 10 to 22 feet away while maintaining an appropriate error margin.

The system is designed to work in a closed environment where there is a backwall eliminates ranges greater than 22 feet. This is important because it gives the system an established starting point. Its importance will be described later on.

One assumption is that the LASER spot identified by the imaging software is the actual LASER spot as opposed to an erroneous point such as a specular reflection. Conditions were controlled to limit these erroneous points.

CHAPTER 3

MATHEMATICAL BACKGROUND

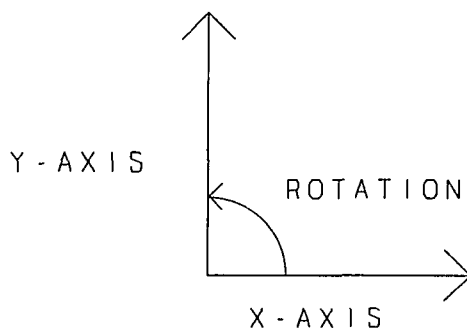
This chapter describes the mathematic fundamentals required to navigate in a 3-D coordinate system. When moving in three dimensions, each motion (rotation or translation) is mathematically described by a transformation matrix. These matrices transform points or coordinate axis to a new orientation or position in the reference coordinate system. While all of this project's work was done in three dimensions, in order to introduce the concepts of transformation matrices we will start by describing them in two dimensions. Examples in a 2-D system are easier to understand and can also be better described on paper than a 3-D system. The concepts for the 3-D systems are the same as the 2-D and are applied in exactly in the same way.

After describing the necessary transformations in 2-D and giving examples of them, the 3-D transformations will be defined. They will be used develop the equations needed to describe the proposed system.

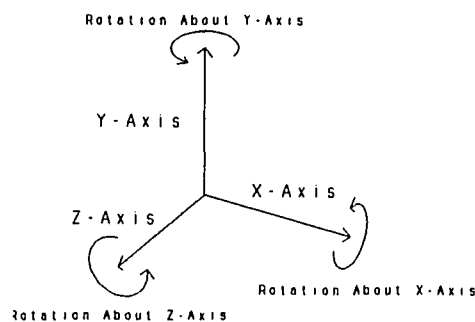
3.1 Coordinate Systems

This project deals with measuring the distances to points and also between points in three dimensions. The coordinate system provides a standardized framework which relatively defines the location, or coordinates, of these points. For the purposes of this project two coordinate systems are used to identify points. One is a fixed World Coordinate System and the other is a transformable Camera Coordinate System representing the camera's image plane.

The World Coordinate System (WCS) is the primary reference coordinate system fixed to ground. Coordinates in this system are denoted by upper case X&Y or XY&Z. The WCS provides a standardized framework in which points or other coordinate axis can be translated or rotated relative to the WCS.



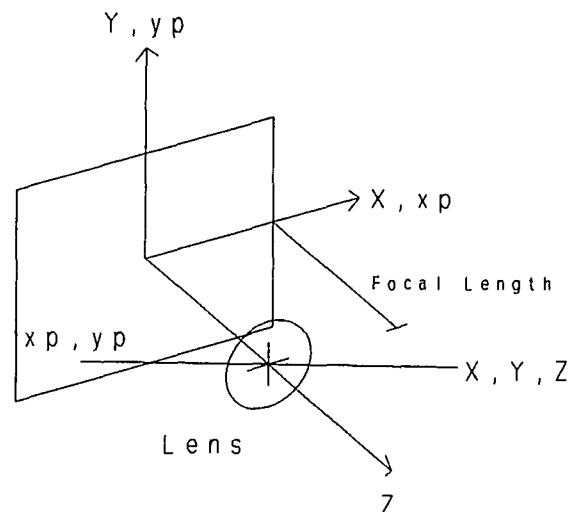
2-D COORDINATE SYSTEM
FIGURE 3.1 a



3-D COORDINATE SYSTEM
FIGURE 3.1 b

The Camera Coordinate System (CCS) is used to represent the position and orientation of the camera's image plane. Coordinates defined relative to the CCS are denoted by lower

case x & y or xy & z . Points on the image plane are denoted by x_p & y_p . Both the WCS and CCS are right hand coordinate systems as shown in figure 3.1 a & b for two and three dimensions. Figure 3.2 shows the definition of the Camera Coordinate System. Points in 3-D are projected through the lens center on to the image plane in the focal plane.



CAMERA COORDINATE SYSTEM
FIGURE 3.2

3.2 Transformation Matrices

There are three types of transformations used in this project
1) Translations, 2) Rotations, and 3) Perspective matrices.
These transformations can be used in one of two methods.
Applying the rules of kinematics these transformations can describe the transformations of points or coordinate systems. They can also to be used to change the relative description of points between two coordinate systems by using inverse kinematics.

Each transformation matrix can be combined to describe the movement of the CCS to a position and an orientation relative to the WCS. The Composite Transformation is the collective name for all of the individual transformation matrices. Composite transformations are governed by the rules of matrix multiplication. [10]

$$AB \neq BA.$$

Since matrix multiplication is not commutative, these matrices must be multiplied in a specific order to properly model the system.

The remainder of this chapter is devoted to giving the reader an understanding of these transformations and how they are used to model the camera's placement. Rather than cite linear algebra rule governing the proper order for combining transformation matrices, it is easier to understand them by seeing examples. Successive examples will quickly show how to model the placement of the CCS in two dimensions similar to the actual camera used for this project. Once these principles for a 2-D system are understood they can be directly applied in the same fashion to the 3-D case.

3.3 2-Dimensional Translation Matrices

This section describes the translation, rotation, and perspective transformations. With each transformation there will be an example describing its characteristics and how it's used. Uppercase (X&Y) denote coordinates relative to the WCS and lowercase coordinates (x&y) are relative to the CCS.

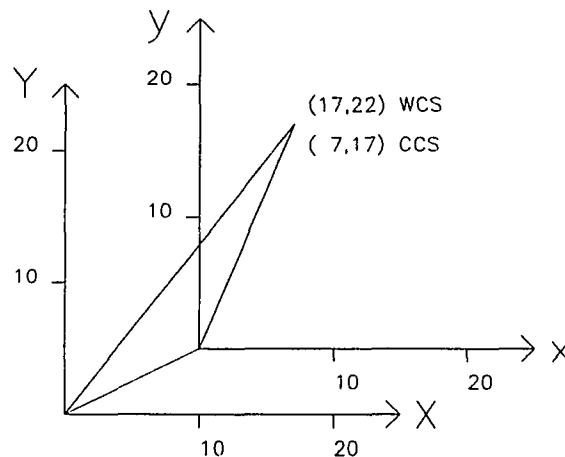
3.3.1 2-D Translations

Translations are used to describe movement along an axis. They can move a coordinate or coordinate system within the WCS or CCS. The basic equation for translation is:[3,4,7]

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} x+\Delta x \\ y+\Delta y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad 3.1$$

EXAMPLE 1:

The CCS is translated +10 along the X axis and +5 along the Y axis. Describe a point (17,22) in the WCS relative to the location of CCS.



TRANSLATION EXAMPLE
FIGURE 3.3

The purpose of these transformations is to change the location of a point or coordinate system within another coordinate system. In this example it is necessary to change the relative description of a point in the WCS to describe it to the CCS. It is easy to see that subtracting the translation of the CCS from the point in WCS will describe it relative to the CCS. Example problem 1 is looking for the redefinition of a point from one coordinate system to another. By taking the inverse transformation

$$\begin{bmatrix} 1 & 0 & -Tx \\ 0 & 1 & -Ty \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & Tx \\ 0 & 1 & Ty \\ 0 & 0 & 1 \end{bmatrix}^{-1}$$

3.2

and substituting it into the translation transformation, the coordinates of points can be described between coordinate systems. This mathematically described by the following equation:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} X - Tx \\ Y - Ty \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & Tx \\ 0 & 1 & Ty \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \quad 3.3$$

SOLUTION 1:

$$\begin{bmatrix} 7 \\ 17 \\ 1 \end{bmatrix} = \begin{bmatrix} 17 - 10 \\ 22 - 5 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 10 \\ 0 & 1 & 5 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 17 \\ 22 \\ 1 \end{bmatrix}$$

ANSWER 1:

$$\begin{aligned} x &= 7 \\ y &= 17 \end{aligned}$$

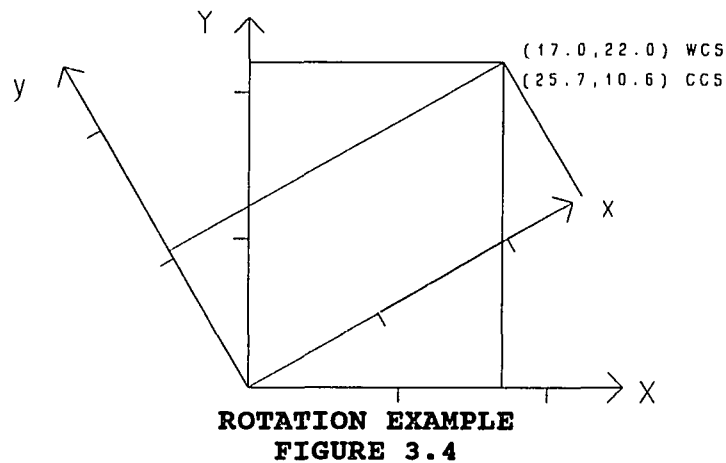
3.3.2 2-D Rotations

Rotations are used to rotate a point or a coordinate system about an axis perpendicular to XY plane. The equation for rotations are:

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} x \cos(\theta) - y \sin(\theta) \\ x \sin(\theta) + y \cos(\theta) \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad 3.4$$

EXAMPLE 2:

The CCS is rotated by +30 degrees about the origin of the WCS. Describe a point (17,22) relative to the WCS in terms of the CCS.

**SOLUTION:**

The inverse of the rotation transformation matrix is

$$\begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \quad 3.5$$

When substituting it into the rotation transformation the equation becomes

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} X \cdot \cos(\theta) + Y \cdot \sin(\theta) \\ X \cdot \sin(\theta) - Y \cdot \cos(\theta) \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \quad 3.6$$

When values are substituted into the equation

ANSWER 2:

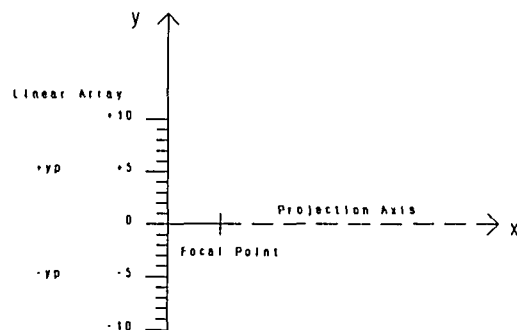
$$\begin{bmatrix} 17*\cos(30)+22*\sin(30) \\ -17*\sin(30)+22*\cos(30) \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(30) & \sin(30) & 0 \\ -\sin(30) & \cos(30) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 17 \\ 22 \\ 1 \end{bmatrix}$$

$$x = 25.7$$

$$y = 10.6$$

3.3.3 2-D Projections

Projection transformations are more difficult to understand than the translation or rotation transformations. There are two methods projections can be implemented 1) Direct or 2) Inverse kinematics. The direct method takes a point relative to the CCS, projects it through the focal point, and on to the linear array. In inverse kinematics a location on the linear array is selected. From this point a line can be projected through the focal point into 2-D space. More information is required to determine a discrete point along this line.



2-D LINEAR ARRAY
FIGURE 3.5

For this section the transformation defines the x axis as the

projection axis. The linear array is +/- 10 and the focal length is 5 units.

The equation for the projection transformation as described is:

$$\begin{bmatrix} x' \\ y' \\ k \end{bmatrix} = \begin{bmatrix} X \\ Y \\ -\frac{X}{\lambda} + 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\frac{1}{\lambda} & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \quad 3.7$$

The initial result from applying the projection transformation is in a nonstandard format since the third term in the coordinate array is k instead of 1. This can be converted to standard format by:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \frac{1}{k} \begin{bmatrix} x' \\ y' \\ k \end{bmatrix} \quad 3.8$$

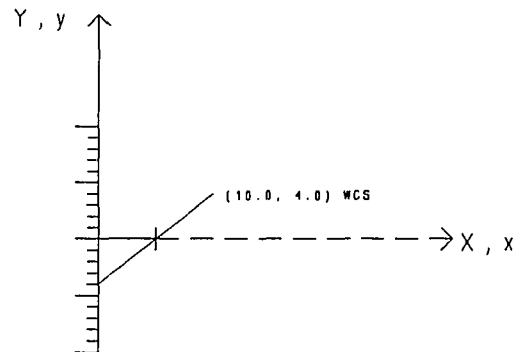
This gives a final result in the standard format where y' is the location yp on the linear array. The x' has no meaning here and is not needed.

The upcoming examples will show step by step how these transformations are used in more complicated scenarios. There are three things these examples will emphasize

- 1) The proper order translation and rotation matrices needed to transform a point in the WCS to the CCS,
- 2) The location of the projection on the linear array, and
- 3) How projection transformations are used.

EXAMPLE 1

The CCS is coincident with the WCS. A point (10.0, 4.0) in the WCS is projected onto the linear array. What is the location y_p on the linear array?



PROJECTION EXAMPLE 1
FIGURE 3.6

$$\begin{bmatrix} 10 \\ 4 \\ -1 \end{bmatrix} = \begin{bmatrix} 10 \\ 4 \\ -\frac{10}{5} + 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\frac{1}{5} & 0 & 1 \end{bmatrix} \begin{bmatrix} 10 \\ 4 \\ 1 \end{bmatrix}$$

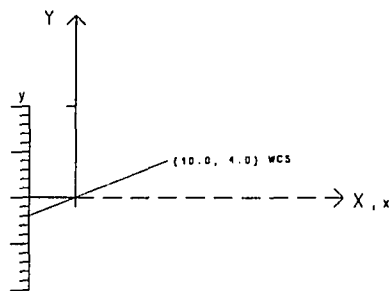
Converting the above result into standard form is done by:

$$\begin{bmatrix} -10 \\ -4 \\ 1 \end{bmatrix} = \frac{1}{-1} \begin{bmatrix} 10 \\ 4 \\ -1 \end{bmatrix}$$

From the final result the location of the projection on the linear array is at -4.0. The figure for this example makes the result look trivial. However the formulas needed to describe this system will be further complicated in the upcoming examples.

EXAMPLE 2

The CCS is translated by the focal length in the negative direction along the projection axis. A point (10.0, 4.0) in the WCS is projected onto the linear array. What is the location y_p on the linear array?



PROJECTION EXAMPLE 2
FIGURE 3.7

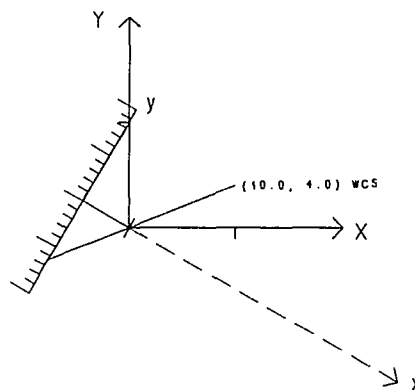
$$\begin{bmatrix} 15 \\ 4 \\ -2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\frac{1}{5} & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -5 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 10 \\ 4 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} -7.5 \\ -2 \\ 1 \end{bmatrix} = \frac{1}{-2} \begin{bmatrix} 15 \\ 4 \\ -2 \end{bmatrix}$$

In this example the translation puts the focal point at the origin of the WCS. This is done to allow all adjustments of rotations to occur about the focal point. In modeling the system it was preferable to having these adjustments occurring at the focal point as opposed to the center of the linear array. This will be shown in the next example.

EXAMPLE 3

The CCS is 1) translated by the focal length in the negative direction along the projection axis and then 2) rotated by -30 degrees. A point $(10.0, 4.0)$ in the WCS is projected onto the linear array. What is the location yp on the linear array?



PROJECTION EXAMPLE 3
FIGURE 3.8

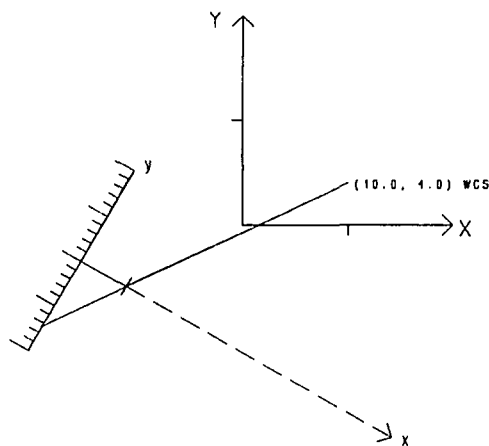
$$\begin{bmatrix} 11.66 \\ 6.46 \\ -1.33 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\frac{1}{5} & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -5 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} \cos(-30) & -\sin(-30) & 0 \\ \sin(-30) & \cos(-30) & 0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 10 \\ 4 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} -8.75 \\ -6.35 \\ 1 \end{bmatrix} = \frac{1}{-1.33} \begin{bmatrix} 11.66 \\ 8.46 \\ -1.33 \end{bmatrix}$$

This example begins to show these transformations are ordered. The location of the projection on the linear array shown in the figure verifies the method of implementation. In this example first the CCS is translated then it's rotated. The purpose of the inverse of these transformations are to convert the point $(10,4)$ from the WCS to the CCS.

EXAMPLE 4

The CCS is 1) translated by the focal length in the negative direction along the projection axis, 2) rotated by -30 degrees, and then 3) translated by -11 in the x and -6 in the y directions. A point (10.0, 4.0) in the WCS is projected onto the linear array. What is the location y_p on the linear array?



PROJECTION EXAMPLE 4
FIGURE 3.9

$$\begin{bmatrix} 18.19 \\ 19.16 \\ -2.64 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\frac{1}{5} & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -5 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} \cos(-30) & -\sin(-30) & 0 \\ \sin(-30) & \cos(-30) & 0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & 0 & -11 \\ 0 & 1 & -6 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 10 \\ 4 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} -6.90 \\ -7.27 \\ 1 \end{bmatrix} = \frac{1}{-2.64} \begin{bmatrix} 18.19 \\ 19.16 \\ -2.64 \end{bmatrix}$$

This example just reinforces the order demonstrated with a

slightly more complex system. The principles used for these examples in 2-D can be directly applied to the three dimensional case. The only differences are that the transformations are 4X4 matrices and the locations of the projections are onto an image plane.

3.4 3-D TRANSFORMATIONS

In this section the matrices for three dimensional transformations will be defined. Because they are so similar to the 2-D transformations refer to the examples in the previous section if there are any questions about how to use them.

These 3-D transformations will be used to describe the camera equations and indexer position.

3.4.1 TRANSLATION

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} x+Tx \\ y+Ty \\ z+Tz \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & Tx \\ 0 & 1 & 0 & Ty \\ 0 & 0 & 1 & Tz \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad 3.9$$

3.4.2 ROTATION ABOUT X-AXIS

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y*\cos(\alpha) - z*\sin(\alpha) \\ y*\sin(\alpha) + z*\cos(\alpha) \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) & 0 \\ 0 & \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad \mathbf{3.10}$$

3.4.3 ROTATION ABOUT Y-AXIS

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} x*\cos(\theta) + z*\sin(\theta) \\ y \\ -x*\sin(\theta) + z*\cos(\theta) \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad \mathbf{3.11}$$

3.4.4 ROTATION ABOUT Z-AXIS

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} x*\cos(\beta) + y*\sin(\beta) \\ x*\sin(\beta) - y*\cos(\beta) \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\beta) & -\sin(\beta) & 0 & 0 \\ \sin(\beta) & \cos(\beta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad \mathbf{3.12}$$

3.4.5 PERSPECTIVE

The Z axis is used as the projection axis in this project.

$$\begin{bmatrix} X \\ Y \\ Z \\ -\frac{Z}{\lambda}+1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{1}{\lambda} & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad 3.13$$

For a point $[X \ Y \ Z \ 1]^T$ shown in figure 3.10.

$$\begin{bmatrix} X \\ Y \\ Z \\ -\frac{Z}{\lambda}+1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{X\lambda}{\lambda-Z} \\ \frac{Y\lambda}{\lambda-Z} \\ \frac{Z\lambda}{\lambda-Z} \\ 1 \end{bmatrix} \quad 3.14$$

Where the first two components (x,y) are the (x_p,y_p) coordinates on the image plane of a projected 3D point (X,Y,Z) and the third component (z) is of no interest in our model.

These transformations are the building blocks used to mathematically model the camera placement and determine the projection of the laser for a given motor's position. Three dimensional transformations will be used quite extensively in the upcoming chapters.

CHAPTER 4

CAMERA CALIBRATION

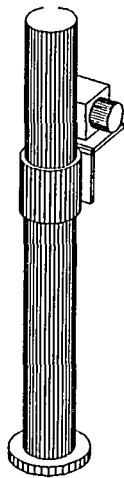
The camera is used as an input sensor to view the work area and locate the position of the laser spot. This chapter uses transformations explained in the previous chapter to describe camera placement. The first section details the physical location of the camera. It shows how the transformations are used in the proper order to describe the XY&Z locations of 1) the stationary focal point and 2) the variably defined points on the camera's image plane. In section 4.2 these transformations will be used to derive equations that are implemented in the computer program.

It is very important to have the most accurate numeric values for the variables of translation, rotation, and perspective transformations to minimize system errors describing camera location. Section 4.3 describes in great detail the theory and method used to obtain the most accurate values for rotations, translations, and perspective parameters used in the ranging software.

4.1 DESCRIPTION OF CAMERA SET-UP

This chapter describes the XY&Z location of two points in the

World Coordinate System. The first is the stationary location of the camera's lens center or focal point. It is through this point that all projections on to the image plane pass based on the assumption of geometric optics. The second is the variable point existing anywhere on the image plane. This point can either be a location designated by the operator via the monitor or the location of the laser spot calculated by the image processing routines. These two points are used to define the equation of a line that is projected out from the camera, into the work space, and onto the selected object. It is necessary to have these points defined relative to the WCS.



CAMERA/POLE SETUP
FIGURE 4.1

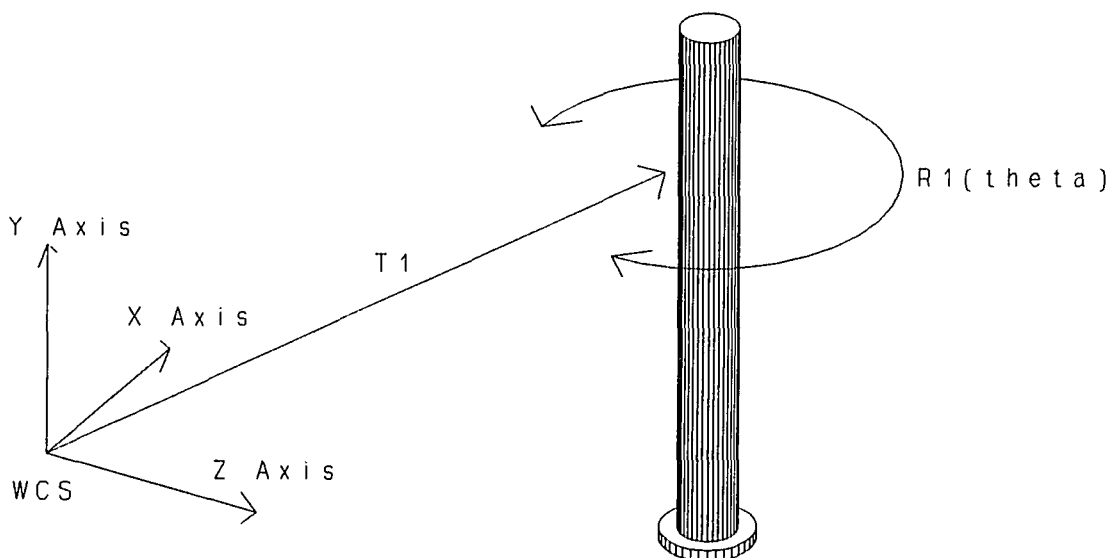
Figure 4.1 shows the camera mounted to a collar that can be positioned on the pole. In table 4.1 the order of the transformations used to define the placement of the camera's image plane and the associated variable names.

Definition of Transformation Matrix Notation and Associated Variables		
T1	Translation along X, Y, & Z	X1, Y1, & Z1
R1	Rotation about Y	Theta
T2	Translating along X & Z	X2 & Z2
R2	Rotation about X	Alpha
R3	Rotation about Z	Beta
T3	Translation along Z	Lambda
P1	Perspective along Z axis	Lambda
T4	Translation on Image Plane	x _p & y _p

TABLE 4.1

T1 (Translation #1) - This is the first translation where X & Z are the coordinates of the location of the pole center mounted on the table. The value of Y represents the height of the lens center above the WCS. (figure 4.2)

R1 (Rotation #1) - This is the first rotation where the collar, that the camera is mounted to, can rotate about the pole. (figure 4.2)

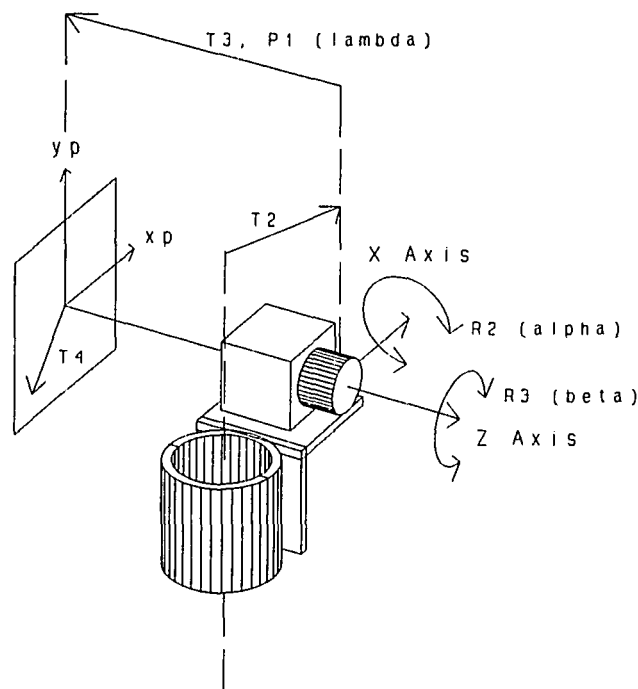


TRANSLATION/ROTATION I
FIGURE 4.2

T2 (Translation #2) - This is the second translation going from the pole center to the camera center in the XZ plane. There is no Y component for this translation, it is taken care of in T1. (figure 4.3)

R2 & R3 (Rotations #2 & #3) - These two rotations take into account for minimal rotations that exist at the lens center but are too small measure. (figure 4.3)

T3 (Translation #3) - This is the final translation that translates the image plane by the focal length. The effect of this translation allows the lens center to be stationary point. This point is one of two that is used to define the equation of a line. The other point is a variable one defined on the image plane. (figure 4.3)



TRANSLATION/ROTATION II
FIGURE 4.3

P1 (Perspective #1) - This is the perspective transformation used to project points in the WCS on to the image plane.

T4 (Translation #4) - This translation is used to move to the selected point on the image plane from the center. This point defines the the second point defining a line.

4.2 EQUATIONS FOR CAMERA PLACEMENT

In describing camera placement, equations need to be developed to describe the location of 1) the lens center and 2) points on the image plane in WCS. These two points are needed to define the equation of a line in WCS which will be used later. The composite transformations are used to describe the location of these two points. The manner in which they are combined defines the two points which in turn defines the line that is projected out into the WCS.

By assuming the lens in WCS center coincides with the camera center, the location of the lens center is defined by:

$$\begin{bmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{bmatrix} = T1 \cdot R1 \cdot T2 \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad 4.1$$

where

$$X_C = X2 \cdot \cos(\theta) + Z2 \cdot \sin(\theta) + X1$$

$$Y_C = Y1$$

$$Z_C = -X2 \cdot \sin(\theta) + Z2 \cdot \cos(\theta) + Z1$$

and $[0 \ 0 \ 0 \ 1]^T$ is the origin of the WCS.

Since assuming that small rotations R2 and R3 exist about the lens center $[X_C \ Y_C \ Z_C \ 1]$, the final transformation matrix which

defines the Camera Coordinate System (CCS) shown in figure 3.2 becomes

$$T_{CCS} = T1 \cdot R1 \cdot T2 \cdot R2 \cdot R3 \cdot T3 \quad 4.2$$

where

$$T3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -\lambda \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

T4 is used to convert the pixel location of a point (xp,yp) selected by a user from pixels to inches. By post multiplying T4 in equation 4.2 the actual (X,Y,Z) position in the WCS can be determined for any (xp,yp).

The next logical step is to find the location of the image position in the CCS defined in equation 4.2 for a given (X,Y,Z) location in the WCS. Using the perspective transformation in section 3.4.5, first we have to determine the position of the (X,Y,Z) location relative to the CCS.

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}_{CCS} = T_{CCS}^{-1} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}_{WCS} \quad 4.3$$

Equation 4.3 takes (X,Y,Z) in WCS and defines it relative to the CCS (x,y,z).

Using equation 3.13, the image position in the CCS becomes

$$\begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix}_{CCS} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{1}{\lambda} & 1 \end{bmatrix} T_{CCS}^{-1} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}_{WCS} \quad 4.4$$

Equation 4.4 builds upon 4.3 taking the point relative to the CCS and projecting it onto the image plane.

To convert the image position (currently in CCS) to the WCS, the location is multiplied by T_{CCS} .

$$\begin{bmatrix} X_p \\ Y_p \\ Z_p \\ 1 \end{bmatrix}_{WCS} = T_{CCS} \begin{bmatrix} x_p \\ y_p \\ 0 \\ 1 \end{bmatrix}_{CCS} \quad 4.5$$

Equation 4.1 and 4.5 define the two points in the WCS which are needed to define the image axis in the WCS.

It should be noted that the perspective transformations in equation 4.4 is not necessary when the image axis was determined using the image processing board in the computer, i.e. (x_p, y_p) can be directly obtained from the image processing routines and use this for equation 4.5 to calculate the image position in the WCS. However, we need to use equation 4.4 for camera calibration procedures which will be discussed next.

4.3 CAMERA CALIBRATION

Most camera calibration/placement methods are heavily based on linear algebra variable reduction.[4] The equations of camera placement are developed to relate a point in $XY&Z$ with $xp&yp$. A data set of $XY&Z$ and its corresponding $xp&yp$ is collected as an input. It has a number of points equal to the number of unknowns. The equations are manipulated by variable reduction or matrix inversion to solve for the translation, rotation, and/or perspective transformation parameters.

There are two drawbacks in calibrating a system using this technique. The first is that it isn't always possible to isolate variables for complicated systems. The second, due to the small number of data points there is an implied assumption that the data set is collected without error. The method developed for this project accounts for existing measurement errors and minimizes their effect on the system.

4.3.1 METHOD OF MINIMIZATION

The method devised to solve for camera placement is quite simple. Two data sets of points are collected 1) $X, Y, & Z$ (XYZ) points in world coordinates and 2) the corresponding (xp_m, yp_m) points measured on the image plane. From the transformation matrices, the points of (X, Y, Z) can be mathematically projected onto the image plane. These points

are denoted as calculated points on the image plane (x_{pc}, y_{pc}) . The objective is to adjust the translation, rotation, and perspective parameters so as to minimize the errors (distances) between (x_{pm}, y_{pm}) and (x_{pc}, y_{pc}) .

4.3.2 CAMERA SYSTEM AND EQUATIONS

Table 4.1 shows the order of the transformation matrices needed to define the CCS. All of the transformation matrices are based on the conventional right hand coordinate systems. Using equations 4.2 and 4.4 the image position (x_{pc}, y_{pc}) can be defined in the CCS for a given location (X, Y, Z) in the WCS.

$$\begin{bmatrix} kx_{pc} \\ ky_{pc} \\ kz \\ k \end{bmatrix} = T3^{-1} R3^{-1} R2^{-1} T2^{-1} R1^{-1} T1^{-1} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad 4.4$$

Combining the camera transformation matrices into one collective transformation matrix T_{CCS}^{-1} , the simplified equation becomes:

$$\begin{bmatrix} kx_{pc} \\ ky_{pc} \\ kz \\ k \end{bmatrix} = T_{CCS}^{-1} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad 4.4$$

where in CCS

$$T_{CCS}^{-1} = \begin{bmatrix} A & B & C & D \\ E & F & G & H \\ I & J & K & L \\ M & N & O & P \end{bmatrix}$$

in WCS and A-P for T_{CCS}^{-1} is

$$A = \cos(\beta) \cdot \cos(\theta) + \sin(\beta) \cdot \sin(\alpha) \cdot \sin(\theta)$$

$$B = \sin(\beta) \cdot \cos(\alpha)$$

$$C = -\cos(\beta) \cdot \sin(\theta) + \sin(\beta) \cdot \sin(\alpha) \cdot \cos(\theta)$$

$$D = \cos(\beta) \cdot [\sin(\theta) \cdot Z1 - \cos(\theta) \cdot X1 - X2] - \dots \\ \sin(\beta) \cdot \cos(\alpha) \cdot Y1 - \dots \\ \sin(\beta) \cdot \sin(\alpha) \cdot [\sin(\theta) \cdot X1 + \cos(\theta) \cdot Z1 + Z2]$$

$$E = -\sin(\beta) \cdot \cos(\theta) + \cos(\beta) \cdot \sin(\alpha) \cdot \sin(\theta)$$

$$F = \cos(\beta) \cdot \cos(\alpha)$$

$$G = \sin(\beta) \cdot \sin(\theta) + \sin(\beta) \cdot \cos(\alpha) \cdot \cos(\theta)$$

$$H = \sin(\beta) \cdot [\cos(\theta) \cdot X1 - \sin(\theta) \cdot Z1 - X2] - \dots \\ \cos(\beta) \cdot \cos(\alpha) \cdot Y1 - \dots \\ \cos(\beta) \cdot \sin(\alpha) \cdot [\sin(\theta) \cdot X1 + \cos(\theta) \cdot Z1 + Z2]$$

$$M = \frac{-\cos(\alpha) \sin(\theta)}{\lambda}$$

$$N = \sin(\alpha)$$

$$O = \frac{-\cos(\alpha) \cdot \cos(\theta)}{\lambda}$$

$$P = \frac{-\sin(\alpha) \cdot Y1 + \cos(\alpha) \cdot [\sin(\theta) \cdot X1 + \cos(\theta) \cdot Z1 + Z2]}{\lambda}$$

The values for I, J, K, L are not needed because kz is a meaningless term.

Multiplying T_{CCS}^{-1} and (X, Y, Z) in WCS becomes

$$\begin{bmatrix} k \cdot xp_c \\ k \cdot yp_c \\ k \end{bmatrix} = \begin{bmatrix} A \cdot X + B \cdot Y + C \cdot Z + D \\ E \cdot X + F \cdot Y + G \cdot Z + H \\ M \cdot X + N \cdot Y + O \cdot Z + P \end{bmatrix} \quad 4.7$$

In order to get the xp_c & yp_c , c_h must be multiplied by $1 / k$.

The values for xp_c & yp_c are:

$$xp_c = \frac{A \cdot X + B \cdot Y + C \cdot Z + D}{M \cdot X + N \cdot Y + O \cdot Z + P} \quad 4.8$$

$$yp_c = \frac{E \cdot X + F \cdot Y + G \cdot Z + H}{M \cdot X + N \cdot Y + O \cdot Z + P} \quad 4.9$$

Manipulating the variables for camera placement adjusts the locations of (xp_c, yp_c) on the image plane. This is a very complicated task to do. It is necessary to use the IMSL routines to manipulated these variables. The objective is to get the points of (xp_c, yp_c) to coincide as closely as possible to (xp_m, yp_m) .

IMSL is a problem-solving software system. It is a collection of Fortran subroutines and functions that can be called from a user written program.[18] For this application IMSL routines were used to minimize the error between the (x_{p_m}, y_{p_m}) and (x_{p_c}, y_{p_c}) . The routine used in IMSL is BCLSF which can be used to solve for multiple constrained variables. The BCLSF IMSL routine "...uses a modified Levenberg-Marquardt method to solve nonlinear least square problems subjected to simple bounds". After running the program the routines would return optimal values for the variables. A copy of the program and results is included in appendicies A and B.

The objective function's error is the distance between (x_{p_c}, y_{p_c}) and (x_{p_m}, y_{p_m}) . This is defined as:

$$U = \sqrt{[(x_{p_c} - x_{p_m})^2 + (y_{p_c} - y_{p_m})^2]} \quad 4.10$$

This represents the error for a single point. The average error is the summation of the single point errors divided by the number of points.

4.4 DATA SETS

The first data set collected was the physical measurements of the camera location along with tolerances. This was used as the initial guess and also to determine the validity of the IMSL result. All translations are in inches and all rotations are in radians shown in table 4.2.

Initial Measurements and Tolerances (inches and radians)		
Variable	Measurement	Tolerance
X1	41.6250	+/- 0.125
Y1	17.6250	+/- 0.250
Z1	-4.8125	+/- 0.125
X2	3.7500	+/- 0.125
Z2	2.0000	+/- 2.000
Lambda / Z3	54.2200	+/- 2.000
Alpha	0.0000	+/- 0.100
Theta	0.0000	+/- 0.100
Beta	0.0000	+/- 0.100

TABLE 4.2

Lambda and Z3 are two distinct variables given the same value. The camera does not actually have a focal length of 54.22 inches. The reason for doing this is to keep from working with extremely small values in the program. Likewise the image plane was also appropriately scaled up by the same factor thus also affecting the pixel size.

The other two data sets collected are values for XYZ and the corresponding pixel locations (x_{p_m}, y_{p_m}) . These data sets consisted of 60 points in a 5 by 12 grid which uniformly covers the image plane. The data was collected by a movable calibrated target. The target can be moved to various locations on the floor defining several points in XYZ with an accuracy of about $\pm 1/8$ to $1/4$ of an inch. The data were taken at a distance of 240 inches in the Z axis -24 to +36 inches in the Y axis and 0 to 108 inches in the X axis in 12 inch increments. The placement error of the target was subjectively determined as $\pm 1/4$ ". At a distance of 20' this error corresponded to being equal to \pm one pixel. Likewise, if the target were placed at 120 inches, the placement error would be the same, but would show up on the monitor as \pm two pixels. By taking the data at this distance in the Z axis helps minimize the magnitude of the placement error of the target. While these errors that are not associated with camera placement they are system errors that the camera will be subjected to.

The initial data set has XYZ in inches and (x_{p_m}, y_{p_m}) in pixel location. The following equations convert pixel location into inches.

$$x_{p_m} = 1.254 \cdot 0.0449 \cdot (x_{p_m} - 255) \quad 4.11$$

$$y_{p_m} = 0.0449 \cdot (y_{p_m} - 238) \quad 4.12$$

The height of the pixel is 0.0449 inches and 1.254 is the aspect ratio of width : height. The values of 255 and 238 translate the origin to the center of the image plane. Tables of the data sets are located in appendix A.

4.5 RESULTS

For this project two test were performed. The first was with equal weightings between the 60 data points. In the second the relative weight of the pixels decreased as they moved further from the center of the image plane. This was explored to see what differences, if any, occurred in the variables and the effect on the analysis.

Results of Measured and Optimized Values			
Variable	Measured Values	Test 1 Equal Weight	Test 2 Weighted
X1	41.6250	41.6346	41.6245
Y1	17.6250	17.5141	17.5886
Z1	-4.8125	-4.8125	-4.8125
X2	3.7500	3.7665	3.7565
Z2	2.0000	1.7354	1.7943
Lambda / Z3	54.2200	54.6164	54.6439
Alpha	0.0000	0.0222	0.0225
Theta	0.0000	0.0323	0.0323
Beta	0.0000	-0.0057	-0.0061

TABLE 4.3

The converted data set for (x_{p_m}, y_{p_m}) in appendix A was used for input in both cases.

Using the values for the variables of the measured and both test cases and the XYZ data sets were substituted into the equations to generate three data sets of (xp_c, yp_c) . These data sets are needed with (xp_m, yp_m) for the next section in analyzing the results.

4.6 ANALYSIS OF RESULTS

The basis of the analysis of results is the comparison between the data sets of (xp_m, yp_m) and one of the (xp_c, yp_c) . The differences between (xp_m, yp_m) and (xp_c, yp_c) are initially in inches. Due to the extremely small physical dimensions of a pixel , 0.0449" (H) X 0.0563" (W), it is easy to under estimate the magnitude of the error. Because of this the dimensions divided out to express the differences in terms of pixels.

$$xp_E = \frac{xp_c - xp_m}{0.0563''}$$

$$yp_E = \frac{yp_c - yp_m}{0.0449''}$$

$$magp_E = \sqrt{xp_E^2 + yp_E^2}$$

From this point, the study of error is based on pixels (xp_E, yp_E) . This in no way biases the analysis, it only serves to place the errors in another form that is more tangible to

the reader.

Looking for biases in x_{p_E} and y_{p_E} the errors are respectively sorted and the median value is taken. Likewise the mag_{p_e} is sorted in the same fashion to find median magnitude error of the camera placement. The median value for mag_{ep} is called the Circle Error Probable (CEP). It is the value for which 50% of the points fall within unless otherwise noted.

There are inherent error associated with the collection of the data sets (X,Y,Z) & (x_{p_m},y_{p_m}) . These errors are detrimental to accurate camera placement, but they do exist and are overall system errors that the camera model is expected to handle. Examining the median values for biases and total error is more representative values associated with camera placement rather than data collection.

Analysis of Measured Versus Optimized			
Median Bias (PIXELS)	Measured Values	Test 1 Equal Weight	Test 2 Weighted
X	-31.972	0.062	-0.195
Y	27.461	0.056	0.007
CEP	42.263	0.656	0.740
CEP 90%	43.621	1.366	1.534

TABLE 4.4

It is clear from the analysis that even the most accurate physical measurement of the camera system is unacceptable. It has heavy biases in both the x & y of -31.972 and 27.461

pixels respectively. The placement error of 42.263 pixels represents an uncertainty of ten inches at a distance of 20 feet from the camera. With these errors the LASER range finding system will not work at all.

The analysis for both of the test cases shows a dramatic improvement with a placement error of 0.656 of one pixel in the unweighted and .740 for the weighted cases. The biases in both cases are negligible. The uncertainty for the test cases is less than 0.25 inches at twenty feet.

The majority of error can be attributed to the rotations. The rotations are so small that they cannot be physically measured yet their effect is pronounced. The error attributed to the translations helps to increase the accuracy by approximately 1/2 of a pixel. The translation values are extremely close to the measured ones. These differences do not have a true physical significance because it is almost impossible to measure the distances with the accuracy calculated.

CHAPTER 5

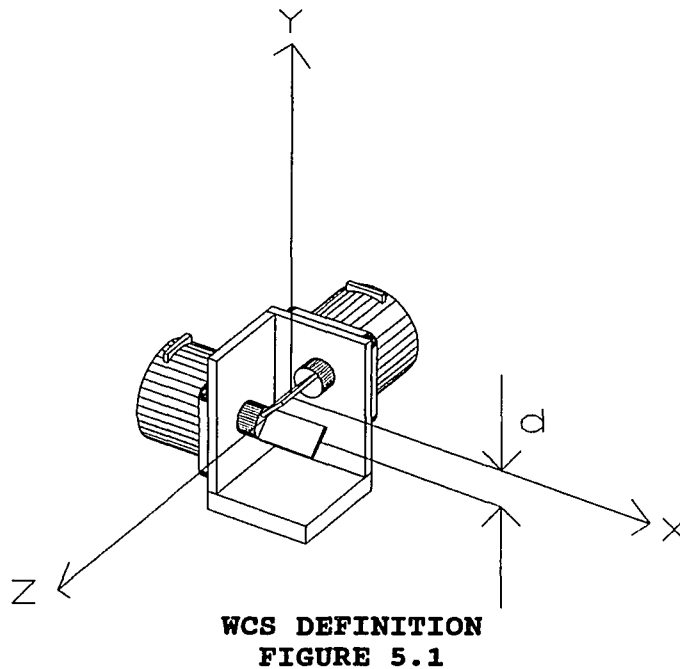
X-Y LASER PROJECTOR

One of the main premises of this project is accurately controlling the projection of the laser beam. One method could be to mount the laser emitter so that it could be pointed in the desired direction. The mass of the assembly made this configuration impractical. Instead it is easier to deal with manipulating the massless beam. The direction of the beam can be redirected by reflecting it off of mirrors. The stepper motors are used to quickly and accurately position the mirrors that inturn affects the direction of the laser beam into three-dimensional space. Like with the camera where accurate modeling was of utmost importance, the same attention to accuracy is also critical here.

The first section of this chapter describes the physical set up of the projector assembly. The next section develops the equations that model the projector. These equations are used to further develop more equations that relate motor position and coordinate location in XY&Z. The last section describes inherent errors associated with the motors and how they were calibrated. This section also explains how the whole projection unit was calibrated with its environment.

5.1 Description of Stepper Motor Set-Up

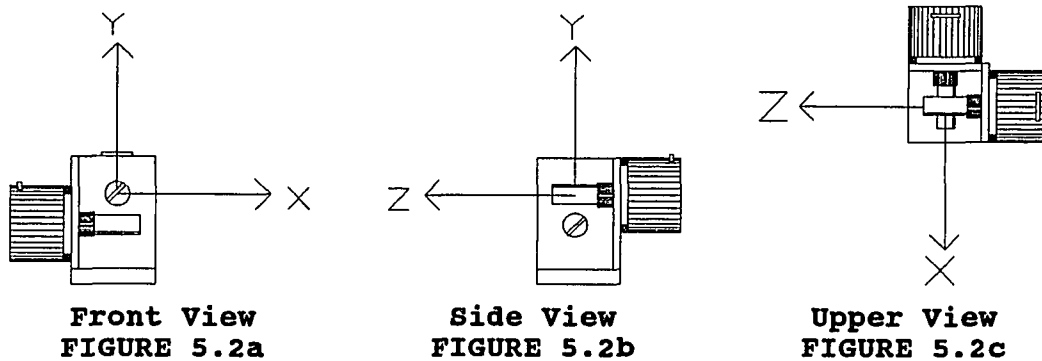
As mentioned earlier the stepper motors are used to position mirrors which reflects the laser beam. Figure 5.1 shows how the two motors are mounted orthogonally to each other.



The laser emitter is mounted so that its beam is exactly projected, in the negative direction, along the X axis toward the first mirror. The first motor/mirror unit, the upper one, is mounted so that its axis of rotation is coincident with the Z axis. The beam is reflected off the first mirror so that it is projected onto the second. The second motor/mirror unit mounted directly underneath the first by a distance of 'd' rotates about an axis parallel to the X axis. From there the

beam is reflected off of mirror 2 and projected into three-dimensional space.

The stationary World Coordinate System XYZ is positioned so that its origin is at the center of mirror 1. Figures 5.2 a through c show a front, side, and an upper elevation views of the projector assembly and WCS. This helps show the exact location of the WCS that might not be perceptible from figure 5.1.



The upcoming section will develop the mathematical equations that model the projectors. These equations are then used to derive the equations needed to project the beam through a point in XYZ as a function of motor position.

5.2 EQUATION OF STEPPER MOTORS

The ultimate objective of this section is to determine the

proper motor positions needed to project the laser beam through a given XYZ in the 3-D workspace. To develop these equations the transformation matrices described in chapter 3 and used in chapter 4 will be used again here.

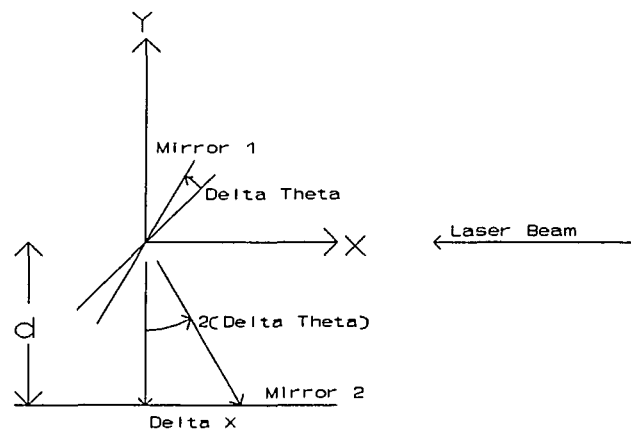
This section will frequently refer to the 'equation of a line' or the 'equation of the laser beam or projection' which is meant to describe the line or laser in terms of its vector components $(i\ j\ k)^T$ or $(l\ m\ n)^T$. Another point of clarification is the terms 'projection' and 'reflection'. The beam is always presumed to moving outward, projected, from the emitter or mirrors. Reflection is meant to describe the immediate change in the equation of the projection after being reflected off of a mirror. Once the equation of the reflection has been defined, it will then be referred to as the next projection.

The equation of the final projection off of the second mirror is defined as a symmetric equation of a line:

$$\frac{(X - X_1)}{l} = \frac{(Y - Y_1)}{m} = \frac{(Z - Z_1)}{n} \quad 5.1$$

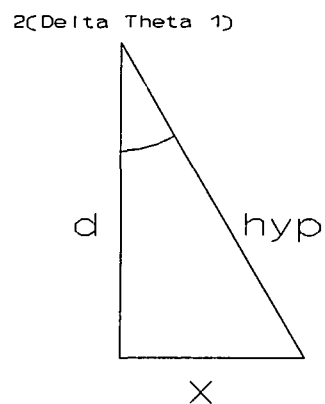
The direction vector $(lmn)^T$ and point $(X_1, Y_1, Z_1)^T$ define the projection as a unique vector. $(X_1, Y_1, Z_1)^T$ is the location where the reflection takes place on the second mirror and $(XYZ)^T$ is any point in the workspace that the laser is projected through. All of these are defined in the WCS.

Figure 5.3 shows the laser beam projected towards mirror 1 and then is reflected a fixed distance 'd' towards mirror 2. As shown in figure 5.3 an adjustment of mirror 1 by θ affects



ROTATION MIRROR 1
FIGURE 5.3

the reflection by two times theta. This can be described as a



ROTATION MIRROR 1
FIGURE 5.4

right triangle in figure 5.4. The hypotenuse represents the reflection of the beam off of mirror 1. There are two things

to determine from the reflection 1) its vector equation and 2) the location that it projects to on the second mirror. The $(XYZ)^T$ location on mirror 2 is the $(X_1Y_1Z_1)^T$ needed as part of the symmetric equation. While this location was relatively quick and easy to determine, the vector components $(lmn)^T$ of the final projection are not.

Equations 5.2 through 5.4 derive 'x' as a function of theta 1.

$$x = hyp \cdot \sin(2\theta_1) \quad 5.2$$

$$d = hyp \cdot \cos(2\theta_1) \Rightarrow hyp = \frac{d}{\cos(2\theta_1)} \quad 5.3$$

$$x = d \frac{\sin(2\theta_1)}{\cos(2\theta_1)} \Rightarrow d \cdot \tan(2\theta_1) \quad 5.4$$

This gives the location needed for the symmetric equation as:

$$\begin{aligned} X_1 &= d \cdot \tan(2\theta_1) \\ Y_1 &= -d \\ Z_1 &= 0 \end{aligned} \quad 5.5$$

The direction of the reflection vector is:

$$\begin{aligned} l &= \cos(90 - 2 \cdot \theta_1) = -\sin(2 \cdot \theta_1) \\ m &= \cos(180 + 2 \cdot \theta_1) = -\cos(2 \cdot \theta_1) \\ n &= \cos(90) = 0 \end{aligned} \quad 5.6$$

This completely describes the reflection of the beam off of the first mirror as a function of theta one.

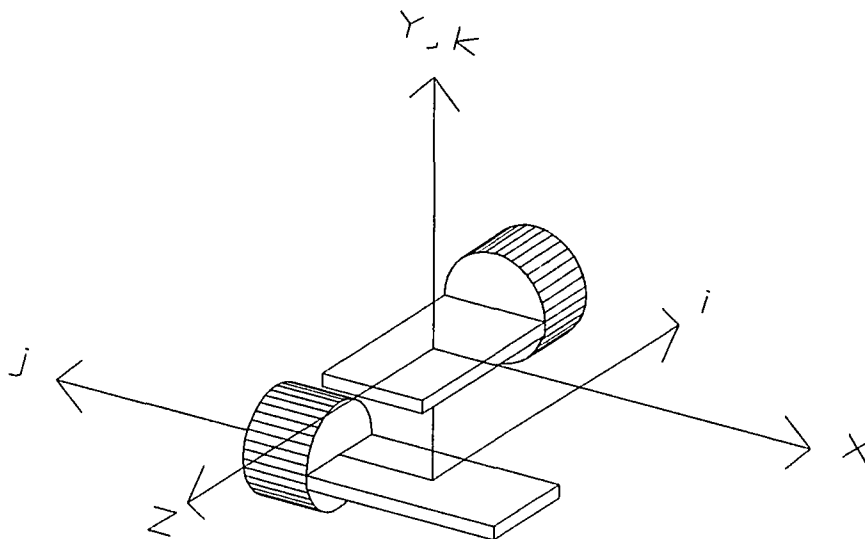
The upcoming steps will be used to find the third projection

into the workspace in terms of the WCS.

- 1) Define a coordinate system for the second mirror (M2CS),
- 2) Define an adjacent frame transformation to convert between the WCS and the M2CS,
- 3) Convert the equation of the second projection from the WCS to the M2CS,
- 4) Determine the equation of the second reflection in M2CS, and finally
- 5) Convert the equation of the second reflection back to the WCS as the third and final projection.

This will give all the necessary information to solve for the symmetric equation.

In order to determine the equation of the second reflection it is necessary to take it about a coordinate system relative to



MIRRORS 1 AND 2
FIGURE 5.5

the second mirror. In figure 5.5 a Mirror 2 Coordinate System (M2CS) established for the second mirror.

In this part this reflection must be described in terms of the Mirror2 Coordinate System (M2CS). Once in the M2CS the reflection off of mirror 2 can be easily determined as the beam is projected into the 3-D workspace. The equation of this final projection is finally converted from M2CS back to the WCS.

An adjacent transformation matrix can be derived to redefine coordinates and vectors between the WCS and the M2CS. In figure 5.5 the orientation shown of i is negative to Z, j is negative to X, and k is coincident with Y. The M2CS is translated beneath the WCS by a distance of '-d' along the Y axis. This can be expressed as a matrix $A_{ijk \rightarrow XYZ}$ converting from the M2CS to the WCS.

$$A_{ijk \rightarrow XYZ} = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & -d \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad 5.7$$

Mirror 2 rotates about the j axis and its rotation can be defined by the following transformation matrix:

$$Rot_j = \text{Rotation Mirror 2} = \begin{bmatrix} \cos(\theta_2) & 0 & \sin(\theta_2) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta_2) & 0 & \cos(\theta_2) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad 5.8$$

The adjacent and rotation transformation matrices are combined to create a total transformation that converts between the two

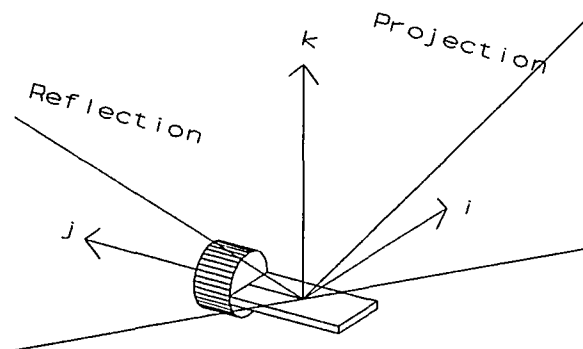
coordinate systems as a function of theta 2.

$$T_{ijk-XYZ} = A_{ijk-XYZ} \cdot Rot_j = \begin{bmatrix} 0 & -1 & 0 & 0 \\ -\sin(\theta_2) & 0 & \cos(\theta_2) & -d \\ -\cos(\theta_2) & 0 & -\sin(\theta_2) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad 5.9$$

This transformation is now combined with the equation of the second projection to express it in terms of the M2CS.

$$\begin{bmatrix} 0 & -\sin(\theta_2) & -\cos(\theta_2) & -d \cdot \sin(\theta_2) \\ -1 & 0 & 0 & 0 \\ 0 & \cos(\theta_2) & -\sin(\theta_2) & d \cdot \cos(\theta_2) \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -\sin(2 \cdot \theta_1) \\ -\cos(2 \cdot \theta_1) \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \cos(2 \cdot \theta_1) \cdot \sin(\theta_2) \\ \sin(2 \cdot \theta_1) \\ -\cos(2 \cdot \theta_1) \cdot \cos(\theta_2) \end{bmatrix} \quad 5.10$$

This is now the equation of the projection inbound towards mirror 2 in terms of M2CS. The reflection on mirror takes place about the k axis as shown in figure 5.6.



REFLECTION OFF OF MIRROR 2
FIGURE 5.6

The vector components of i & j remain unchanged between the

projection and the reflection. While the magnitude of k remains constant it is reversed in direction by the reflection. This reflection can be described as:

$$\begin{bmatrix} \cos(2 \cdot \theta_1) \cdot \sin(\theta_2) \\ \sin(2 \cdot \theta_1) \\ -\cos(2 \cdot \theta_1) \cdot \cos(\theta_2) \\ 0 \end{bmatrix} \Rightarrow \begin{bmatrix} \cos(2 \cdot \theta_1) \cdot \sin(\theta_2) \\ \sin(2 \cdot \theta_1) \\ \cos(2 \cdot \theta_1) \cdot \cos(\theta_2) \\ 0 \end{bmatrix} \quad 5.11$$

This second reflection is the third and final projection into the 3-D workspace. By using the transformation matrix $T_{ijk \rightarrow XYZ}$, which is the inverse of $T_{XYZ \rightarrow ijk}$ coordinates and vector components can be converted from the M2CS to the WCS.

$$T_{XYZ \rightarrow ijk} = T_{ijk \rightarrow XYZ}^{-1} = \begin{bmatrix} 0 & -\sin(\theta_2) & -\cos(\theta_2) & -d \cdot \sin(\theta_2) \\ -1 & 0 & 0 & 0 \\ 0 & \cos(\theta_2) & -\sin(\theta_2) & d \cdot \cos(\theta_2) \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad 5.12$$

When this is combined with the equation of the projection in the M2CS it is expressed in the desired WCS.

$$\begin{bmatrix} l \\ m \\ n \\ 0 \end{bmatrix} = T_{ijk \rightarrow XYZ} \cdot \begin{bmatrix} i \\ j \\ k \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 & 0 \\ -\sin(\theta_2) & 0 & \cos(\theta_2) & -d \\ -\cos(\theta_2) & 0 & -\sin(\theta_2) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(2 \cdot \theta_1) \cdot \sin(\theta_2) \\ \sin(2 \cdot \theta_1) \\ \cos(2 \cdot \theta_1) \cdot \cos(\theta_2) \\ 0 \end{bmatrix} \quad 5.13$$

Which calculates to be:

$$\begin{bmatrix} 1 \\ m \\ n \\ 0 \end{bmatrix} = \begin{bmatrix} -\sin(2 \cdot \theta_1) \\ -\sin^2(\theta_2) \cdot \cos(2 \cdot \theta_1) + \cos^2(\theta_2) \cdot \cos(2 \cdot \theta_1) \\ -\cos(\theta_2) \cdot \sin(\theta_2) \cdot \cos(2 \cdot \theta_1) - \sin(\theta_2) \cdot \cos(\theta_2) \cdot \cos(2 \cdot \theta_1) \\ 0 \end{bmatrix} \quad 5.14$$

and simplifies to:

$$\begin{bmatrix} 1 \\ m \\ n \\ 0 \end{bmatrix} = \begin{bmatrix} -\sin(2 \cdot \theta_1) \\ \cos(2 \cdot \theta_1) \cdot \cos(\theta_2) \\ -\sin(2 \cdot \theta_2) \cdot \cos(2 \cdot \theta_1) \\ 0 \end{bmatrix} \quad 5.15$$

This equation of the projection is now substituted into the symmetric equation 5.1 along with the location of $(X_1, Y_1, Z_1)^T$ previously derived in the section. The symmetric equation can now be described as a function of theta 1&2, d, and X, Y, & Z.

$$\frac{x + d \cdot \tan(2 \cdot \theta_1)}{-\sin(2 \cdot \theta_1)} = \frac{y + d}{\cos(2 \cdot \theta_1) \cdot \cos(2 \cdot \theta_2)} = \frac{z - 0}{-\sin(2 \cdot \theta_2) \cdot \cos(2 \cdot \theta_2)} \quad 5.16$$

This equation will be used to create equations isolating theta 1&2 when a given X, Y, & Z coordinate is given in the WCS (d is a fixed variable). There are three equations and two unknowns, theta 1&2, for a inputted X, Y, & Z. First these equations are used to create two equations in terms of theta 1&2. They are then used to isolate theta 1 from 2.

The first equation 5.17 is defined as:

$$x + d \cdot \tan(2 \cdot \theta_1) = \frac{-\sin(2 \cdot \theta_1)}{\cos(2 \cdot \theta_1) \cdot \cos(2 \cdot \theta_2)} y + d \quad 5.17$$

which can be expressed as:

$$x + d \cdot \tan(2 \cdot \theta_1) = -\frac{\tan(2 \cdot \theta_1)}{\cos(2 \cdot \theta_2)} y + d \quad 5.18$$

The second equation 5.18 is defined as:

$$x + d \cdot \tan(2 \cdot \theta_1) = \frac{\sin(2 \cdot \theta_1)}{\cos(2 \cdot \theta_1) \cdot \sin(2 \cdot \theta_2)} z \quad 5.19$$

which can be expressed as:

$$x + d \cdot \tan(2 \cdot \theta_1) = \frac{\tan(2 \cdot \theta_1)}{\sin(2 \cdot \theta_2)} z \quad 5.20$$

Equations 5.18 and 5.19 are equated for 'X + d TAN (2*theta_1)'.
(2*theta_1)'

$$y + d = -\frac{\cos(2 \cdot \theta_1) \cdot \cos(2 \cdot \theta_2)}{\cos(2 \cdot \theta_1) \cdot \sin(2 \cdot \theta_2)} z \quad 5.21$$

This simplifies and isolates theta 2 as:

$$y + d = -\frac{z}{\tan(2 \cdot \theta_2)} \quad 5.22$$

Solving for theta 2 is:

$$\theta_2 = \frac{1}{2} \text{Atan2}(-z, y+d) \quad 5.23$$

Where ATAN2 is a function similar to the arctangent but also accounts for the proper quadrant based on the signs of the arguments.

Now that an expression is developed for theta_2 it can be used to solve for theta_1. Using the following equation:

$$x \cdot \sin(2\theta_2) + d \cdot \tan(2\theta_1) \cdot \sin(2\theta_2) - z \cdot \tan(2\theta_1) = 0 \quad 5.24$$

it can be arranged to express theta_1 as:

$$\theta_1 = \frac{1}{2} \text{Atan2}(x \cdot \sin(2\theta_2), z - d \cdot \sin(2\theta_2)) \quad 5.25$$

These two equations 5.23 and 5.25 can solve for proper mirror positions theta_1&2 for a given XYZ. In the upcoming applications the computer program calculates an XYZ coordinate in 3-D space that the laser is to be projected through.

5.3 STEPPER MOTOR CALIBRATION

The equations developed do not account for uncertainties that exist in the projection system. The majority of the uncertainties can be attributed to either installation errors or repeatability of the motors. It is important to account for these uncertainties which, if not, can adversely affect the overall system errors. These errors were accounted for through calibration.

This section will explain 1) how the projection unit is

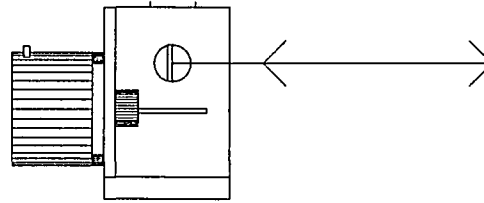
calibrated with itself and its environment and 2) calibrates the indexer motors for positioning errors and repeatability errors. The motors have a consistent inherent positioning error that must be accounted for.

5.3.1 PROJECTOR CALIBRATION

The calibration of the projector deals with properly positioning and aligning the laser emitter. Three words that describe the calibration process are 'Squared', 'Leveled', and 'Centered'. The derived equations assume that the LASER beam projection is level and that it squarely hits the first mirror about the center of rotation. The equations also assume that the two mirrors are level and orthogonal to each other.

The levelness of the LASER beam can easily be checked by projecting it over a long distance and measuring it with a 48" level. This is more reliable than measuring the height of the projection at two points above the floor. It was noticed that the floor isn't entirely level. Once the LASER is leveled the projector must be properly positioned and aligned with the projector.

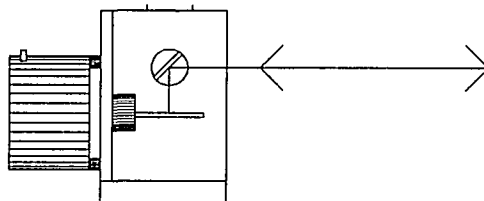
The easiest way to determine if the first mirror was square with the LASER beam was to reflect the beam back at the laser. Once the reflection and projection are coincident the beam is perfectly squared with the beam as shown in figure 5.7.



CALIBRATION 1
FIGURE 5.7

Adjustments can be made to align the beam with the center of rotation of the mirror without affecting squareness.

The second mirror is aligned with the LASER by rotating the first mirror 45 degrees so that it's reflection is towards the second mirror. The second mirror is positioned to reflect the beam back towards the first mirror. Once the final reflection is coincident with the initial projection the second mirror is properly aligned with the LASER beam as shown in figure 5.8.



CALIBRATION 1
FIGURE 5.8

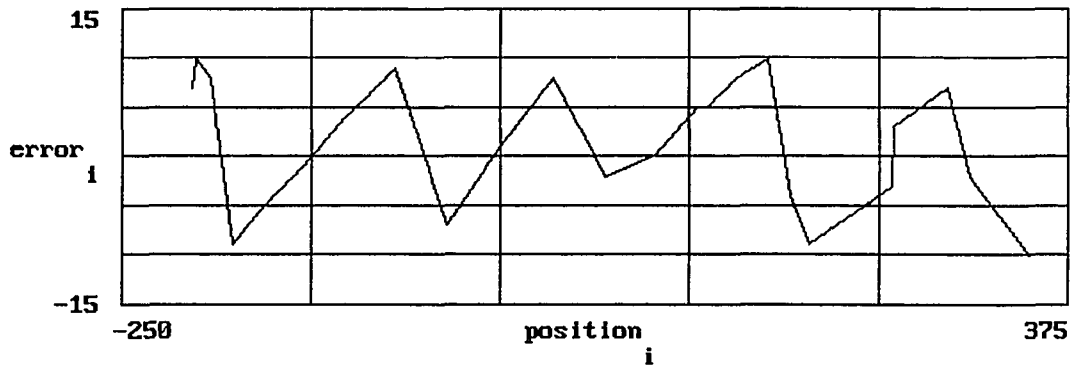
These adjustments can become rather tedious. After aligning one mirror the other mirror's alignment must be checked and realigned. Once the alignments for the first and second mirror can be done without any further adjustments the projection unit is properly aligned with the LASER.

5.3.2 INDEXER MOTOR CALIBRATION

The last conceivable thing that needs to be calibrated are the index motors. As mentioned earlier the indexer motors have 200 discrete steps per 360 degrees of revolution. Between each step the motors are capable of moving to 125 locations giving the motors a capability of 25,000 steps per revolution. Unfortunately the motors do not resolve the 125 steps equally between each discrete step.

Initial analysis of the stepper motors was done by moving the LASER five steps and physically measuring the deflection. The results showed a cyclic compression and refraction of the distance between each increment. Further analysis of the deflection measurements showed two important characteristics that allowed compensation of this error. First the cycle was periodic with a wavelength of 125 steps. This can be directly associated with the 1/125th resolution mode the indexer motors were in during the tests. The second characteristic is the consistency of wavelength regardless of any phase shift. One other important characteristic were the motors repeatability in moving to an exact location. They exhibited very little error when moving from various locations both far and near to a specific point.

Figure 5.9 is a graph of several cycles versus positioning error. The graph shows the periodicity of 125 steps and an



STEPPER MOTOR ERROR PLOT
FIGURE 5.9

amplitude of +/- 10 steps. Some of the cycles are crudely defined due to the lack of data points and also the deviation of the magnitude of the error.

This inherent error was rather simple to compensate for. A target was constructed that had concentric circles on it. The radii varied from 10, 20, and 30 inches. The LASER was commanded to move to locations on each circle that were 10 degrees apart. This was controlled via a separate computer program that specifically written to calibrate the stepper motor. Once the location of the target had been entered in and the desired circle was known by the computer, it would move the LASER in 10 degree arcs about that circle. Due to the error that existed in the stepper motors the LASER spot was usually not exactly on the mark. The user would correct the LASER back to the actual location on the target. Once this point was identified as being the actual location the computer would record the locations the stepper motors had

initially been moved to, the locations they actually had to be moved to, and finally the difference between the two. This method had the advantage of calibrating both motors at the same time.

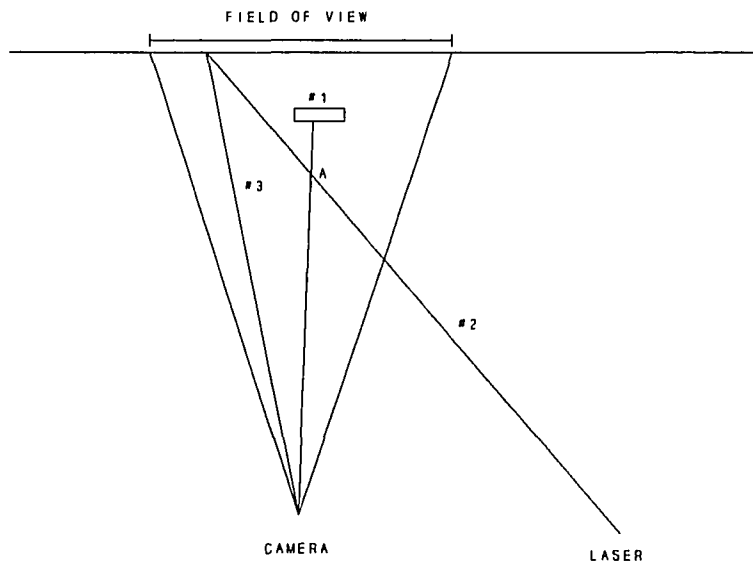
The LASER was commanded to move to hundreds of points in this manner. These points exercised the motors throughout the full range of motion needed for this project. The data was stored in two files (one for each motor). This data was used to construct the the graph in figure 5.9 and was also used to create an error correction table. When the main computer program used for this project commanded either stepper motor to move to a location the position was looked up in the table and moved to the corrected value.

This method provided a quick flexible way of correcting for the error. No experiments were conducted to quantify the amount of improvement, but the test patterns were run using the error correction tables. Some error still existed however the improvement is substantial.

CHAPTER 6

TRIANGULATION OF NON-INTERSECTING LINES

This chapter builds on the results of the previous two chapters. Chapters 4 and 5 work to define the equations of lines from a location on the image plane or the positioning of two orthogonal mirrors. While both chapters go into much detail developing these equations, their ultimate objective is to define the equation of a line. This chapter uses these equations to triangulate an intersection of these lines and return an XYZ point in the WCS. The method of triangulation



DEFINITION OF LINES
FIGURE 6.1

developed in this chapter takes into account that the lines may not mathmatically intersect.

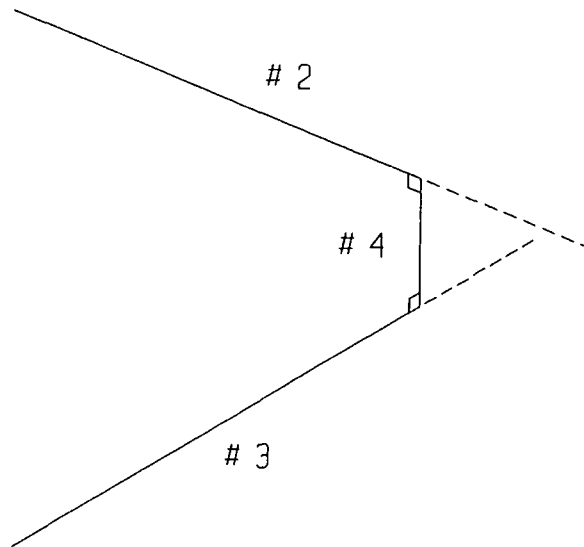
The equations are used when a LASER spot is detected by the camera's image plane. Figure 6.1 clarifies which lines are which and how they are used for this chapter.

Line #1 is selected by the user going from the camera to the object of interest. Line #2 is the projection of the LASER through a selected point 'A' on line #1. The LASER spot of this projection may or may not fall on the object of interest and it may or may not fall within the field of view of the camera. Line #3 goes from the camera to the spot if it is percappable by the camera. This chapter triangulates between lines #2 and #3 to determine the XYZ intersection.

6.1 XYZ LOCATION OF NON-INTERSECTING LINES

Due to the nature of the system it is quite likely that lines #2 and #3 don't mathmatically intersect. Rather they pass extremely close to each other as shown in figure 6.2.

Line #4 is the shortest line that passes through both of the other lines. It is defined as the one and only line that is orthoganal to both lines #2 and #3.[5] The XYZ point returned as the intersection is the mid point of the line segment #4.



DEFINITION OF CROSSING LINES
FIGURE 6.2

6.2 DEVELOPMENT OF THE EQUATIONS

From the work done in the previous chapters lines #2 and #3 can be defined in one of two ways. The first way is in vector form by taking any two points on the line which is expressed as:

$$(X_{end} - X_{beg})i + (Y_{end} - Y_{beg})j + (Z_{end} - Z_{beg})k \quad 6.1$$

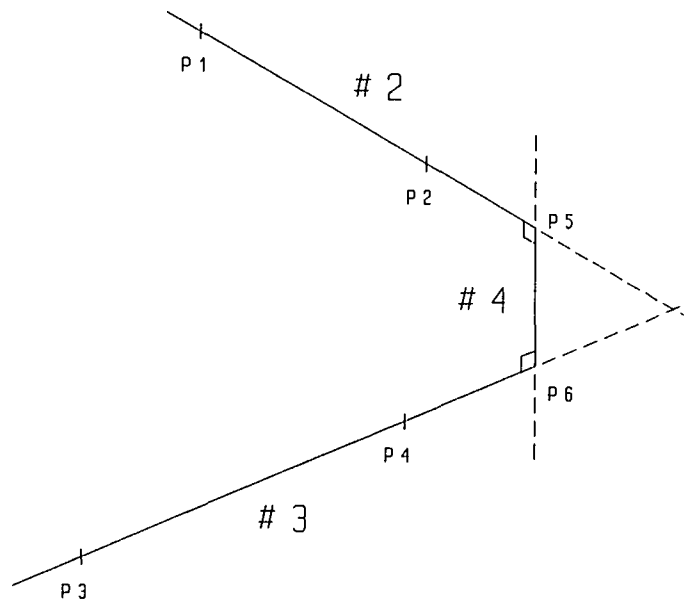
This form defines the orientation of the line in three dimensional space. However in this form it is not a unique vector; the equation represents an infinite number of parallel lines to the given line. The second way is in parametric form which can also be defined by taking the same two points as:

Parametric form defines a unique line in three dimensional

$$\begin{aligned}
 x(t) &= X_{beg} + t(X_{end} - X_{beg}) \\
 y(t) &= Y_{beg} + t(Y_{end} - Y_{beg}) \\
 z(t) &= Z_{beg} + t(Z_{end} - Z_{beg})
 \end{aligned}
 \tag{6.2}$$

space by breaking each coordinate into a separate equation. Any point on the line can be defined for a given value of 't' which can vary between +/- infinity.

This chapter uses both vector and parametric forms to solve for the intersection. The vector form is used to solve for the dot products between lines while the parametric form solves for a unique line #4.



**DEFINITION OF POINTS
FIGURE 6.3**

All three lines are defined with two points as shown in figure 6.3. Points P1 and P2 define line #2 while P3 and P4 line #3. Line #4 is defined by points P5 and P6 where P5 is also on

line #2 and P6 on line #3.

Lines 2, 3, and 4 can be defined in vector form.

$$\begin{aligned} \text{line2} &= x_2i + y_2j + z_2k \\ \text{line3} &= x_3i + y_3j + z_3k \\ \text{line4} &= x_4i + y_4j + z_4k \end{aligned} \quad \mathbf{6.3}$$

where

$$\begin{aligned} X_2 &= X_{P2} - X_{P1} & Y_2 &= Y_{P2} - Y_{P1} & Z_2 &= Z_{P2} - Z_{P1} \\ X_3 &= X_{P4} - X_{P3} & Y_3 &= Y_{P4} - Y_{P3} & Z_3 &= Z_{P4} - Z_{P3} \\ X_4 &= X_{P6} - X_{P5} & Y_4 &= Y_{P6} - Y_{P5} & Z_4 &= Z_{P6} - Z_{P5} \end{aligned} \quad \mathbf{6.4}$$

The parametric equations of lines #2 and #3 to solve for P5 and P6 can be defined as:

$$\begin{aligned} X_{P5} &= X_{P1} + t(X_2) & X_{P6} &= X_{P3} + s(X_3) \\ Y_{P5} &= Y_{P1} + t(Y_2) & Y_{P6} &= Y_{P3} + s(Y_3) \\ Z_{P5} &= Z_{P1} + t(Z_2) & Z_{P6} &= Z_{P3} + s(Z_3) \end{aligned} \quad \mathbf{6.5}$$

At this time the exact location of P5 and P6 on each line has not yet been defined. A value for 't' and 's' represents the XYZ locations for P5 and P6 on lines #2 and #3 respectively.

By definition the dot product of perpendicular lines is zero.

Since lines #2 & #4 and lines #3 & #4 are perpendicular their dot products equal zero.

$$\begin{aligned} \text{line 2} \cdot \text{line 4} &= x_2x_4 + y_2y_4 + z_2z_4 = 0 \\ \text{line 3} \cdot \text{line 4} &= x_3x_4 + y_3y_4 + z_3z_4 = 0 \end{aligned} \quad 6.6$$

This can be expanded to:

$$\begin{aligned} (x_{P2} - x_{P1})(x_{P6} - x_{P5}) + (y_{P2} - y_{P1})(y_{P6} - y_{P5}) + (z_{P2} - z_{P1})(z_{P6} - z_{P5}) &= 0 \\ (x_{P4} - x_{P3})(x_{P6} - x_{P5}) + (y_{P4} - y_{P3})(y_{P6} - y_{P5}) + (z_{P4} - z_{P3})(z_{P6} - z_{P5}) &= 0 \end{aligned} \quad 6.7$$

The parametric representation of points P5 and P6 are substituted in to the above equation. Next the terms with 't' and 's' are gathered and set equal to the other terms. When simplified it can be written as:

$$\begin{aligned} (x_2^2 + y_2^2 + z_2^2)t - (x_2x_3 + y_2y_3 + z_2z_3)s &= x_2x_0 + y_2y_0 + z_2z_0 \\ (x_2x_3 + y_2y_3 + z_2z_3)t - (x_3^2 + y_3^2 + z_3^2)s &= x_3x_0 + y_3y_0 + z_3z_0 \end{aligned} \quad 6.8$$

where:

$$\begin{aligned} x_0 &= x_{P3} - x_{P1} \\ y_0 &= y_{P3} - y_{P1} \\ z_0 &= z_{P3} - z_{P1} \end{aligned} \quad 6.9$$

This can now be put in matrix form as:

$$\begin{bmatrix} (x_2^2 + y_2^2 + z_2^2) & -(x_2x_3 + y_2y_3 + z_2z_3) \\ (x_2x_3 + y_2y_3 + z_2z_3) & -(x_3^2 + y_3^2 + z_3^2) \end{bmatrix} \begin{bmatrix} t \\ s \end{bmatrix} = \begin{bmatrix} x_2x_0 + y_2y_0 + z_2z_0 \\ x_3x_0 + y_3y_0 + z_3z_0 \end{bmatrix} \quad 6.10$$

Finally 't' and 's' can be solved for by multiplying both sides the inverse of the 2X2 matrix. This is simplified as:

$$\begin{bmatrix} t \\ s \end{bmatrix} = \begin{bmatrix} (x_2^2 + y_2^2 + z_2^2) & -(x_2x_3 + y_2y_3 + z_2z_3) \\ (x_2x_3 + y_2y_3 + z_2z_3) & -(x_3^2 + y_3^2 + z_3^2) \end{bmatrix}^{-1} \begin{bmatrix} x_2x_0 + y_2y_0 + z_2z_0 \\ x_3x_0 + y_3y_0 + z_3z_0 \end{bmatrix} \quad \mathbf{6.11}$$

Once values for 't' and 's' are determined they can be put back into the parametric equations to solve for points P5 and P6. As mentioned earlier P5 and P6 define line segment #4. The returned triangulated XYZ location is just the average between P5 and P6.

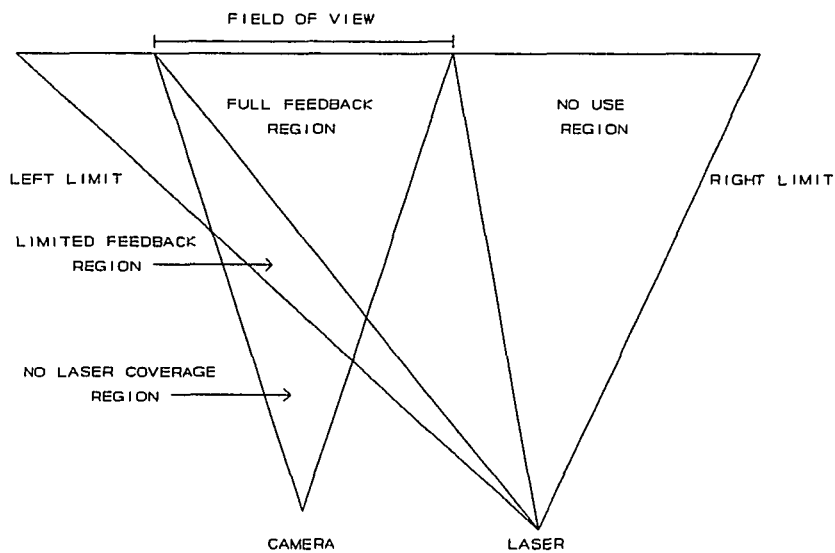
The concept of triangulation is fairly straight forward. However, to compensate for non-intersection of the lines makes these equations just a little more complicated. Yet these equations are easy to implement in a computer program given two points per line.

Now everything that is needed to make this project work has been developed; the next step is to apply it. In the upcoming chapters the working environment will be defined, the optimizational method used, the way the system was calibrated, how data was collected, the results of the data, and finally discuss applications for this work.

CHAPTER 7

DETECTION PRINCIPLES

The equations developed in the previous chapters gives the mathematical description of desired operations, i.e. LASER manipulation or optical detection. These formulas are combined and incoorporated in a computer program that controls the experiment. Before this is done, careful analysis of the system and environment should be performed to develop logical a method for implementing these equations into the program. There are certain physical attributes of the system that can



DEFINITION OF REGIONS
FIGURE 7.1

be used to assist in optimization of a range solution. The

results of the analysis and how they are used is discussed in the upcoming sections of this chapter.

7.1 DEFINITION OF REGIONS

The first step is to examine how the coverage of the LASER projector intersects with the field of view of the camera. Figure 7.1 shows a top down view of the setup. The physical placement between the camera and projector creates four regions defined by the angular deflection of the LASER. Proper analysis of the system is vital to develop an efficient method for implementing the mathematic formulas in the computer program.

The projector's left and right limits are the maximum deflection defining the coverage pattern. As shown in figure 7.1 the LASER covers all but a small area of the camera's field of view.

NO USE / LASER COVERAGE REGIONS - These areas have no intersection between the LASER's coverage and the camera's field of view. These areas cannot be used at all.

FULL FEEDBACK REGION - This triangular area is one of two areas defined by the intersection of the LASER's coverage and the camera's field of view. When the LASER is projected into this area the camera should be able to see it. There are some instances when the camera cannot see the LASER spot because it falls within a shadow created by an object. In this region most methods of optimization can be used to expedite a solution.

LIMITED FEEDBACK REGION - This is the area located from the left side of the FEEDBACK REGION to the left

projector deflection extreme. In this area feed back will not exist until the LASER spot illuminates the object. Once this happens optimization methods can be used to expedite a solution. However until this happens the only way to interrogate this area is by the exhaustive search methods.

While definition of these regions may seem trivial, it leads to a fundamental understanding of what methods to apply for a given situation.

7.2 OPTIMIZATION

"Engineering optimizations theory is a body of mathematical results and numerical methods for finding and identifying the best candidate from a collection of alternatives without having to explicitly enumerate and evaluate all possible alternatives." More simply, optimizations is the ability to find the most accurate answer, within the bounds of significant measurement, while minimizing the amount of data needed to achieve this end.

Unlike in chapter 3 where several variables needed to be solved to find the optimal camera placement, in this section a cost function is formulated for only one variable. The cost function models the system producing an error value for the given independent variables. In order to properly formulate a cost function that can be solved through optimizations one of three conditions must exist; 1) a maximum point, 2) a minimum point, or 3) a zero crossing.

The process begins once the user selects an object on the TV screen. This in turn defines an equation of a line from the camera to the object and beyond terminating at the backwall

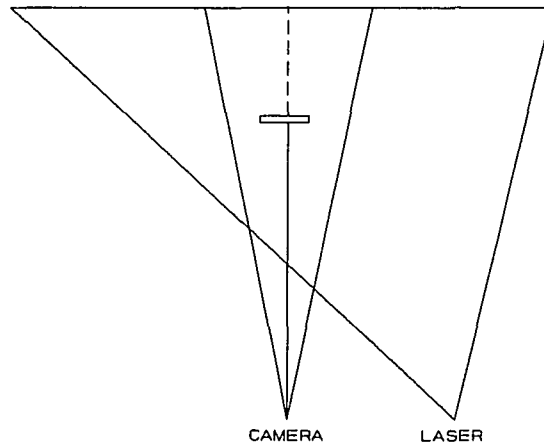


FIGURE 7.2

shown in figure 7.2. The line is parametrically defined in the computer program using 't' as the independent variable. By varying 't' any XYZ point on the line can be defined.

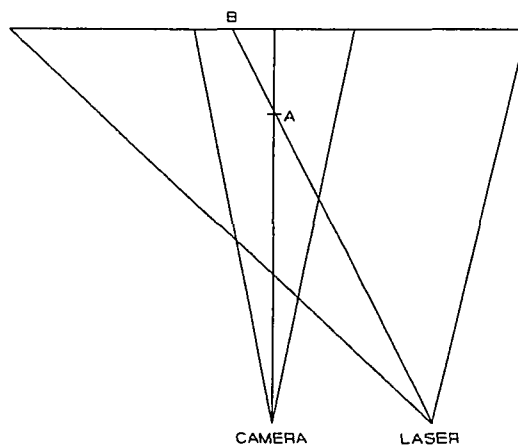


FIGURE 7.3

Once the LASER spot is seen by the camera, the XYZ location of the spot can be ascertained. At this point the locations of two points are known. Figure 7.3 shows point A along the selected line defined by 't' and point B the location of the LASER spot. The cost function is based on the distance between the XYZ point where the LASER is projected through and the XYZ location of the LASER SPOT.

$$N = \text{Distance} = \sqrt{(X_A^2 - X_B^2) + (Y_A^2 - Y_B^2) + (Z_A^2 - Z_B^2)}$$

By varying 't' a succession of points is taken and the cost function can be plotted versus 't'.

Once the line has been selected, bounds for 't' must be established on the line. These bounds can be established anywhere on the line and given any value. Since the intended

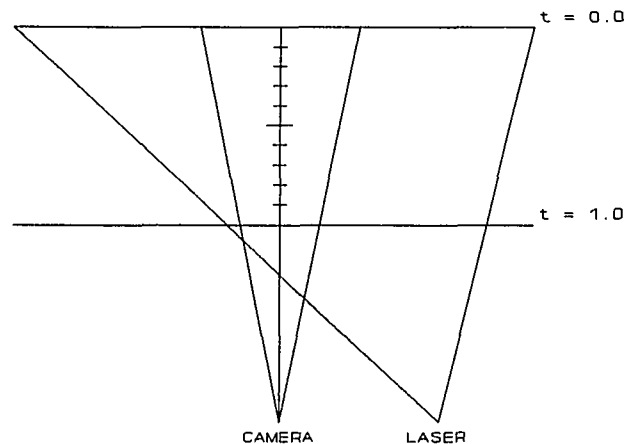


FIGURE 7.4

envelope of the rangefinder is between 10 and 22 feet the

bounds are established based on the Z value of the line. The backwall is established as the starting point at $Z = 22$ feet with $t = 0$ and at $Z = 10$ feet with $t = 1$ shown in figure 7.4. These bounds are established to solve for ' t ' between 0 and 1, however ' t ' can be varied outside the bounds. The next step is to interrogate the line by taking a series of pot shots through it as shown in figure 7.5.

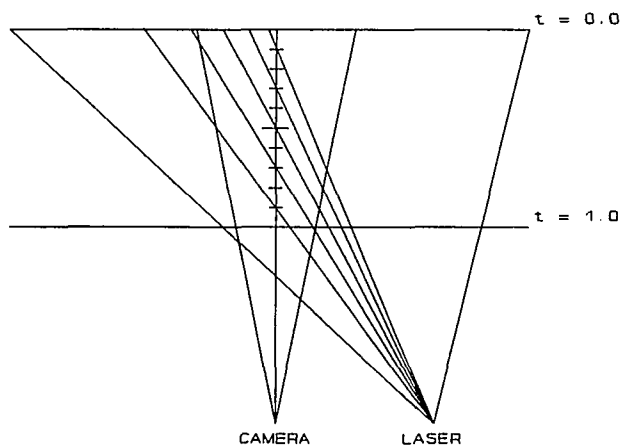


FIGURE 7.5

Figure 7.5 also shows that without any object the cost function converges to the point located on the backwall. Because of this the backwall is the most logical beginning location. If the spot is seen then the XYZ location is on the backwall and the range is solved in one iteration.

When an object does exist it creates a shadow area where the LASER spot can be projected but not be seen by the camera. If

the LASER spot is not seen because of this or it is projected across the field of view it is assumed that it falls on the back wall. Eventually a bound will be found that can eliminate these ambiguous points.

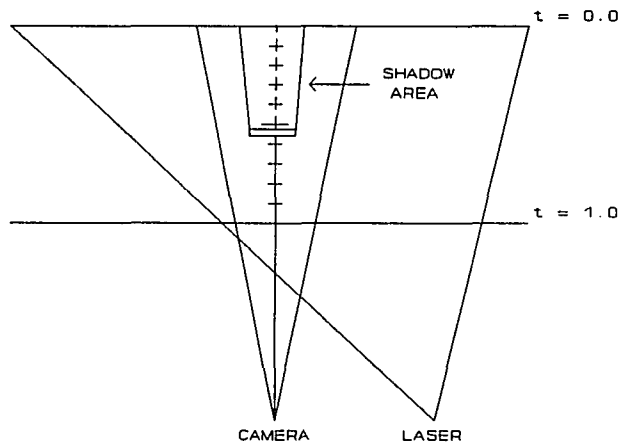


FIGURE 7.6

Putting this all together, in the example shown in figure 7.7, the user wants to know the range to the center of the object.

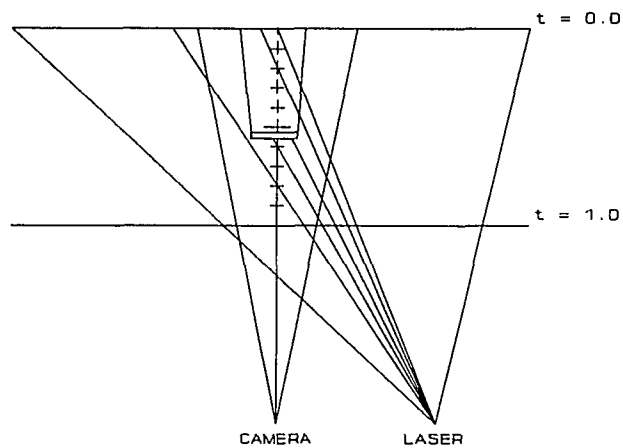


FIGURE 7.7

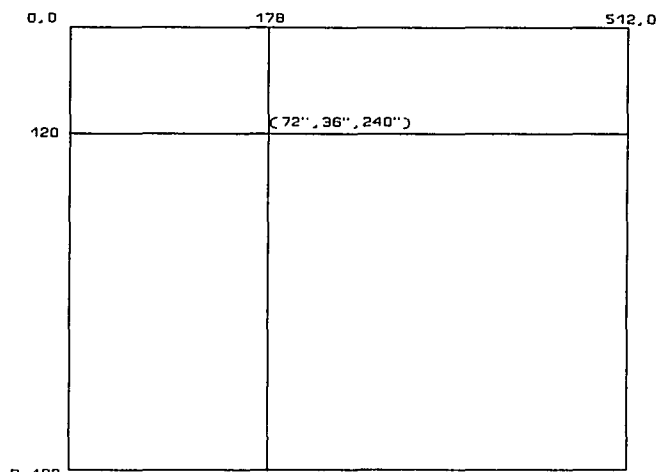
Selecting this point defines a line to and through the object terminating at the back wall. The 't' bounds are established on the line and it is subdivided in to ten equal increments. Starting at $t = 0.0$ and continuing in 0.2 increments the LASER is projected through points on the line for $t = 0.0, 0.2, 0.4, \dots 1.0$. The first two iterations fall in the shadow area and are assumed to fall on the backwall. On the third iteration $t = 0.4$ the first LASER spot is found to the right of the desired point. With this iteration the first two iterations can be eliminated. Continuing on the fourth iteration establishes a bound to the left of the desired point. At this point it is known tht the desired location is between $t = 0.4$ and 0.6. Any number of optimization routines can be run to further determine the location of the desired location. The fifth iteration exemplifies a case where the LASER is projected across the field of view. If needed it could be used as a bound until a more definitive one can be acquired.

The method used is to first interrogate the line from $t = 0.0$ to one in 0.2 increments or until two bounds are established. If at the end of the first interrogation the line can be reinterrogated from $t = 0.9$ to 0.1 in 0.2 increments. If at this point no bounds have been established then the object is assumed to exist in front of the operating envelope.

7.3 ALTERNATE OPTIMIZATION METHOD

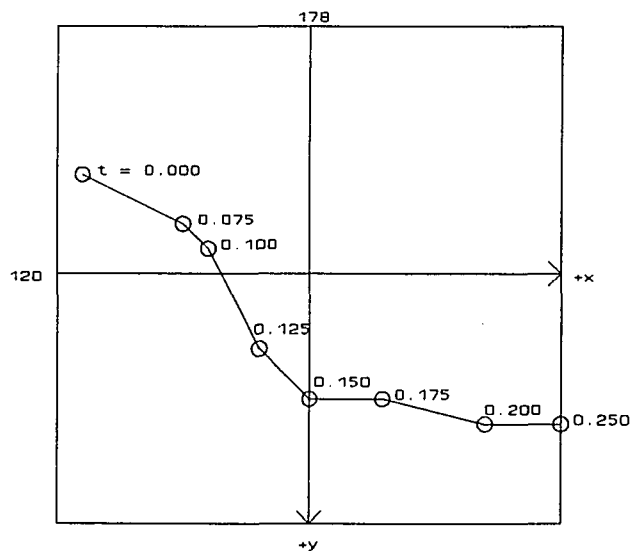
The previous section develops the cost based on the difference of two XYZ points 1) the point in space that the LASER is projected through and 2) location of the spot if seen by the camera in the WCS. This section develops an alternative method for determining the optimal location. The reason for this method will become apparent, but the main reason is based on the inherent error of the stepper motors.

Taking a typical point (72.0,36.0,240.0) the optimal location was found to be (72.2,36.5,237.9). This result is taken from Chapter 8 Results / Analysis which will describe in more detail how this was calculated. The error between the actual and optimal values is 2.177 inches. Initial analysis of the system showed that another set of optimal values could be attained by tracking the LASER spot as it crosses the image plane.



DEFINITION OF SELECTED PIXEL
FIGURE 7.8

Figure 7.8 shows the image plane and the location of pixels in the x and y axis. The (0,0) location is in the upper lefthand corner and the positive direction of the y axis is oriented downward. A location was selected on the monitor at the pixel location (178,120). This point on the image plane was known to correspond to a defined point in the WCS at (72.0,36.0,240). A coordinate system is established with the origin located at the selected pixel.



**TRACK OF LASER SPOT
FIGURE 7.8**

The location of LASER spot is tracked, in figure 7.8, as it progressed across the newly established coordinate system. Each spot location is indexed with its value of t. These values are taken from the example in table 8.1. Ideally if there were no error, this progression should go directly

through the origin of this axis system. When the spot crosses either the x or y axis these can be calculated as zero crossings which in turn can be used as optimal solutions. As the LASER crosses the ordinate an optimal value of (72.3,36.5,236.5) with an associated error of 3.548 inches. For the abscissa the optimal value is (71.5,36.0,239.8) with an error of 0.588 inches.

There are three different optimal values with three different error values. The upcoming chapter will examine the accuracy of these different methods and determine if one is statistically better than the others to cope with the error.

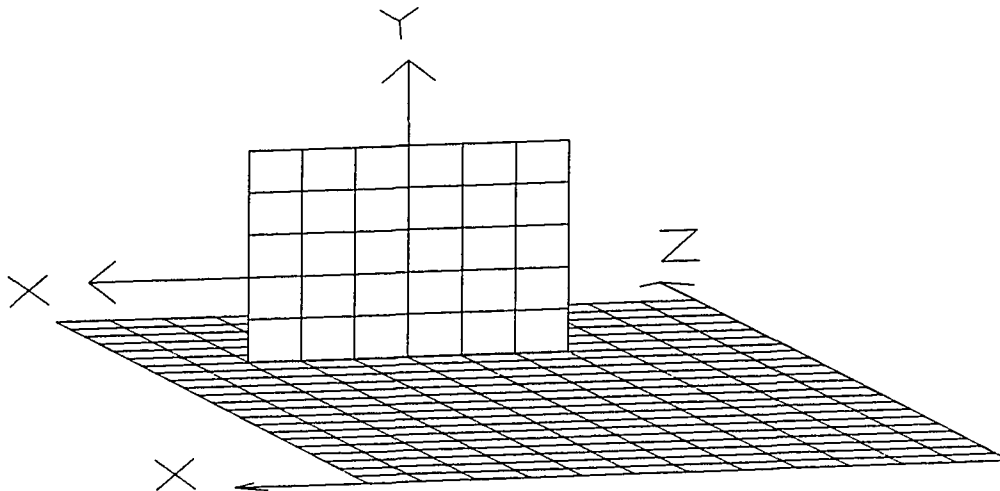
CHAPTER 8

RESULTS / ANALYSIS

This chapter implements what has been developed in the chapter seven. As mentioned in chapter 7 there are three methods for optimizing the location; one based on difference the of XYZ points and the other two based on tracking the LASER spot on the image plane. The primary goal of this chapter is to find out the system's accuracy. A secondary goal is to determine if any of the optimization methods are significantly better or worse than the others. Ideally, if there was no error in the system, all three methods would converge to the exact same 't' value. The third and final goal are to check for any bias in the methods and measure their dispersion.

8.1 COLLECTION OF DATA

For this project a series of data points were collected to test the system through its operating envelope. First, a way of defining the XYZ location in the WCS had to be devised. This was done by gridding the floor as the X&Z plane in one foot increments. Next a six foot by five foot moveable target was constructed and gridded in one foot increments representing the X&Y plane. This target could easily be moved to any location on the floor defining any 3-D point. Figure 8.1 shows how this was set-up and how the axes are defined.



TARGET SETUP
FIGURE 8.1

One assumption made is that the XYZ locations defined by the target are the actual values in the 3-D space. While it is known that these are not the 'exact locations', when taking into account the system resolution, this method provided an acceptable means of defining locations with a high degree of repeatability. There was no way to quantify the accuracy. Subjectively, this method could define a point within +/- 1/4 to 1/2 of an inch.

Each point was collected by selecting an XYZ location defined on the target through the camera. As mentioned before this defines a line from the camera to the desired location. The LASER was then projected through point on the line as a function of 't'. Tabel 8.1 shows the data collected for just one point. The header information shows what the defined XYZ

location of the point is as well as the associated location of this point on the camera's image plane. The data in the other columns is as follows:

t: is used to as an index parameter which is defined along the line that the LASER is projected through.

PROJECTED: is the XYZ location on the line that the LASER is projected through.

TRIANGULATED: is the XYZ location where the LASER spot is observed by the camera.

OBSERVED: is the xy pixel location on the image plane where the LASER spot is seen.

X = 72", Y = 36", & Z = 240" ; x = 178 & y = 120								
t	TRIANGULATED			PROJECTED			OBSERVED	
	X	Y	Z	X	Y	Z	x	y
0.000	70.9	35.7	246.0	74.7	37.8	260.0	187	124
0.075	72.0	36.3	243.8	73.5	37.0	249.5	182	122
0.100	71.2	35.8	239.4	73.2	36.7	246.0	183	121
0.125	72.2	36.5	240.7	72.8	36.4	242.5	180	117
0.150	72.2	36.5	238.5	72.4	36.2	239.0	179	116
0.175	72.3	36.5	236.5	72.0	35.9	235.5	178	115
0.200	72.5	36.2	234.7	71.6	35.7	232.0	176	117
0.225	73.9	36.8	236.6	71.2	35.4	228.5	171	115
0.250	74.7	37.1	236.9	70.8	35.1	225.0	168	114
0.275	76.3	37.1	239.2	70.5	34.9	221.5	163	111
0.300	77.0	38.1	239.2	70.1	34.6	218.0	160	110

TABLE 8.1

The data set is comprised of points taken at different ranges one at Z = 240 inches, 144 inches and 96 inches. Each of these data sets had equal spacing of the points such that

their coverage of the camera's field of view is uniform.

8.2 METHODS

This section goes through step by step how the optimal value for the range is obtained for each method for a single point. In the next section the results for the system will be obtained by examining all the points in the data set. At the optimal value for each method, there are two solutions for the range. In addition to that the error will be examined in 2-D and 3-D. This means that for even one point twelve calculated results will be examined.

Because of the number of calculations it is important to understand what each one means and where it comes from. Starting with three optimization methods, they are defined and will be referred to as:

METHOD 1: This method determines the optimal value from the difference between XYZ two points in the WCS. Referring back to figure 7.3 point A is the selected point on the line that the LASER is projected through. Point B is the location that the LASER spot is seen by the camera and its XYZ location can be ascertained by triangulation. The error function is the distance between these two points.

METHODS 2 & 3: These two methods are very similar in determining the optimal value from tracking the LASER spot location across the image plane. Referring to figure 7.8 a coordinate system is established on the selected pixel. The optimal value is obtained as the LASER spot crosses the axis. Once two spot locations straddle an axis the optimal value can be interpolated. Method 2 is associated with crossing the x axis and method 3 with the y axis.

Because error exists in the system, these methods are being examined to see if which method is less sensitive to error. Method one is the main method devised for this project, however methods two and three provide equally feasible solutions. Each method may have its optimal value at different values of 't'.

All of the data in table 8.1 is collected by the computer. This data can easily be manipulated by the computer as shown in table 8.2. The value for method one is the subtraction of

DIFFERENCES			
t	METHOD 1 (inches)	METHOD 2 (pixels)	METHOD 3 (pixels)
0.000	-14.656	-9	-4
0.075	-6.954	-5	-2
0.100	-5.931	-4	-1
0.125	-1.863	-2	3
0.150	-0.488	-1	4
0.175	1.120	0	5
0.200	4.699	3	5
0.225	8.650	7	5
0.250	12.680	10	6
0.275	18.850	15	9
0.300	22.567	18	10

TABLE 8.2

line segment OA from segment OB. Likewise methods two and three are the difference of xy pixel locations on the image plane. The optimal values for each method can be obtained

through interpolation. As can be seen each method has a different optimal value of 't'. Initial inspection shows that methods one and two are more similar than method three.

At each optimal value there are two solutions per method.

SOLUTION 1: This solution uses the triangulated XYZ location of the LASER spot in the WCS.

SOLUTION 2: This solution uses the point on the line that the LASER is projected through as the XYZ location.

There are good justifications for considering the value of solution two separately from that of solution one. First if the system was free of error, solution one and two would be the exact same at the optimal value of 't'. However it is reasonable to conclude that some error does exist in both the placement of the camera and also in the operation of the stepper motors. When coupled together it is logical to conclude that even if the error is minimal, at the optimal value of 't' solutions 1 and 2 are two different locations. Furthermore solution 1 incorporates both camera and projector error in the location of XYZ at the optimal value of 't' while the optimal location for solution 2 incorporates only the error of camera placement.

Table 8.3 summarizes the calculated XYZ locations, in inches, for each method and each solution at the associated optimal value.

METHOD #1; OPTIMAL LOCATIONS			
	X	Y	Z
DEFINED	72.0	36.0	240.0
SOLUTION 1	72.2	36.5	237.9
SOLUTION 2	72.3	36.1	237.9
METHOD #2; OPTIMAL LOCATIONS			
	X	Y	Z
DEFINED	72.0	36.0	240.0
SOLUTION 1	72.3	36.5	236.5
SOLUTION 2	72.0	35.9	235.5
METHOD #3; OPTIMAL LOCATIONS			
	X	Y	Z
DEFINED	72.0	36.0	240.0
SOLUTION 1	71.5	36.0	239.8
SOLUTION 2	73.1	36.6	245.1

TABLE 8.3

These values are determined by interpolating their values from table 8.2 based on the optimal value of 't'.

Because of all the numbers in table 8.3 it is hard to assess the accuracy of each method/solution result. The error is expressed in inches and defined as the difference between the method/solution calculated XYZ location and the actual XYZ location defined by the target. In reporting the error it can

be broken into two forms; 1) deflection error and 2) ranging error. The deflection error is 2-D error in the XY plane accounting for errors parallel to the image plane while the ranging error is 1-D error perpendicular to the the image plane. Total error is the 3-D error made up of the combination of range and deflection errors. Table 8.4 reports the deflection and total errors for each method/solution combination.

METHOD #1; ERRORS (inches)		
	DEFLECTION ERROR (2-D)	TOTAL ERROR (3-D)
SOLUTION 1	0.551	2.177
SOLUTION 2	0.299	2.083
METHOD #2; ERRORS (inches)		
	DEFLECTION ERROR (2-D)	TOTAL ERROR (3-D)
SOLUTION 1	0.583	3.548
SOLUTION 2	0.100	4.501
METHOD #3; ERRORS (inches)		
	DEFLECTION ERROR (2-D)	TOTAL ERROR (3-D)
SOLUTION 1	0.551	0.588
SOLUTION 2	1.265	5.279

TABLE 8.4

From table 8.4 it might seem as if the triangulation method was the best for 3-D location of the methods. It is important to keep in mind that this is the result from only one point at

one range. In the next section a series of points taken at a variety of ranges will be used to examine the different method/solution combinations.

8.3 ANALYSIS OF METHODS

This section examines the errors for each method for several points to determine if one is better than the other. This is done by calculating the error and the dispersion of the error which in turn are used to calculate the significance of bias.

The data set consists of a series of points at three ranges; $Z = 240"$, $144"$, & $96"$. As shown in section 8.2, the errors are calculated for each point, and the individual errors are grouped into subsets where statistics can be employed to determine performance. In this section three subsets will be defined by range. This allows us to examine the system performance as a function of range. The fourth set analyzes the entire data set allowing us to determine the overall system performance throughout the entire envelope.

The statistics used to analyze each set are fundamental equations: mean and standard Deviation. The mean is used to check for the collective error of the set. The standard

deviation is used to determine the dispersion of the individual errors about the mean. From these basic formulas substantial inferences can be made about the performance.

Table 8.5 shows the mean deflection error and standard deviation, both in inches, for each method primarily grouped by range.

DEFLECTION ERROR COMPARISON AT RANGE				
RANGE	METHOD	SOLUTION	MEAN (inches)	STANDARD DEVIATION
240 INCHES	1	1	0.745	0.640
		2	0.679	0.657
	2	1	0.622	0.597
		2	0.823	0.430
	3	1	0.750	0.537
		2	0.930	0.751
144 INCHES	1	1	0.916	0.509
		2	0.792	0.520
	2	1	0.780	0.515
		2	1.161	0.326
	3	1	0.963	0.537
		2	1.040	0.668
96 INCHES	1	1	0.936	0.284
		2	0.619	0.203
	2	1	0.623	0.193
		2	1.923	1.105
	3	1	2.311	3.409
		2	0.892	0.443

TABLE 8.5

Table 8.6 is the similar to 8.5 except that it is the total 3-D error.

TOTAL ERROR COMPARISON AT RANGE				
RANGE	METHOD	SOLUTION	MEAN (inches)	STANDARD DEVIATION
240 INCHES	1	1	4.647	2.870
		2	5.311	3.129
	2	1	4.630	2.878
		2	3.990	2.210
	3	1	4.637	2.923
		2	3.540	1.707
144 INCHES	1	1	2.576	1.439
		2	2.846	1.722
	2	1	2.533	1.458
		2	2.168	0.832
	3	1	2.667	1.504
		2	2.805	2.114
96 INCHES	1	1	2.448	1.100
		2	3.136	1.412
	2	1	2.362	1.110
		2	3.063	1.059
	3	1	2.684	1.235
		2	1.619	1.216

TABLE 8.6

Tables 8.7 and 8.8 are the collective errors for all the points in the data set.

DEFLECTION ERROR (2-D) COMPARISON OVERALL (inches)				
RANGE	METHOD	SOLUTION	MEAN (inches)	STANDARD DEVIATION
OVER ALL	1	1	0.847	0.549
		2	0.713	0.551
	2	1	0.686	0.519
		2	1.130	0.689
	3	1	1.100	1.574
		2	0.971	0.673

TABLE 8.7

TOTAL ERROR (3-D) COMPARISON OVERALL (inches)				
RANGE	METHOD	SOLUTION	MEAN (inches)	STANDARD DEVIATION
OVER ALL	1	1	3.514	2.427
		2	4.038	2.715
	2	1	3.476	2.445
		2	3.165	1.841
	3	1	3.583	2.451
		2	2.943	1.926

TABLE 8.8

The previous four tables have documented lots of numbers whose importance and interrelation may not be too clear. The questions needed to be answered from these tables are 1) do any methods have any biases and 2) is any one method

statistically more accurate than the others.

The Empirical Rule in statistics states that 68% of all values fall within +/- 1 standard deviation, 95% within +/- 2 standard deviations, and essentially all are within +/- 3 standard deviations. For this project a 95% significance level (corresponding to +/- 2 standard deviations) was used as the threshold for determining bias.

Upper and lower confidence intervals are established about the mean and are functions of the significance level. They are defined as:

$$CI_{(Upper, \frac{\alpha}{2})} = Mean + 2.0 \cdot Standard\ Deviation$$

$$CI_{(Lower, \frac{\alpha}{2})} = Mean - 2.0 \cdot Standard\ Deviation$$

If the lower confidence interval is greater than zero then a bias exists. Likewise if the lower confidence interval is less than or equal to zero then no bias exists. In the overall categories the lower confidence intervals are less than zero in every method/solution combination thus none of them have a bias. However as a function of range biases become prevalent.

When examining the standard deviation they are rather large in comparison to the error. Because of this no method/solution combination is statistically better or worse than any of the

others.

Ideally the method presented should all converge to the same point if everything was calibrated to exact perfection. Realistically this high degree of perfection is unattainable. However, the variation of errors points to saying that this system can probably be more finely tuned.

Most of the error can be attributed to the indexer motors. Currently the motors directly drive the the mirror's positions. Since speed of positioning was not a problem of the motors they could be connected to the mirrors through a reduction gear of about 10:1 ratio. This would increase the accuracy by 10 times, reduce the positioning error and internal friction errors by 1/10th. It would incur a gear meshing error that currently does not exist in the present configuration.

CHAPTER 9

CONCLUSION

There were two objectives addressed in this project to make it practical; 1) reduce the number of iterations and 2) maintain an acceptable determination for the XYZ location. No quantitative limits were established on the objectives to determine success or failure. The average number of iterations used to determine location was eight iterations. Recalling that an iteration consists of projecting the LASER through a XYZ point, processing the image to find any potential LASER spot occurrences, interrogating the occurrences to find the actual LASER spot location, and finally triangulating on the spot to determine the XYZ location. This number of iterations is acceptable by reducing potentially hundreds if not thousands of needless iterations. The XYZ positioning was not as impressive as the reduction of the number of iterations. It has a deflection error of 9/10 of an inch which accounts for a total error of 3.5 inches. This amount of error is too much for precision applications, but is still a viable method for other applications such as navigation. Most of the error can be attributed to the indexer motors used to position the mirrors. As mentioned earlier, the indexer motors have a sinusoidal positioning error. Minimizing this error could significantly improve the

accuracy.

APPENDIX A

Appendix A includes the includes five tables of data that was collected and calculated. Tables 1 through 4 are associated with camera placement. Table 5 is the data set associated with calibration of the total range finding system.

TABLE A.1 108

Table A.1 was collected as the input data for calibration of the camera placement. A series of (X,Y,Z) points in the WCS with their associated pixel location on the image plane. The location of each point on the image plane was converted from pixel coordinates to inches relative to the CCS. This table was used as input for tables A.2 through table A.4.

TABLE A.2 110

Table A.2 takes the (X,Y,Z) and uses the measured values for translation and rotation (see table 4.2) to project each one onto the image plane. Table A.2 compares these values and reports the error in inches and also in pixels.

TABLE A.3**112**

Table A.3 is similiar to table A.2 except that it takes the optimal value returned for camera placement (table 4.3) where each pixel has an equal weight. It also reports just the error in both inches and pixels.

TABLE A.4**114**

Table A.4 is similar to table A.3 except that the pixels in the center are weighted more than the others towards the edge of the image plane.

TABLE A.5**116**

Table A.5 is a list of the 42 data points used to calibrate the system once everything was working. Nineteen points were taken at a distance of 240 inches, 15 at 144 inches and 8 at 96 inches. The index number was used to catagorize each point when the data was collected.

X	XYZ (inches)		xymp (pixels)		xymp (inches)	
	Y	Z	xmp	ymp	xmp	ymp
0	36	240	469	122	12.050	-5.253
0	24	240	469	181	12.050	-2.604
0	12	240	469	241	12.050	0.089
0	0	240	469	302	12.050	2.829
0	-12	240	469	361	12.050	5.478
0	-24	240	468	421	11.990	8.172
12	36	240	421	121	9.347	-5.298
12	24	240	421	181	9.347	-2.604
12	12	240	421	241	9.347	0.089
12	0	240	421	301	9.347	2.784
12	-12	240	420	361	9.290	5.478
12	-24	240	420	421	9.290	8.172
24	36	240	373	121	6.644	-5.298
24	24	240	373	180	6.644	-2.649
24	12	240	373	241	6.644	0.089
24	0	240	373	301	6.644	2.784
24	-12	240	373	361	6.644	5.478
24	-24	240	372	421	6.588	8.172
36	36	240	324	120	3.885	-5.343
36	24	240	324	180	3.885	-2.649
36	12	240	325	240	3.941	0.044
36	0	240	324	300	3.885	2.739
36	-12	240	324	360	3.885	5.433
36	-24	240	323	420	3.829	8.127
48	36	240	275	121	1.126	-5.298
48	24	240	276	180	1.182	-2.649
48	12	240	276	239	1.182	0.000
48	0	240	275	300	1.126	2.739
48	-12	240	275	359	1.126	5.388
48	-24	240	275	419	1.126	8.082
60	36	240	229	120	-1.464	-5.343
60	24	240	229	179	-1.464	-2.694
60	12	240	229	239	-1.464	0.000
60	0	240	228	299	-1.520	2.694
60	-12	240	228	359	-1.520	5.388
60	-24	240	227	419	-1.577	8.082
72	36	240	181	119	-4.167	-5.388
72	24	240	180	179	-4.223	-2.694
72	12	240	180	239	-4.223	0.000
72	0	240	180	298	-4.223	2.649
72	-12	240	180	358	-4.223	5.343

Table A.1

X	XYZ (inches)		xymp (pixels)		xymp (inches)	
	Y	Z	xmp	ymp	xmp	ymp
72	-24	240	180	418	-4.223	8.037
84	36	240	134	119	-6.813	-5.388
84	24	240	133	178	-6.869	-2.739
84	12	240	133	238	-6.869	-0.044
84	0	240	132	297	-6.925	2.604
84	-12	240	132	357	-6.925	5.298
84	-24	240	131	416	-6.982	7.947
96	36	240	86	119	-9.515	-5.388
96	24	240	86	179	-9.515	-2.694
96	12	240	86	238	-9.515	-0.044
96	0	240	87	297	-9.459	2.604
96	-12	240	87	357	-9.459	5.298
96	-24	240	86	416	-9.515	7.947
108	36	240	39	119	-12.160	-5.388
108	24	240	39	178	-12.160	-2.739
108	12	240	39	238	-12.160	-0.044
108	0	240	39	297	-12.160	2.604
108	-12	240	39	357	-12.160	5.298
108	-24	240	39	416	-12.160	7.947

Table A.1 (cont.)

Data set used for the error analysis of the measured values.

xyep (inches)			xyep (pixels)		
xep	yep	magep	xep	yep	magep
-1.92	1.15	2.238	-34.1	25.6	42.65
-1.92	1.18	2.254	-34.1	26.2	43.05
-1.92	1.16	2.246	-34.1	25.9	42.87
-1.92	1.10	2.216	-34.1	24.6	42.08
-1.92	1.13	2.231	-34.1	25.3	42.48
-1.86	1.12	2.173	-33.0	25.0	41.44
-1.89	1.19	2.239	-33.6	26.6	42.90
-1.89	1.18	2.232	-33.6	26.2	42.69
-1.89	1.16	2.224	-33.6	25.9	42.50
-1.89	1.15	2.217	-33.6	25.6	42.31
-1.83	1.13	2.160	-32.6	25.3	41.30
-1.83	1.12	2.153	-32.6	25.0	41.11
-1.87	1.19	2.220	-33.2	26.6	42.58
-1.87	1.22	2.236	-33.2	27.2	43.00
-1.87	1.16	2.205	-33.2	25.9	42.18
-1.87	1.15	2.197	-33.2	25.6	41.98
-1.87	1.13	2.189	-33.2	25.3	41.78
-1.81	1.12	2.134	-32.2	25.0	40.80
-1.79	1.24	2.179	-31.8	27.6	42.14
-1.79	1.22	2.171	-31.8	27.2	41.92
-1.84	1.21	2.209	-32.8	26.9	42.49
-1.79	1.19	2.155	-31.8	26.6	41.52
-1.79	1.18	2.147	-31.8	26.3	41.31
-1.73	1.16	2.092	-30.8	26.0	40.34
-1.71	1.19	2.088	-30.4	26.6	40.41
-1.76	1.22	2.151	-31.4	27.2	41.60
-1.76	1.25	2.169	-31.4	27.9	42.06
-1.71	1.19	2.089	-30.4	26.6	40.44
-1.71	1.22	2.106	-30.4	27.3	40.89
-1.71	1.21	2.098	-30.4	27.0	40.68
-1.80	1.24	2.187	-32.0	27.6	42.27
-1.80	1.27	2.205	-32.0	28.2	42.71
-1.80	1.25	2.197	-32.0	27.9	42.51
-1.74	1.24	2.143	-31.0	27.6	41.56
-1.74	1.22	2.134	-31.0	27.3	41.33
-1.68	1.21	2.079	-30.0	27.0	40.37
-1.77	1.28	2.194	-31.6	28.6	42.62
-1.72	1.27	2.140	-30.6	28.2	41.66
-1.72	1.25	2.131	-30.6	27.9	41.45
-1.72	1.28	2.150	-30.6	28.6	41.92
-1.72	1.27	2.141	-30.6	28.3	41.69

Table A.2

xyep (inches)			xyep (pixels)		
xep	yep	magep	xep	yep	magep
-1.72	1.25	2.133	-30.6	28.0	41.48
-1.81	1.28	2.221	-32.2	28.6	43.07
-1.75	1.31	2.194	-31.2	29.2	42.79
-1.75	1.30	2.185	-31.2	28.9	42.57
-1.70	1.33	2.160	-30.2	29.6	42.33
-1.70	1.31	2.150	-30.2	29.3	42.10
-1.64	1.34	2.125	-29.2	30.0	41.87
-1.78	1.28	2.199	-31.7	28.6	42.71
-1.78	1.27	2.191	-31.7	28.2	42.49
-1.78	1.30	2.209	-31.7	28.9	42.95
-1.84	1.33	2.272	-32.7	29.6	44.15
-1.84	1.31	2.264	-32.7	29.3	43.93
-1.78	1.34	2.237	-31.7	30.0	43.66
-1.82	1.28	2.228	-32.3	28.6	43.17
-1.82	1.31	2.245	-32.3	29.2	43.62
-1.82	1.30	2.237	-32.3	28.9	43.41
-1.82	1.33	2.255	-32.3	29.6	43.88
-1.82	1.31	2.247	-32.3	29.3	43.65
-1.82	1.34	2.265	-32.3	30.0	44.12

Table A.2 (cont.)

Data set used for the error analysis of the equally weighted values.

xyep (inches)			xyep (pixels)		
xep	yep	magep	xep	yep	magep
0.050	-0.079	0.0934	0.88	-1.75	1.971
0.020	-0.008	0.0215	0.35	-0.17	0.397
-0.010	0.013	0.0160	-0.10	0.28	0.332
-0.040	-0.018	0.0438	-0.10	-0.40	0.815
-0.070	0.035	0.0782	-1.24	0.77	1.467
-0.030	0.037	0.0476	-0.53	0.82	0.981
0.021	-0.043	0.0478	0.37	-0.95	1.028
-0.005	-0.021	0.0215	-0.08	-0.46	0.476
-0.030	-0.005	0.0304	-0.53	-0.10	0.543
-0.056	0.005	0.0562	-0.99	0.11	1.001
-0.025	0.009	0.0265	-0.44	0.20	0.487
-0.050	0.007	0.0504	-0.88	0.15	0.901
0.001	-0.052	0.0520	0.01	-1.15	1.158
-0.021	0.011	0.0237	-0.37	0.24	0.446
-0.044	-0.022	0.0493	-0.78	-0.49	0.926
-0.067	-0.017	0.0691	-1.19	-0.37	1.249
-0.089	-0.017	0.0906	-1.58	-0.37	1.626
-0.056	-0.024	0.0609	-0.99	-0.53	1.129
0.046	-0.015	0.0483	0.81	-0.33	0.882
0.026	-0.002	0.0260	0.46	-0.04	0.464
-0.049	0.005	0.0492	-0.87	0.11	0.878
-0.013	0.006	0.0143	-0.23	0.13	0.266
-0.032	0.002	0.0320	-0.56	0.04	0.570
0.004	-0.009	0.0098	0.07	-0.20	0.212
0.100	-0.069	0.1215	1.77	-1.53	2.349
0.027	-0.015	0.0308	0.47	-0.33	0.584
0.010	0.033	0.0341	0.17	0.72	0.748
0.050	-0.015	0.0522	0.88	-0.33	0.948
0.033	0.021	0.0391	0.58	0.46	0.749
0.017	0.006	0.0180	0.30	0.13	0.330
-0.007	-0.033	0.0337	-0.12	-0.73	0.745
-0.021	0.017	0.0270	-0.37	0.37	0.531
-0.035	0.015	0.0382	-0.62	0.34	0.709
0.008	0.008	0.0113	0.14	0.17	0.227
-0.006	-0.005	0.0078	-0.10	-0.11	0.154
0.038	-0.024	0.0449	0.67	-0.53	0.861
0.008	0.003	0.0085	0.14	0.06	0.157
0.053	0.004	0.0531	0.94	0.08	0.945
0.042	-0.002	0.0420	0.74	-0.04	0.747
0.031	0.032	0.0445	0.55	0.71	0.900
0.021	0.014	0.0252	0.37	0.31	0.486

Table A.3

xyep (inches)			xyep (pixels)		
xep	yep	magep	xep	yep	magep
0.010	-0.009	0.0134	0.17	-0.20	0.267
-0.026	-0.005	0.0264	-0.46	-0.11	0.475
0.022	0.036	0.0421	0.39	0.80	0.891
0.014	0.026	0.0293	0.24	0.57	0.625
0.063	0.055	0.0836	1.11	1.22	1.659
0.055	0.034	0.0646	0.97	0.75	1.236
0.104	0.051	0.1158	1.84	1.13	2.169
0.005	-0.014	0.0148	0.08	-0.31	0.324
0.000	-0.022	0.0220	0.00	-0.49	0.490
-0.005	0.009	0.0099	-0.08	0.19	0.212
-0.066	0.034	0.0742	-1.17	0.72	1.396
-0.071	0.008	0.0714	-1.26	0.17	1.274
-0.020	0.022	0.0297	-0.35	0.49	0.605
-0.010	-0.022	0.0241	-0.17	-0.49	0.521
-0.010	0.010	0.0141	-0.17	0.22	0.284
-0.020	-0.008	0.0217	-0.35	-0.18	0.401
-0.020	0.013	0.0238	-0.35	0.28	0.458
-0.020	-0.017	0.0262	-0.35	-0.37	0.519
-0.020	-0.008	0.0215	-0.35	-0.17	0.397

Table A.3 (cont.)

Data set used for the error analysis of the weighted values.

xyep (inches)			xyep (pixels)		
xep	yep	magep	xep	yep	magep
0.060	-0.077	0.0976	1.06	-1.71	2.019
0.030	-0.004	0.0302	0.53	-0.08	0.540
0.000	0.018	0.0183	0.00	0.40	0.407
-0.030	-0.011	0.0319	-0.53	-0.24	0.586
-0.060	0.044	0.0744	-1.06	0.98	1.448
-0.030	0.048	0.0566	-0.53	1.06	1.194
0.026	-0.042	0.0494	0.46	-0.93	1.043
-0.001	-0.018	0.0180	-0.01	-0.40	0.401
-0.028	-0.000	0.0280	-0.49	-0.00	0.497
-0.055	0.011	0.0560	-0.97	0.24	1.007
-0.025	0.017	0.0302	-0.44	0.37	0.583
-0.052	0.017	0.0547	-0.92	0.37	0.998
0.004	-0.052	0.0521	0.07	-1.15	1.160
-0.020	0.012	0.0233	-0.35	0.26	0.445
-0.044	-0.019	0.0479	-0.78	-0.42	0.888
-0.068	-0.012	0.0690	-1.20	-0.26	1.237
-0.091	-0.010	0.0915	-1.61	-0.22	1.632
-0.059	-0.015	0.0608	-1.04	-0.33	1.100
0.047	-0.017	0.0499	0.83	-0.37	0.916
0.026	-0.002	0.0260	0.46	-0.04	0.464
-0.051	0.007	0.0515	-0.90	0.16	0.920
-0.016	0.010	0.0188	-0.28	0.22	0.361
-0.036	0.008	0.0368	-0.63	0.17	0.663
-0.001	-0.001	0.0014	-0.01	-0.02	0.028
0.098	-0.072	0.1216	1.74	-1.60	2.367
0.024	-0.016	0.0288	0.42	-0.35	0.555
0.007	0.033	0.0344	0.12	0.75	0.762
0.045	-0.012	0.0465	0.79	-0.26	0.842
0.027	0.026	0.0374	0.47	0.57	0.751
0.010	0.013	0.0164	0.17	0.28	0.339
-0.011	-0.037	0.0386	-0.19	-0.82	0.846
-0.025	0.015	0.0291	-0.44	0.33	0.555
-0.040	0.015	0.0428	-0.71	0.34	0.788
0.001	0.010	0.0100	0.01	0.22	0.223
-0.014	-0.001	0.0140	-0.24	-0.02	0.249
0.029	-0.018	0.0341	0.51	-0.40	0.652
0.002	-0.002	0.0028	0.03	-0.04	0.056
0.046	0.001	0.0460	0.82	0.02	0.817
0.035	-0.003	0.0351	0.62	-0.06	0.625
0.023	0.032	0.0394	0.40	0.71	0.821
0.011	0.017	0.0202	0.19	0.37	0.426

Table A.4

xyep (inches)			xyep (pixels)		
xep	yep	magep	xep	yep	magep
-0.001	-0.005	0.0050	-0.01	-0.11	0.112
-0.034	-0.012	0.0360	-0.60	-0.26	0.660
0.014	0.032	0.0349	0.24	0.71	0.754
0.005	0.023	0.0240	0.08	0.52	0.532
0.052	0.055	0.0756	0.92	1.22	1.534
0.043	0.035	0.0554	0.76	0.77	1.091
0.092	0.055	0.1072	1.63	1.22	2.042
-0.005	-0.021	0.0215	-0.08	-0.46	0.476
-0.011	-0.027	0.0291	-0.19	-0.60	0.632
-0.016	0.005	0.0168	-0.28	0.11	0.307
-0.078	0.032	0.0843	-1.38	0.71	1.558
-0.084	0.008	0.0843	-1.49	0.17	1.503
-0.034	0.024	0.0416	-0.60	0.53	0.806
-0.020	-0.031	0.0368	-0.35	-0.69	0.776
-0.030	0.004	0.0302	-0.53	0.08	0.540
-0.030	-0.012	0.0326	-0.53	-0.28	0.605
-0.030	0.010	0.0316	-0.53	0.22	0.577
-0.040	-0.018	0.0438	-0.71	-0.40	0.815
-0.040	-0.007	0.0406	-0.71	-0.15	0.727

Table A.4 (cont.)

INDEX #	POSITION (WCS)			PIXEL LOCATION	
	X	Y	Z	xp	yp
1	0	0	240	470	304
2	0	-24	240	468	425
3	0	12	240	470	244
4	0	36	240	470	122
5	24	36	240	373	121
6	24	12	240	372	243
7	24	0	240	372	304
8	24	-24	240	371	425
9	48	-24	240	273	423
10	48	0	240	274	302
11	48	12	240	274	242
12	48	36	240	275	120
15	72	36	240	178	120
16	72	12	240	177	241
19	72	0	240	177	301
20	72	-24	240	177	421
21	96	-24	240	80	418
23	96	0	240	81	299
24	96	12	240	81	239
26	24	-12	144	427	457
27	24	0	144	428	358
28	24	12	144	429	259
29	24	24	144	429	159
31	24	36	144	429	60
33	48	36	144	269	57
34	48	24	144	269	157
35	48	12	144	268	257
37	48	0	144	267	356
38	48	-12	144	267	456
39	72	-12	144	108	452
41	72	0	144	107	353
42	72	12	144	107	254
44	72	24	144	108	155
45	72	36	144	109	57
46	24	0	96	499	427
47	24	12	96	501	280
49	24	24	96	502	130
50	36	24	96	383	128
51	36	12	96	382	278
52	36	0	96	381	427
53	48	0	96	260	425
56	60	24	96	142	126

Table A.5

APPENDIX B

Appendix B contains two main programs used for this project. The first is the program was used to find the optimal location of the camera placement. The second was the program used to operated the system.

BCLSF.FOR

118 thru 123

RANGE.C

124 thru 134

```

C                                     bclsfl.f
C      INTEGER      N, M, LDFJAC
C      PARAMETER    (LDFJAC=80, M=80, N=9)
C
C      INTEGER      IPARAM(7), ITP, NOUT, II
C      REAL         FJAC(LDFJAC,N), FSCALE(M), FVEC(M), RPARAM(7),
&                  X(N), XGUESS(N), ROSBRK, XLB(N), XUB(N),
&                  XS(N), ERROR(N), ERRT
C      EXTERNAL     ROSBRK, UMACH, BCLSF, U4LSF
C      DATA        FSCALE/M*1.0E0/

C X1 = RX      0.00000 +/- .1000
C X2 = RZ      0.00000 +/- .1000
C X3 = RY      0.00000 +/- .1000
C X4 = TX1     41.6875 +/- .0625
C X5 = TY1     17.6250 +/- .2500
C X6 = TZ1     -4.8750 +/- .0625
C X7 = TX2      3.8300 +/--1.0000
C X8 = TZ2      1.8500 +/--1.0000
C X9 = LAM     54.2200 +/--1.0000

XGUESS (1) = 0.0000
XGUESS (2) = -0.0000
XGUESS (3) = 0.0000
XGUESS (4) = 41.6250
XGUESS (5) = 17.6250
XGUESS (6) = -4.8125
XGUESS (7) = 3.7500
XGUESS (8) = 2.0500
XGUESS (9) = 54.2200

ERROR (1) = 0.1000
ERROR (2) = 0.1000
ERROR (3) = 0.1000
ERROR (4) = 0.5000
ERROR (5) = 0.5000
ERROR (6) = 0.5000
ERROR (7) = 0.5000
ERROR (8) = 1.0000
ERROR (9) = 2.0000

```

```

XS (1) = 10.0
XS (2) = 10.0
XS (3) = 10.0
XS (4) = 1.0
XS (5) = 1.0
XS (6) = 1.0
XS (7) = 1.0
XS (8) = 1.0
XS (9) = 1.0

DO 100 II = 1,N
  XLB(II) = XGUESS(II) - ERROR(II)
  XUB(II) = XGUESS(II) + ERROR(II)
100 CONTINUE

ITP = 0

CALL U4LSF (IPARAM, RPARAM)
RPARAM(4) = 10.0EO*RPARAM(4)
IPARAM(3) = 1000
IPARAM(4) = 1000

CALL BCLSF (ROSBRK, M, N, XGUESS, ITP, XLB, XUB, XS,
&          FSCALE, IPARAM, RPARAM, X, FVEC, FJAC,
&          LDFJAC)

ERRT = 0.0

DO 200 II = 1,M
  ERRT = ERRT + FVEC(II)
200 CONTINUE
ERRT = ERRT/60.0

CALL UMACH (2,NOUT)
WRITE(NOUT,99999) IPARAM(3), IPARAM(4),
& RPARAM(1), ERRT
99999 FORMAT (' Iterations = '10X, I3, /,
& ' The number of function evaluations is ', I3,/,
& ' Epsilon = ', F9.4,/,/, ' Average error = ', F9.4//)

DO 205 II = 1,N
  WRITE(NOUT,88888) XLB(II), X(II), XUB(II),
& XGUESS(II), ERROR(II)
205 CONTINUE

88888 FORMAT ( 4F9.4, ' +/-', F9.4)

END
C

```

```

SUBROUTINE ROSBRK (NF, N, X, F)
INTEGER    N, II, NF
REAL      X(N), F(NF), VL(6,20), VL2(6,20),
&         VL3(6,20),XP, YP, SCL, A, B, C, D, E, FF,
&         G, H, M, NN, O, P, VL4(6,20)
REAL X1,X2,X3,X4,X5,X6,X7,X8,X9
DATA VL / 0.0, 36.0, 240.0, 12.05, -5.253, 0.25,
+         0.0, 24.0, 240.0, 12.05, -2.604, 0.25,
+         0.0, 12.0, 240.0, 12.05, 0.0898, 0.25,
+         0.0, 0.0, 240.0, 12.05, 2.829, 0.25,
+         0.0, -12.0, 240.0, 12.05, 5.478, 0.25,
+         0.0, -24.0, 240.0, 11.99, 8.172, 0.25,
+         12.0, 36.0, 240.0, 9.347, -5.298, 0.50,
+         12.0, 24.0, 240.0, 9.347, -2.604, 0.50,
+         12.0, 12.0, 240.0, 9.347, 0.0898, 0.50,
+         12.0, 0.0, 240.0, 9.347, 2.784, 0.50,
+         12.0, -12.0, 240.0, 9.290, 5.478, 0.50,
+         12.0, -24.0, 240.0, 9.290, 8.172, 0.50,
+         24.0, 36.0, 240.0, 6.644, -5.298, 0.75,
+         24.0, 24.0, 240.0, 6.644, -2.649, 0.75,
+         24.0, 12.0, 240.0, 6.644, 0.0898, 0.75,
+         24.0, 0.0, 240.0, 6.644, 2.784, 0.75,
+         24.0, -12.0, 240.0, 6.644, 5.478, 0.75,
+         24.0, -24.0, 240.0, 6.588, 8.172, 0.50,
+         36.0, 36.0, 240.0, 3.885, -5.343, 0.75,
+         36.0, 24.0, 240.0, 3.885, -2.649, 1.00/
DATA VL2/36.0, 12.0, 240.0, 3.941, 0.045, 1.00,
+         36.0, 0.0, 240.0, 3.885, 2.739, 1.00,
+         36.0, -12.0, 240.0, 3.885, 5.433, 0.75,
+         36.0, -24.0, 240.0, 3.829, 8.127, 0.50,
+         48.0, 36.0, 240.0, 1.126, -5.298, 0.75,
+         48.0, 24.0, 240.0, 1.182, -2.649, 1.00,
+         48.0, 12.0, 240.0, 1.182, 0.000, 1.00,
+         48.0, 0.0, 240.0, 1.126, 2.739, 1.00,
+         48.0, -12.0, 240.0, 1.126, 5.388, 0.75,
+         48.0, -24.0, 240.0, 1.126, 8.082, 0.50,
+         60.0, 36.0, 240.0, -1.464, -5.343, 0.75,
+         60.0, 24.0, 240.0, -1.464, -2.694, 1.00,
+         60.0, 12.0, 240.0, -1.464, 0.000, 1.00,
+         60.0, 0.0, 240.0, -1.520, 2.694, 1.00,
+         60.0, -12.0, 240.0, -1.520, 5.388, 0.75,
+         60.0, -24.0, 240.0, -1.577, 8.082, 0.50,
+         72.0, 36.0, 240.0, -4.167, -5.388, 0.75,
+         72.0, 24.0, 240.0, -4.223, -2.694, 1.00,
+         72.0, 12.0, 240.0, -4.223, 0.000, 1.00,
+         72.0, 0.0, 240.0, -4.223, 2.649, 1.00/

```

```

DATA VL3/72.0, -12.0, 240.0, -4.223, 5.343, 0.75,
+ 72.0, -24.0, 240.0, -4.223, 8.037, 0.50,
+ 84.0, 36.0, 240.0, -6.813, -5.388, 0.75,
+ 84.0, 24.0, 240.0, -6.869, -2.739, 0.75,
+ 84.0, 12.0, 240.0, -6.869, -0.045, 0.75,
+ 84.0, 0.0, 240.0, -6.925, 2.604, 0.75,
+ 84.0, -12.0, 240.0, -6.925, 5.298, 0.75,
+ 84.0, -24.0, 240.0, -6.982, 7.947, 0.50,
+ 96.0, 36.0, 240.0, -9.515, -5.388, 0.50,
+ 96.0, 24.0, 240.0, -9.515, -2.694, 0.50,
+ 96.0, 12.0, 240.0, -9.515, -0.045, 0.50,
+ 96.0, 0.0, 240.0, -9.459, 2.604, 0.50,
+ 96.0, -12.0, 240.0, -9.459, 5.298, 0.50,
+ 96.0, -24.0, 240.0, -9.515, 7.947, 0.50,
+ 108.0, 36.0, 240.0, -12.16, -5.388, 0.25,
+ 108.0, 24.0, 240.0, -12.16, -2.739, 0.25,
+ 108.0, 12.0, 240.0, -12.16, -0.045, 0.25,
+ 108.0, 0.0, 240.0, -12.16, 2.604, 0.25,
+ 108.0, -12.0, 240.0, -12.16, 5.298, 0.25,
+ 108.0, -24.0, 240.0, -12.16, 7.947, 0.25/
DATA VL4/24.0, 36.0, 120.0, 11.20, -9.474, 1.00,
+ 24.0, 24.0, 120.0, 11.26, -4.266, 1.00,
+ 24.0, 12.0, 120.0, 11.26, 1.033, 1.00,
+ 24.0, 0.0, 120.0, 11.20, 6.286, 1.00,
+ 36.0, 36.0, 120.0, 5.968, -9.519, 1.00,
+ 36.0, 24.0, 120.0, 5.968, -4.310, 1.00,
+ 36.0, 12.0, 120.0, 5.912, 0.943, 1.00,
+ 36.0, 0.0, 120.0, 5.912, 6.196, 1.00,
+ 48.0, 36.0, 120.0, 0.619, -9.654, 1.00,
+ 48.0, 24.0, 120.0, 0.563, -4.400, 1.00,
+ 48.0, 12.0, 120.0, 0.563, 0.943, 1.00,
+ 48.0, 0.0, 120.0, 0.507, 6.196, 1.00,
+ 60.0, 36.0, 120.0, -4.673, -9.609, 1.00,
+ 60.0, 24.0, 120.0, -4.730, -4.400, 1.00,
+ 60.0, 12.0, 120.0, -4.786, 0.853, 1.00,
+ 60.0, 0.0, 120.0, -4.730, 6.106, 1.00,
+ 72.0, 36.0, 120.0, -9.853, -9.609, 1.00,
+ 72.0, 24.0, 120.0, -9.966, -4.445, 1.00,
+ 72.0, 12.0, 120.0, -10.08, 0.808, 1.00,
+ 72.0, 0.0, 120.0, -10.02, 6.017, 1.00/

```

```

C X1 = Rotation 'Z' = -0.0066 .0243
C X2 = Rotation 'Y' = 0.0300 -.0028
C X3 = Rotation 'X' = 0.0254 .0237
C X4 = Translation 'X1' = 41.6526
C X5 = Translation 'Y1' = 17.5000
C X6 = Translation 'Z1' = -4.8750
C X7 = Translation 'X2' = 3.9300
C X8 = Translation 'Z2' = 3.1800
C X9 = Perspective 'Z' = 54.4100 54.2549

```



```

X1 = X(1)
X2 = X(2)
X3 = X(3)
X4 = X(4)
X5 = X(5)
X6 = X(6)
X7 = X(7)
X8 = X(8)
X9 = X(9)

A = COS(X2)*COS(X3) + SIN(X2)*SIN(X3)*SIN(X1)
B = SIN(X2)*COS(X1)
C = -COS(X2)*SIN(X3) + COS(X3)*SIN(X2)*SIN(X1)
D = COS(X2)*(SIN(X3)*X6 - COS(X3)*X4 - X7) -
&     SIN(X2)*COS(X1)*X5 - SIN(X2)*SIN(X1)*
&     (SIN(X3)*X4 + COS(X3)*X6 + X8)
E = -SIN(X2)*COS(X3) + COS(X2)*SIN(X1)*SIN(X3)
FF= COS(X2)*COS(X1)
G = SIN(X2)*SIN(X3) + COS(X3)*COS(X2)*SIN(X1)
H = SIN(X2)*(COS(X3)*X4 - SIN(X3)*X6 + X7) -
&     COS(X2)*COS(X1)*X5 - COS(X2)*SIN(X1)*
&     (SIN(X3)*X4 + COS(X3)*X6 + X8)
M = -(COS(X1)*SIN(X3)) / X9
NN= SIN(X1) / X9
O = -(COS(X1)*COS(X3)) / X9
P = -SIN(X1)*X5/X9 + COS(X1)*(SIN(X3)*X4 +
&     COS(X3)*X6 + X8)/X9

DO 100 II = 1,20

XP = (A*VL(1,II)) + (B*VL(2,II)) + (C*VL(3,II)) + D
YP = (E*VL(1,II)) + (FF*VL(2,II))+ (G*VL(3,II)) + H
SCL= (M*VL(1,II)) + (NN*VL(2,II))+ (O*VL(3,II)) + P

XP = XP/SCL
YP = YP/SCL

F(II) = ((XP-VL(4,II))**2 + (YP-VL(5,II))**2)**.5
F(II) = VL(6,II)*F(II)

XP = (A*VL2(1,II))+ (B*VL2(2,II))+ (C*VL2(3,II))+ D
YP = (E*VL2(1,II))+ (FF*VL2(2,II))+ (G*VL2(3,II))+ H
SCL= (M*VL2(1,II))+ (NN*VL2(2,II))+ (O*VL2(3,II))+ P

XP = XP/SCL
YP = YP/SCL

```

```
F(II+20) = ((XP-VL2(4,II))**2 + (YP-VL2(5,II))**2)**.5  
F(II+20) = VL2(6,II)*F(II+20)
```

```
XP = (A*VL3(1,II))+ (B*VL3(2,II))+ (C*VL3(3,II))+ D  
YP = (E*VL3(1,II))+ (FF*VL3(2,II))+ (G*VL3(3,II))+ H  
SCL= (M*VL3(1,II))+ (NN*VL3(2,II))+ (O*VL3(3,II))+ P
```

```
XP = XP/SCL  
YP = YP/SCL
```

```
F(II+40) = ((XP-VL3(4,II))**2 + (YP-VL3(5,II))**2)**.5  
F(II+40) = VL3(6,II)*F(II+40)
```

```
XP = (A*VL4(1,II))+ (B*VL4(2,II))+ (C*VL4(3,II))+ D  
YP = (E*VL4(1,II))+ (FF*VL4(2,II))+ (G*VL4(3,II))+ H  
SCL= (M*VL4(1,II))+ (NN*VL4(2,II))+ (O*VL4(3,II))+ P
```

```
XP = XP/SCL  
YP = YP/SCL
```

```
F(II+60) = ((XP-VL4(4,II))**2 + (YP-VL4(5,II))**2)**.5  
F(II+60) = VL4(6,II)*F(II+40)
```

100 CONTINUE

```
RETURN  
END
```

```

#define LINT_ARGS
#include <stdio.h>
#include <middle.h>
#include <struct.h>
#include <conio.h>
#include <debugat.h>
#include "stepper.h"
#include <snap.h>
#include <math.h>
#include <stdlib.h>
#include <string.h>

#define SYNCH 35 /* 35 ms. synchronization pause */
#define s 69.450
#define d .892
#define r_2_d 57.2958
#define pi 3.1415927
#define CLS printf ("\x1B[2J\n")
#define CUP(q,r) printf ("\x1B[%d;%dH",q,r)
#define TRUE 1
#define FALSE 0
#define MEM_SEG 0xD000 /* memory base segment */
#define IO_BASE 0x300 /* I/O base address */
#define RADIUS2 36

struct POINT { unsigned y; unsigned x;};
struct SCRPOINT graph[6];
struct WLDPOINT world[6];
struct DEFLECT angle[6];
struct POINT hipoints[64]; /* storage for the high points' coords */

unsigned char far *window = ( char far * ) ( (long) MEM_SEG << 16);
/* the 'window' onto video memory */
unsigned long int htable[258];

int search (unsigned, unsigned, unsigned char, unsigned);
int zeroed = 0;
int num2 = 0;
int pprcnt = 0;
int ppointer; /* point pointer */

long xsum, ysum; /* for computing x and y means */
char pad [80]; /* storage for indexer replies */
long i1p=0l ;
long i2p=0l ;
long i1pi=0l ;
long i2pi=0l ;
long inc=5l;

float x,y,z;
float xp,yp,xpc,ypc,xpix,ypix,xpixd,ypixd;
float A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P;
float x1,y1,z1,x2,z2,lam,alpha,beta,theta;
float xb1,yb1,zb1,xel,yel,zel;
float xb2,yb2,zb2,xel,yel,ze2;
float xb2c,yb2c,zb2c;
float lt,ltinc,ltoff;
float xlb,ylb,zlb, xlt,ylt,zlt, xle,yle,zle;
float tb,te;
float xpixc = 255.0;
float ypixc = 239.0;
int pos1 [500] [2], pos2 [500] [2];

```

```

/* This program allows the user to align the lazer and calibrate the
camera. An interactive display on the AT shows the lazer position,
and the video camera shows what the camera sees. After the lazer is
zeroed, the user can position the lazer, and ask the AT to find the
2-D coordinates of the dot on the screen. This information, along
with the user-supplied 3-D coordinates, goes into the file CALDATA,
which is later used for determining a 3-D to 2-D transformation
matrix. The program SNAPCAL performs a similar function, but does
not use the lazer or indexers.

```

```

LOCKIN does a FINDOT, blacks out the high-valued pixels, displays
information, and determines if FINDOT correctly found the lazer.
If the user says it's ok, it returns 1, otherwise 0;

```

```

*/
/*****
int
lockin ()
{ pipr(); return (1);}

update()
{
  CLS;
  CUP (25,8);
  printf ("Use Arrow Keys...Home...PgUp,PgDn change increment...END
quits");
  CUP (10,17); printf (" INDEXER 1");
  CUP (15,15); printf ("position: ");
  CUP (15,25); printf ("%08ld ",i1p);
  CUP (19,20); printf ("step:");
  CUP (19,25); printf ("%3ld ",inc);
  CUP (10,47); printf (" INDEXER 2");
  CUP (15,45); printf ("position: ");
  CUP (15,55); printf ("%08ld ",i2p);
  CUP (19,50); printf ("step:");
  CUP (19,55); printf ("%3ld ",inc);
}

update2()
{
  CLS;
  CUP (25,8);
  printf ("F1'-Range 'F2'-Spear Indexer Control 'End+End'-Quits");
  CUP (5,17); printf (" INDEXER 1");
  CUP (7,15); printf ("position: ");
  CUP (7,25); printf ("%08ld ",i1p);
  CUP (9,20); printf ("step:");
  CUP (9,25); printf ("%3ld ",inc);
  CUP (5,47); printf (" INDEXER 2");
  CUP (7,45); printf ("position: ");
  CUP (7,55); printf ("%08ld ",i2p);
  CUP (9,50); printf ("step:");
  CUP (9,55); printf ("%3ld ",inc);
  CUP (11,10); printf ("x-coordinate = %5.1f inches",x);
  CUP (12,10); printf ("y-coordinate = %5.1f inches",y);
  CUP (13,10); printf ("z-coordinate = %5.1f inches",z);
}

```

```

update3()
{
  CLS;
  CUP (25,8);
  printf (" 'F1'-Record 'F2'-New Line 'F3'-Offset 'F4'-tinc 'Del'-quit");
  CUP (2,32); printf ("POINT IDENTIFIER NUMBER: %4d", num2);
  CUP (4,17); printf (" INDEXER 1");
  CUP (5,15); printf ("position: ");
  CUP (5,25); printf ("%08ld ",i1p);
  CUP (6,11); printf ("interpolated: ");
  CUP (6,25); printf ("%08ld ",i1pi);
  CUP (4,47); printf (" INDEXER 2");
  CUP (5,45); printf ("position: ");
  CUP (5,55); printf ("%08ld ",i2p);
  CUP (6,41); printf ("interpolated: ");
  CUP (6,55); printf ("%08ld ",i2pi);
  CUP (8,23); printf ("X      Y      Z");
  CUP (9,11); printf ("t = 0.000  %5.1f %5.1f %5.1f",xlb, ylb, zlb);
  CUP(11,11); printf ("t =%6.3f  %5.1f %5.1f %5.1f",lt+ltoff, xlt, ylt,
    zlt);
  CUP(13,11); printf ("t = 1.000  %5.1f %5.1f %5.1f",xle, yle, zle);
  CUP(10,42); printf ("offset = %5.3f", ltoff);
  CUP(12,42); printf (" t-inc = %5.3f", ltinc);
  CUP(16,11); printf ("Xpc = %3.0f", xpixc);
  CUP(18,11); printf ("Ypc = %3.0f", ypixc);
}

update4()
{
  CUP (25,8);
  printf ("                               Hit Return to Continue");
  CUP(16,21); printf ("Xp = %3.0f", xpix);
  CUP(18,21); printf ("Yp = %3.0f", ypix);
  CUP(16,31); printf ("Xpd = %3.0f", xpixc-xpix);
  CUP(18,31); printf ("Ypd = %3.0f", ypixc-ypix);
  CUP(14,11); printf ("Tri-Range %5.1f %5.1f %5.1f",x, y, z);
  CUP(22,37); printf ("%4d ",pprcnt);
  while(!kbhit());
}

update5()
{
  CLS;
  CUP (25,8);
  printf ("Use Arrow Keys...Home...PgUp,PgDn change increment...END
    quits");
  CUP (10,17); printf ("PIXEL IN  X");
  CUP (15,15); printf ("position: ");
  CUP (15,25); printf ("%3f  ",xpixc);
  CUP (19,20); printf ("step:");
  CUP (19,25); printf ("%3ld  ",inc);
  CUP (10,47); printf ("PIXEL IN  Y");
  CUP (15,45); printf ("position: ");
  CUP (15,55); printf ("%3f  ",ypixc);
  CUP (19,50); printf ("step:");
  CUP (19,55); printf ("%3ld  ",inc);
  CUP (22,37); printf ("%4d ",pprcnt);
}

```

```

/*****

```

```

pikpix()
{
    char c;
    int donn=0;

    pipr();

    update5 ();
    while (!donn) {
        while (!kbhit());
        c = getch ();
        if (c == 0) {
            c = getch ();
            switch (c) {
                case 72 : pipr(); ypixc += (float)inc ; break;
                case 80 : pipr(); ypixc -= (float)inc ; break;
                case 75 : pipr(); xpixc -= (float)inc ; break;
                case 77 : pipr(); xpixc += (float)inc ; break;
                case 73 : inc*=5; break;
                case 81 : inc = ( inc>1 ? inc/5 : 1 ); break;
                case 71 : xpixc = 255.0;
                        ypixc = 239.0;
                        inc = 5;
                        break;
                case 79 : donn = 1 ; break;
                default : putchar (7);
            }
            pipr();
            update5 ();
        }
        else putchar(7);
    }
}

pipr()
{ pprcnt += 1;}

/*****/

```

```
cali()
{
    char c;
    int done=0;

    update ();
    while (!done) {
        while (!kbhit());
        c = getch ();
        if (c == 0) {
            c = getch ();
            switch (c) {
                case 72 : i2p += inc ; break;
                case 80 : i2p -= inc ; break;
                case 75 : ilp -= inc ; break;
                case 77 : ilp += inc ; break;
                case 73 : inc*=5 ; break;
                case 81 : inc = ( inc>1 ? inc/5 : 1 ) ; break;
                case 71 : ilp=i2p=0l ;
                        x = 0.0;
                        y = -1.0;
                        z = 240;
                        break ;
                case 79 : done = 1 ; break;
                default : putchar (7);
            }
            update();
        }
        else putchar(7);
    }
}
/*****/
```

```

calc()
{
    char c;
    int szel, sze2, cnt1, num1;
    int don=0;

    FILE *fptr1,*fptr2,*fptr3,*fp;

    fptr1 = fopen("ndx1cal.dat","r");
    fscanf(fptr1,"%d",&szel);

    for (cnt1 = 0; cnt1 < szel; cnt1 ++ )
    {
        fscanf(fptr1,"%d %d", &pos1 [cnt1] [0], &pos1 [cnt1] [1]);
        printf( "%d %d\n", pos1 [cnt1] [0], pos1 [cnt1] [1]);
    }
    close(fptr1);

    fptr2 = fopen("ndx2cal.dat","r");
    fscanf(fptr2,"%d",&sze2);

    for (cnt1 = 0; cnt1 < sze2; cnt1 ++ )
    {
        fscanf(fptr2,"%d %d", &pos2 [cnt1] [0], &pos2 [cnt1] [1]);
        printf( "%d %d\n", pos2 [cnt1] [0], pos2 [cnt1] [1]);
    }
    close(fptr2);

    fptr3 = fopen("prametr.dat","r");
    fscanf(fptr3,"%f %f %f", &x1, &yy1, &z1);
    fscanf(fptr3,"%f %f %f", &x2, &z2, &lam);
    fscanf(fptr3,"%f %f %f", &alpha, &theta, &beta);
    close(fptr3);

    xe2 = x2*cos(theta) + z2*sin(theta) + x1;
    ye2 = yy1;
    ze2 = -x2*sin(theta) + z*cos(theta) + z1;

    xb1 = 0.0;
    yb1 = -d;
    zb1 = 0.0;

    A = cos(theta)*cos(beta) + sin(theta)*sin(alpha)*sin(beta);
    B = -cos(theta)*sin(beta) + sin(theta)*sin(alpha)*cos(beta);
    C = sin(theta)*cos(alpha);
    E = sin(beta)*cos(alpha);
    F = cos(alpha)*cos(beta);
    G = -sin(alpha);
    I = -sin(theta)*cos(beta) + cos(theta)*sin(alpha)*sin(beta);
    J = sin(theta)*sin(beta) + cos(theta)*sin(alpha)*cos(beta);
    K = cos(theta)*cos(alpha);

    fp = fopen ("caldata","w");

    update2();
}

```



```

while (!don) {
    while (!kbhit());
    c = getch ();
    if (c == 0) {
        c = getch ();
        switch (c) {
            case 59 :
                if(lockin())
                {
                    calcul();
                    fprintf (fp,"%5.1f %5.1f %5.1f %6ld %6ld %d %d\n",
                        x, y, z, ilp, i2p, hipoints[0].x,
                        hipoints[0].y);
                }
                break;
            case 60 : ilp=i2p=01;
                    spear(); break;
            case 72 : i2p += inc ; break;
            case 80 : i2p -= inc ; break;
            case 75 : ilp -= inc ; break;
            case 77 : ilp += inc ; break;
            case 73 : inc*=5 ; break;
            case 81 : inc = ( inc>1 ? inc/5 : 1 ) ; break;

            case 71 : ilp=i2p=01 ;
                    break ;
            case 79 : ilp=i2p=01 ;
                    fclose (fp);
                    don = 1 ;
                    break;
            default : putchar (7);
        }
        update2();
    }
    else putchar(7);
}
}
/*****/

```

```

calcu()
{
    float p1, p2;
    float xm1, xm2;
    float ym1, ym2;
    float zm1, zm2;
    float ai,aj,ak,bi,bj,bk,ci,cj,ck;
    float p11, p12, p22, p21, e1, e2;
    float tt, ss, det;

    p1 = (float)i1p;
    p2 = (float)i2p;

    xb2 = A*xp + B*yp - lam*C + xe2;
    yb2 = E*xp + F*yp - lam*G + ye2;
    zb2 = I*xp + J*yp - lam*K + ze2;

    xe1 = tan((-2.0*pi*p1)/(180.0*s)) * (120.0 +
        d*sin((2.0*pi/180.0)*(p2/s+45.0)));
    ye1 = xe1 / sin((2.0*pi/180.0)*(p2/s+45.0));
    ze1 = 120.0 / tan((-2.0*pi/180.0)*(p2/s+45.0)) - d;
    ze1 = 120.0;

    ai = xe1 - xb1;
    aj = ye1 - yb1;
    ak = ze1 - zb1;

    bi = xe2 - xb2;
    bj = ye2 - yb2;
    bk = ze2 - zb2;

    ci = xb2 - xb1;
    cj = yb2 - yb1;
    ck = zb2 - zb1;

    p11 = ai*ai + aj*aj + ak*ak;
    p22 = -bi*bi - bj*bj - bk*bk;
    p21 = ai*bi + aj*bj + ak*bk;
    p12 = -p21;

    e1 = ai*ci + aj*cj + ak*ck;
    e2 = bi*ci + bj*cj + bk*ck;
    det = p11*p22 - p12*p21;

    tt = (p22*e1 - p12*e2) / det;
    ss = (p11*e2 - p21*e1) / det;

    xm1 = xb1 + tt*(xe1 - xb1);
    ym1 = yb1 + tt*(ye1 - yb1);
    zm1 = zb1 + tt*(ze1 - zb1);

    xm2 = xb2 + ss*(xe2 - xb2);
    ym2 = yb2 + ss*(ye2 - yb2);
    zm2 = zb2 + ss*(ze2 - zb2);

    x = (xm1 + xm2) / 2.0;
    y = (ym1 + ym2) / 2.0;
    z = (zm1 + zm2) / 2.0;
}
/*****/

```

```

spear()
{
  char dd;
  int don2=0;
  FILE *fp2;
  fp2 = fopen("lineopt.dat","a");

  lt = 0.0;

  intline();
  update3();
  while (!don2) {
    while (!kbhit());
    dd = getch ();
    if (dd == 0) {
      dd = getch ();
      switch (dd) {
        case 59 :
          if(lockin())
          {
            calcul();
            update3();
            update4();
            fprintf (fp2,"%4d %6.4f %5.1f %5.1f %5.1f\n",num2, lt, x, y,
              z);
            fprintf (fp2,"%5.1f %5.1f %5.1f %4.0f %4.0f\n", xlt, ylt,
              zlt, xpix, ypix);
            fprintf (fp2,"%4.0f %4.0f %4.0f %4.0f\n", xpixc, ypixc,
              xpixc-xpix, ypixc-ypix);
          }
          break;
        case 60 : intline();
          break;
        case 61 : CLS;
          printf("Enter offset value\n");
          scanf("%f",<toff);
          break;
        case 62 : CLS;
          printf("Enter t increment value\n");
          scanf("%f",<tinc);
          break;
        case 71 : lt = -ltoff; itrsect(); break;
        case 79 : lt = 1.0- ltoff; itrsect(); break;
        case 75 : lt = lt - ltinc; itrsect(); break;
        case 77 : lt = lt + ltinc; itrsect(); break;
        case 83 : don2 = 1;
          fclose(fp2);
          break;
        default : putchar (7);
      }
      update3();
    }
    else putchar(7);
  }
}

```

```

itrsect()
{
    float Q1,Q2,QP1,QP2,QT1,QT2;

    QP2 = 45;
    QP1 = 0;

    xlt = xlb + (lt + ltoff)*(xle-xlb);
    ylt = ylb + (lt + ltoff)*(yle-ylb);
    zlt = zlb + (lt + ltoff)*(zle-zlb);

    Q2 = -.500*atan((zlt)/(ylt+d));
        if (Q2 < 0)
            Q2 += 1.570796;
    Q1 = -.500*atan((xlt*sin(2*Q2))/(zlt+d*sin(2*Q2)));

    Q1 *= r_2_d;
    Q2 *= r_2_d;

    Q2 = Q2 - QP2;
    Q1 = Q1 - QP1;

    i1p = Q1 * s;
    i2p = Q2 * s;
}

intline()
{
    CLS;

    pikpix();
    num2 += 1;

    xpc = 1.254*(xpixc - 255)*.0449;
    ypc = (ypixc - 239)*.0449;

    xb2c = A*xpc + B*ypc - lam*C + xe2;
    yb2c = E*xpc + F*ypc - lam*G + ye2;
    zb2c = I*xpc + J*ypc - lam*K + ze2;

    zlb = 260.0;
    zle = 120.0;

    tb = (zlb-zb2c)/(ze2-zb2c);
    te = (zle-zb2c)/(ze2-zb2c);

    xlb = xb2c + tb*(xe2-xb2c);
    xle = xb2c + te*(xe2-xb2c);
    ylb = yb2c + tb*(ye2-yb2c);
    yle = yb2c + te*(ye2-yb2c);

    ltinc = 0.100;
    ltoff = 0.000;
    lt = 0.000;
    itrsect();
}
/*****/

```

```
main()
{
  update ();
  cali();

  i1p=01;
  i2p=01;

  update2();
  calc();
}
```

BIBLIOGRAPHY

- 1 Burger, P. and Gillies, D., Interactive Computer Graphics, Addison Wesley, 1990.
- 2 Cadillo, J. and Sid-Ahmed, M., "3-D Position Sensing Using a Passive Monocular Vision System", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-13, No. 8, August 1991, pp. 809-813.
- 3 Foley, J., vanDam, A., Feiner, and S., Hughes, J., Computer Graphics Principles and Practices, 2nd Edition, Addison Wesley, 1990.
- 4 Fu, K.S., Gonzales, R.C., and Lee, C.S.G., Robotics: Control, Sensing, Vision, and Intelligence, 3rd Edition, McGraw Hill, 1987
- 5 Grossman, S., Calculus, 2nd Edition, Academic Press, 1981.
- 6 Harrison, D. and Weir, M., "High Speed Triangulation-Based Imaging With Orthonormal Data Projections and Error Detection", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-12, No. 4, April 1990, pp. 409-416.
- 7 Hearn, D. and Baker, M., Computer Graphics, 3rd Edition, Prentice Hall, 1986
- 8 Jarvis, R., "A Computer Vision and Robotics Laboratory", IEEE Computer, June 1982, pp 8-24.
- 9 Jarvis, R., "A Perspective on Range Finding Techniques for Computer Vision", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-5, No. 2, March 1983, pp. 122-139.
- 10 Jones, B., Linear Algebra, 2nd Edition, Holden-Day, Inc., 1973.
- 11 Kahn, P., Kitchen, L., and Riseman, E., "A Fast Line Finder for Vision Guided Robot Navigation", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-12, No. 11, November 1990, pp. 1098-1102.

- 12 Kanade, T. and Asada, H., "Noncontact Visual Three Dimensional Ranging Devices", SPIE 3-D Machine Perception, Vol. 283, 1981, pp. 48-53.
- 13 Liu, Y., Huang, T.S., and Faugeras, O.D., "Determination of Camera Location from 2-D to 3-D Line and Point Correspondences", IEEE Transactions on Pattern Analysis and Machine Intelligence, January 1990, pp. 28-37.
- 14 Mansbach, Peter, "Calibration of a Camera and Light Source by Fitting to a Physical Model", Computer Vision, Graphics, and Image Processing, 1986, pp. 200-219.
- 15 Moring, I., Heikkinen, T., Myllyla, R., and Kilpela, A., "Acquisition of Three-Dimensional Image data by a Scanning LASER Range Finder", Optical Engineering, Vol. 28, No. 8, August 1989, pp. 897-902.
- 16 Nitzan, D., "Three-Dimensional Vision Structure for Robot Applications", Transactions on Pattern Analysis and Machine Intelligence, Vol. 10, No. 3, May 1988, pp. 291-309.
- 17 Parthasarathy, S., Birk, J., and Dessimoz, J., "LASER Range Finder for Robotic Control and Inspection", SPIE Robot Vision Vol. 336, 1982, pp. 2-10.
- 18 Reklaitis, G., Ravindran, A., and Ragsdell, K., Engineering Optimization Methods and Applications, 5th Edition, John Wiley & Sons, Inc., 1983.
- 19 User's Manual IMSL Math/Library, IMSL Problem-Solving Software Systems, 1989.