To the Graduate Council:

I am submitting herewith a thesis written by Min Ching Chiang entitled " Vertex Location for Robotics Applications." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Ernest L. Hall

Ernest L. Hall, Major Professor

We have read this thesis
and recommend its acceptance:

Donald W. Bouldin

Robert E. Bodenheimer

Accepted for the Council:

C W Minkel

The Graduate School

VERTEX LOCATION

FOR

ROBOTICS APPLICATIONS

A Thesis

Presented for The

Master of Science

Degree

The University of Tennessee, Knoxville

Min Ching Chiang

March 1984

DEDICATION

This thesis is dedicated

to my parents, Mr. Yao Chiang & Mrs. Shui-Choo Chiang,

and to my husband, James.

ACKNOWLEDGMENTS

Sincere appreciation is expressed for the patience, guidance, assistance and encouragement of Dr. Ernest L. Hall under whose direction this research was developed into a master's thesis.

Acknowledgement is also made for the reading and comments of the committee members Dr. Bodenheimer and Dr. Bouldin, and to Betty Hall whose suggestions have been very helpful.

The author also expresses her gratitude to her husband James, for many ways in which he contributed to the preparation of this thesis.

ABSTRACT

Machine vision, based on the processing and interpretation of images, is being introduced into many robotics applications. A high-speed algorithm for extracting vertex location is an essential part of the performance of the system. The class of algorithms based on edge detection, small mask vertex extraction and matching line to determine the image coordinates of object surface is considered. The method consists of first performing edge detection on the image which obtained from the camera. An image of the object edges and the distorted grid edges is obtained. The resulting image is then thresholded at a pre-determined gray level to obtain a binary image. Using the small mask vertex extraction method, the vertices of the distorted grid are extracted from the binary image. The true image vertex can be located by using the matching line constraint. An important aspect of this algorithm is that the feature extraction is based upon local statistics to determine the vertices' location of the grid patterns. The significance of this technique is that it demonstrates an economical and high speed vertex location method to obtain the distorted grid pattern of the object surface. This technique plays an important role in many high speed surface measurements as well as in single image surface measurements for robotics applications. The technique has direct application to visually guided robotic systems and some medical treatments which require body surface measurements.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

FIGURE PAGE

1. INTRODUCTION

Machine vision in industrial applications is expanding as designers apply this technology to give robots visual capabilities. With computer vision, robots could identify, classify, sort, inspect and locate objects. Barnaby Feder, New York Times science writer, said that the development of robot vision system is a "key factor in the transformation of robots from valuable but essentially stupid tools into sensitive devices capable of a variety of manufacturing tasks." To be feasible for industrial applications, however, a robot vision system must be relatively inexpensive, simple, fast, accurate and truly three dimensional. Today, applications for machine vision may be fall into three general categories. These are inspection, identification, and guidance and control.

In order for a robot to have access to its environment, it must have the ability to sense errors or visualize the environment. This requires a large number of computations on sensed data to detect features, recognize patterns, or compare the input data with logical expectations. Although many sensory devices have played important roles in modern robots, computer vision appears to offer the richest source of sensory information for intelligent robotics manipulation in the widest range of environments.

Computer vision is the collection of techniques, software, and hardware for measurements and inferences from images [1,2]. A simple computer vision system usually consists of an image-processing unit interfaced with a mini-or mainframe computer system [3]. Many kinds of computer vision are available in the market. A complete computer vision system is available at a price as low as approximately $5000 [4]. However, the majority of these systems are classified as low-level processing systems [5,6]. Low-level processing systems usually consist of image acquisition, edge detection, feature extraction, template matching, and object recognition. Low-level processing is time consuming since it requires a lot of computation. The development of low-cost processors and memory and solid-state imaging devices have allowed the implementation of image-processing algorithms on mini- or mainframe computers [7]. Imaging devices which perform pattern recognition and image processing for a specific task have been widely used in many automated visual inspection systems [8,9].

The Vision Module, developed at SRI International, is a good example of a general-purpose system that performs analysis of binary images [10,11]. The Vision Module consists of a General Electric TN2200 television camera with 128 x 128 resolution, a frame buffer, an interface unit for digitizing the video signal, and a Digital Equipment Computer LSI 11 microcomputer with a 28,000-word, random-access memory. This system is capable of performing dual threshold operation, windowing, connectivity analysis, run-length encoding, and local and global feature extraction. This system is available from Machine Intelligence Corporation.

Another example of a computer vision system is the PIXIE-1000, developed by Applied Intelligent System, that performs analysis of

64 levels of gray-scale images. The vision system consists of a Fairchild charge-couple-device CCD-4000 with 256 x 256 resolution, PIXIE processor, monitor, keyboard and disk drive. This system is capable of identifying and locating objects based upon their size and shape.

In order for a robot to have access to its environment, its vision system must have the ability to provide three-dimensional information. One major application of three-dimensional technology is in robotic seam welding [12]. Another application of three-dimensional vision system is bin picking , where a robot picks up parts which are randomly oriented in a bin or similar container [12].

The measurement of point positions on a three-dimensional surface is a difficult task. In the past, measurements from two images were required to measure the surface of a three-dimensional object. The use of a sequence of range data images corresponding to various views of the object to approximate the surface by polygons is given by Bhanu [13]. He describes a technique for extracting vertices for three-dimensional surface measurements using a projection method. Most of the early work in object location dealt with the interpretation of polygonal surfaces [14,15]. However, a particular class of objects, such as featureless curved objects, has proven to be difficult for recognition or surface measurement [16,17]. Curved or rounded objects which have no unique features on the surface are difficult to distinguish from a two-dimensional image. Unfortunately, this class of objects often encompasses the most important man-made or natural structures. For example, a major task in radiotherapy treatment is to obtain an image of the surface of the patient's body to deliver a uniform dose to a specific region of the patient's body. An excellent paper describing an interactive technique using a single image to measure the body contours of a patient and to obtain the three-dimensional measurement through the use of computer graphics is given by Renner [18]. The use of a two-dimensional tablet to measure the three-dimensional position of points on a surface is given by Parke [15]. The method of using estimation of location and orientation of three-dimensional surfaces using a single two-dimensional image is described in [19]. The use of a vacuum cup to hold the object and present it to a camera for computation of its orientation and for determination of the workpieces was developed at the University of Rhode Island [20].

There are many different approaches that provide the three-dimensional solution necessary for curved object location, recognition, and manipulation. A regional method to represent polyhedra used by Shiria [21] incorporates an approach called "active imaging" which involves a projection system and a camera system. The projection system is used to impose features on the surface of the object. The camera system is used to acquire the object image with the imposed features and to perform the necessary calculations to interpret the image for analysis. Active imaging techniques have been widely used in many military systems. The advantage of the active approach is that it provides a high degree of control over the environment. It can be used as a marker for accuracy of location.

The accuracy of the location of the object with a projected grid system can be controlled by the grid spacing. The ability to accomplish "stereo" imaging with a single image may appear difficult at first consideration. However, if one considers the projected pattern as the first image, and the received image as the second, a standard stereo solution results. The use of coorperative algorithms to determine the surface orientation from a single view is given by Woodham [22]. Using a single camera to recognize overlapping parts is described by Berman [23]. The use of geometric and relational reasoning to recognize three-dimensional objects from a single view is described in [24].

Successful developments in active imaging techniques include the CONSIGHT system developed at General Motors Technical Center [25,26] and the system developed by Albus [27] at the National Bureau of Standards. The CONSIGHT system uses a linear array camera and two projected light lines focused as one line on a belt. The camera detects and tracks silhouettes of passing objects by the displacement of the line on the belt. The system developed by Albus uses a plane of light to determine the position and orientation of parts on a table. The line projection technique works quite well for locating the edges of an object, but requires scanning in order to obtain object surface point measurements.

The use of grid coding for image segmentation based upon the spatial frequencies of the projected image is described by Will and Pennington [28], who trace the history of grid patterns to the present.

The computational techniques associated with a recently developed laser shutter/space encoding system is described by Posdamer and Altschuler [29]. This system employs time and space coding of dots to measure surface coordinates. An introduction to robot manipulators and their computer control is given by Paul [30]. An introduction to both computer vision and robotics is given by Albus [27].

Surfaces, edges, and vertices are the features used to describe the spatial discontinuities of an object in three dimensions. These features also characterize the shape of the object in space. Any object or pattern which can be recognized and classified possesses a number of discriminatory properties or features. The first step in any recognition process, preformed either by a machine or by a human being, is to consider the problem of what discriminatory features to select and how to extract these features such as corners, edges, vertices, etc. The Sight-I system developed by General Motors uses the corners of an intergrated circuit chip to determine the position and orientation of the chip on a heat sink [31]. An automated die-bonding system developed by Mitsubishi Electric uses global feature extraction to determine the location of the chip in an image [32]. An automated arc welding system developed by Nishida of Mitsui Shipbuilding Company used an edge detection method to find a welding position [32]. An image segmentation method called Global-Local-Edge-Coincidence using regional relationships to obtain the vertex location of an object is described by Hwang [33].

This thesis describes a new technique for vertex location using local statistics to obtain the image coordinates of a distorted grid pattern. The method consists of first performing edge detection on the image which is obtained from the digitizer. An image of the object edges and the distorted grid edges is obtained. The resulting image is then thresholded at a pre-determined gray-level to obtain a binary image. Using the small mask vertex method, the vertices of the distorted grid are extracted from the binary image. However, because the projected grid pattern is distorted, more than one vertex may be extracted from a true vertex location in a projected pattern. The method of using the matching line equation to locate the true image of a vertex is described in this thesis. This method is very effective for eliminating false vertex detections.

This thesis is divided into eight sections. Section 2 presents the study of the projection patterns which can be used to identify objects in three-dimensional space. Section 3 describes the method of edge detection to extract the object and the distorted grid pattern. In Section 4, a small mask vertex extraction method is discussed. Section 5 describes the method of estimating the correct position of the detected vertices from the small mask vertex method. Section 6 describes the experimental results of the vertex location. An example of a design to implement a three-dimensional robot vision system is discussed in Section 7. Finally, the applications of vertex location in other areas and conclusions are discussed in Section 8.

2. PROJECTION PATTERNS

A projection which substitutes for the first image of a stereo solution method is the unique feature of the active imaging technique. This technique involves controlling the image environment and image acquisition process through the projection patterns. Using this projection pattern, the time needed to extract the desired information from the image is minimized. Spatial filtering, edge enhancement, template matching, translational invariance, and very high equivalent processing rates are among the advantages of this technique [33]. However, if the resolution of the projected pattern is increased, the problem of extracting the vertex from the projected pattern may become difficult.

Several patterns and various resolutions of the grid pattern are shown in Figure 1. A single dot or point pattern may be considered the simplest pattern. The unique feature of a point is that it can be detected easily in an image since it produces a discontinuity from any direction. However, the point may appear on the background rather than on the object surface in the image which causes the disadvantage of the point pattern. This implies that a scan pattern is required for the single dot or point method in order to obtain sufficient information to distinguish the background from the object surface.

The second pattern to be considered is a line pattern. A line provides a discontinuity in only one direction. A line can be used to obtain the depth profile of the object surface [27]. However, the line has the disadvantage that only one dimension of the object surface can

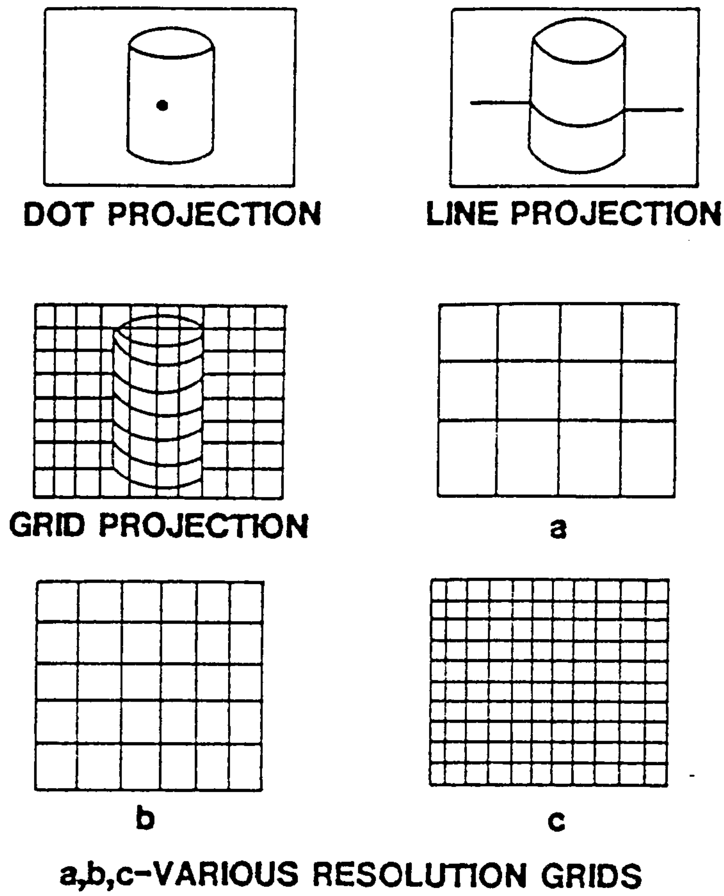
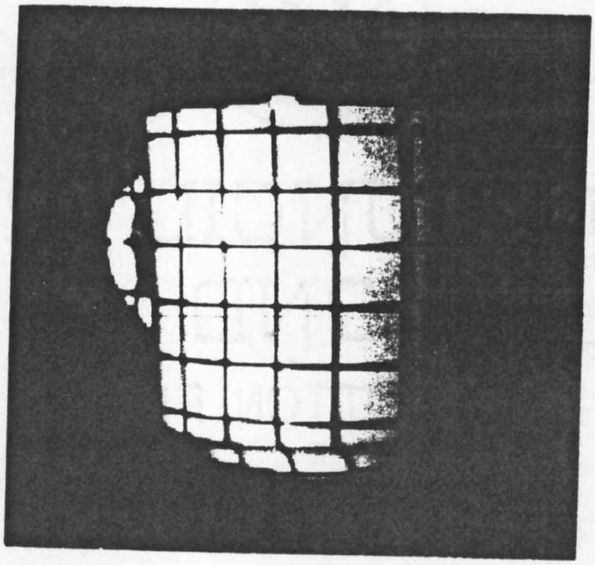


Figure 1. Various patterns for projected image.

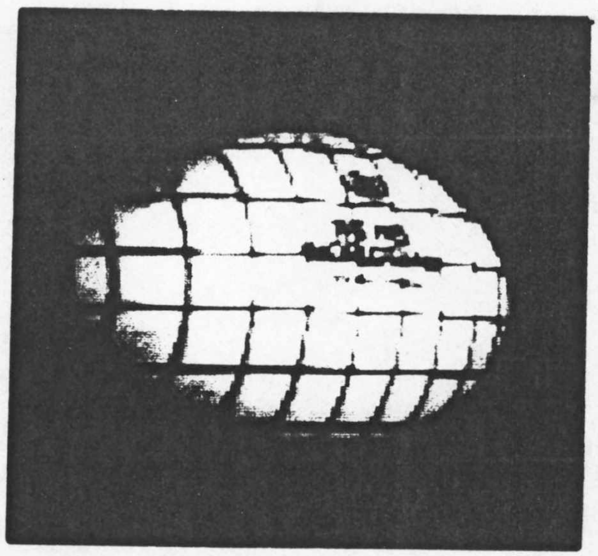
be located. A three-dimensional surface measurement can not be done without scanning the entire object surface. The next pattern, the two-dimensional grid pattern, eliminates the need for scanning.

A grid pattern which consists of a set of crossed vertical and horizontal lines has the advantages of both the point and line methods. Using a projection of the grid pattern, the intersections of the grid lines serve as identifiable points on the surface of the object. The degree to which the projected grid pattern is distorted provides the information of the orientation of the object surface. The grid pattern is focused on the 3-D surface, and the shadow of the grid pattern, which lies on the surface, is distored by the 3-D shape of the surface. Using image processing and photogrammetric techniques, a description of the pattern on the surface can be computed. Sharp discontinuities may indicate that the image either consists of more than one object or that there are some surfaces of the object which cannot be seen in the image. The determination of multiple objects or surfaces which cannot be seen in the image can be distinguished by using the grid coding method described by Will and Pennington [28]. The unique property of a grid pattern is that the entire object surface or entire scene may be covered in a single image. Another advantage of using the grid pattern is that the background and object can be easily identified. Figure 2 shows the original object images with background, object surfaces and the distorted grids.

Various resolutions of the grid pattern may be used to increase the accuracy of the measurement of the object surface. This can be accomplished by placing the object closer to the projector. Another



a. Coffee Cup Picture



b. Football Picture

Figure 2. Original 128 x 128 Digitized Images

possibility is to mount the projector on the robot arm so that the distance of the object from the projector can be controlled. The resolution of the grid pattern can be adjusted based upon the size of the object in the image. In some cases, the complexity of the object surfaces may require the use of higher resolutions to measure the object surface accurately. For robotics applications, a higher image resolution increases the number of data points required to be analyzed. For this reason, it is important that a simple and accurate method of extracting the distorted grid can be developed.

The above patterns are considered as optical projections where the light source is the element of the projection. However, there are many types of sources which can be used as projection systems. A good example is the use of an X-ray source as the projection system for the measurement of material surfaces. Another example is the use of a laser source as the projection system to measure the surface quality.

3. EDGE DETECTION

Analysis of an image usually starts at low levels, where the first step is often to detect lines and edges. Edge detection is important for both human and manchine perception. In many cases, the human observer tends to rely more upon edges and boundaries rather than individual pixel amplitudes for visual recognition and interpretation of an object [34]. A considerable amount of research has been done in attempts to extract edge information from pictures. The method of using small masks, vector operators, and polar histograms to perform edge detection is described in [35]. The outputs from these algorithms indicate the presence of line and edge elements. The reliability of these outputs depends very much on how noisy the picture is, because the noise is usually caused by the rough reflection for the object surface.

An edge can be defined as an amplitude discontinuity in some image attribute, usually brightness. In some cases these variations are due to irregularities of the surface structure of the objects within an image region. In this thesis, the term edge refers to the object edges or the distorted grid pattern.

The creation of an edge picture can be considered as a process of transforming the original gray scale image into a binary-valued image. The presence of edges is usually indicated by a higher gray level. The significance of this process is that edge detection can be used to eliminate the surface reflection of the objects and the background noise.

There are several kinds of noise in a digitized image when using the projection method. The major one is background noise. Since the characteristic of a projection system is to project a high density light source to the object surface, the camera very likely may pick up the reflected light from the background object. The second kind of the noise is the shadow created by the target object when the projector is projected at a different angle other than the one parallel to the table where the object is placed. A third kind of noise is created by the natural color, or characters appearing on the surface of the object. A fourth kind of the noise is the distance of the object from the projector or the camera. In the case of the projector, the resolution of the grid pattern decreases as the distance of the object from the projector increases. The intensity of the light source also decreases as the object is placed at a greater distance. In the case of the camera, the result of placing the camera closer to the object may cause difficulty in recognizing the vertex of the projected grid pattern. Another effect of placing the object at a distance may be reduced size of the object in the digitized image. In conclusion, it is important to make certain constraints to control the environment prior to the digitization process.

Locating abrupt intensity changes in a two-dimensional image is analogous to finding sharp transitions in a one-dimensional waveform. One of the simplest and most commonly used method of locating such transitions is by observing the value of the waveform's derivative, that is, using a gradient operator. This approach can be used for images if the differentiation is extended into two dimensions. The magnitude of the two-dimensional derivative reflects the sharpness of any change in gray level and can be used to identify an edge.

The gradient, or two-dimensional differentiation can be formulated as :

$$G\,[f(i,j)] = \begin{vmatrix} \dfrac{df}{di} \\ \hline \dfrac{df}{dj} \end{vmatrix} \quad ....(3\text{-}1)$$

It is important to note that G is a vector, which points in the direction of the maximum rate of increase of the image function f(i,j).

For a digital image, the derivatives in equation (3-1) can be approximated by differences computed in the appropriate direction. This formulation gives the gradient magnitude as

$$G\,[f(i,j)] = \sqrt{[f(i,j) - f(i+1,j)]^2 + [f(i,j) - f(i,j+1)]^2} \quad ......(3\text{-}2)$$

which is often simplified, in the interest of computation expediency, to

$$G\,[f(i,j)] = |f(i,j) - f(i+1,j)| + |f(i,j) - f(i,j+1)| \quad ......(3\text{-}3)$$

where G [f(i,j)] is the gray level in the gradient image of the pixel at location x=j, y=i.

This operator requires some calculation to be made using the gray levels of small groups of contiguous pixels. The directional relationship between pixels is shown in Figure 3. A two by two group of pixels is the smallest symmetrical group over which a discrete approximation of a two dimensional gray level gradient can be defined. The magnitude of the gradient reflects the degree of abruptness of any changes in gray level within the four pixel neighborhood of $f(i,j)$ and is a reasonable measure of the presence of an edge in that area. The result of edge detection using a gradient operator is shown in Figure 4. The next section will describe the process of vertex extraction.

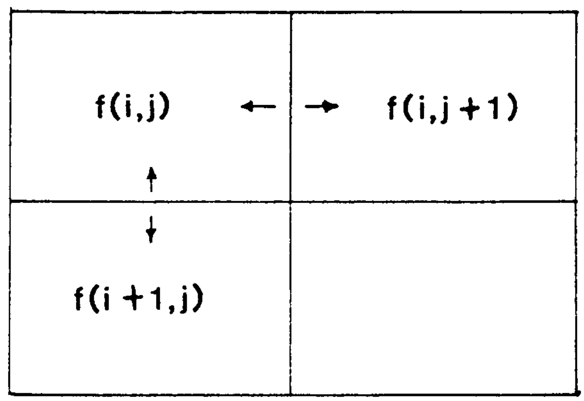
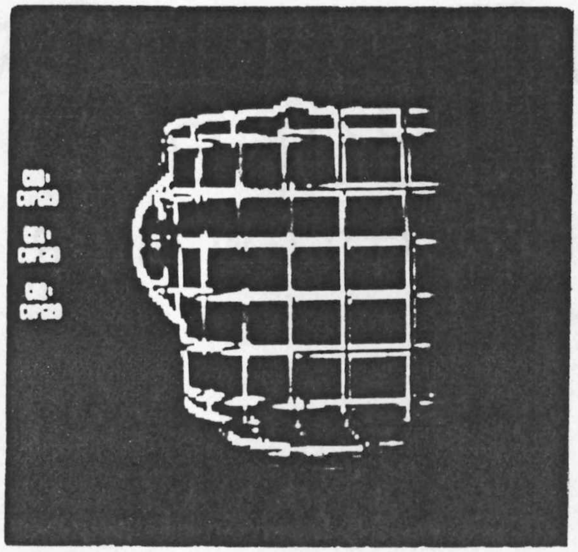
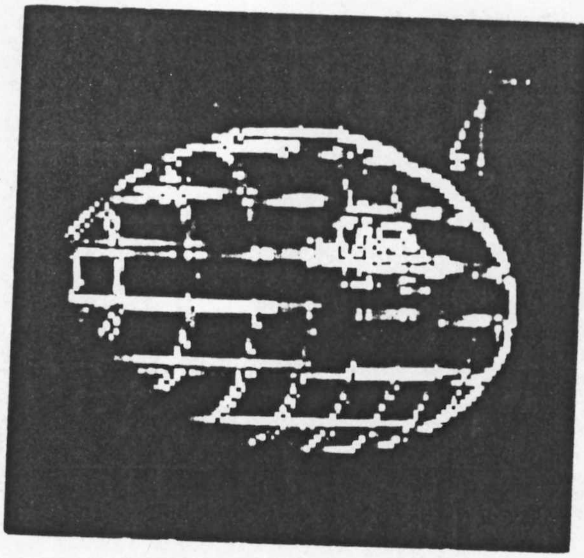


Figure 3. Directional Differences of Standard Gradient Operator



a. Coffee Cup Picture



b. Football Picture

Figure 4. Images of Standard Gradient Operator Edge Detection

4. VERTEX EXTRACTION

A vertex point is the intersection of two lines. The vertices are the features necessary to characterize the shape of a three-dimensional object. An image of the object surface is taken by a camera on black-and-white film which is digitized by a scanner. Each image is digitized into 128 by 128 pixels with gray levels ranging from 0 to 255, where 0 represents black and 255 represents white. The data of grid patterns must be extracted from a digitized image. Several general techniques for doing this have met with success in industrial applications. The simplest is windowing [36], where each region is treated separately to extract information. In order to extract the grid pattern, the grid intersection vertex points must be located before the computation for determining the object coordinates can take place.

The procedure developed for extracting the vertices from an image is described in the following process:

a) Convert the image into a binary image, using a predetermined threshold value.

b) Each small mask is read in by a skeletonization program. These masks are prepared for feature extraction by skeletonizing the image in the raster matrix, that is, reducing the small mask to one pixel width. Simultaneously, each mask is marked according to its significance: 0 - background or 1 - line point.

c) An extended region is then considered if a line point is found. The same raster matrix computation is performed to locate the "+" crossed vertices. A detailed discussion now follows.

The binary image is sufficient for the process described in this thesis. Binary imaging translates each pixel of the image into one of two colors, black or white, while gray-scale imaging allows different shades of gray to be assigned to the image points. Thresholding is a filtering technique that can be used to create a binary image from one originally containing various shade of gray. The threshold value is determined from the histogram of the image. By giving a threshold value, the gray level of the image is represented by 0 - black or 1 - white. This conversion can be represented as

$$B\ (i,j) = \begin{cases} 0 : & G(i,j) < T \\ 1 : & G(i,j) > T \end{cases} \qquad \text{................(4-1)}$$

where T is the threshold value, and G(i,j) is the gray scale at pixel location (i,j). Later in this thesis, the results of a number of different thresholds will be shown.

The process of skeletonization will be described first as it is applied to the whole raster matrix in a quasi-parallel fashion. The sequential equivalent of this process will be treated later. The region size or mask in the image is designed to be varied such as, 2 x 2, 3 x 3, or 5 x 5 pixels so that the interest size can be selected. Figure 5 shows the location of the pixels in a 2 x 2 mask, and Figure 6 shows the location of the pixels in a 3 x 3 mask. The binary pixels arrangement in a 2 x 2 mask is shown in Figure 7. In this thesis, a 3 x 3 mask size is used. For the 3 x 3 mask, the reference pixels for a point P(i,j) will be defined as follows: P(i+1,j), P(i+1,j+1),

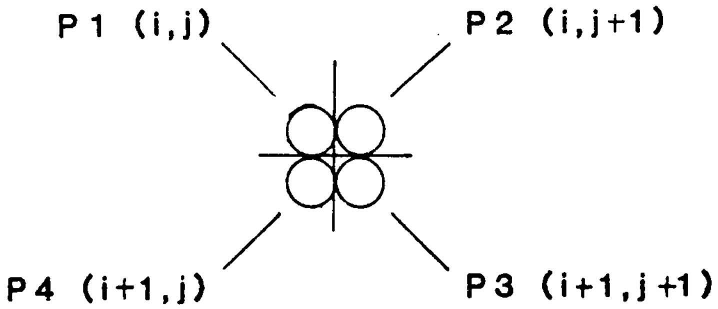


Figure 5. Location of Pixels in 2 X 2 Mask

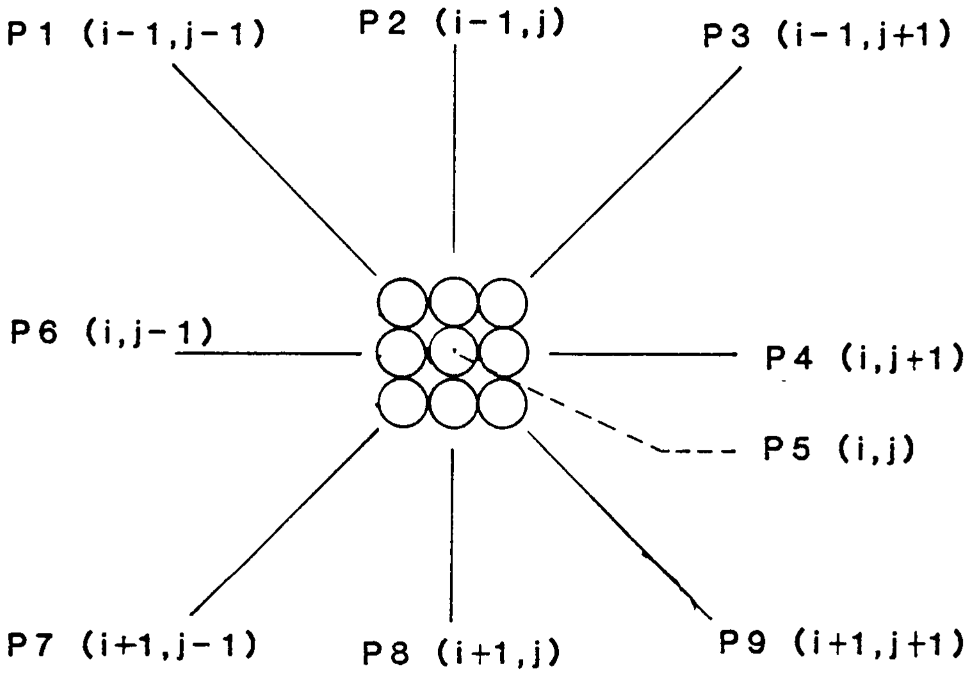


Figure 6. Location of Pixels in 3 X 3 Mask

$$\begin{array}{|c|c|} \hline x\,1 & x\,2 \\ \hline x\,1 & x\,2 \\ \hline \end{array} \quad \begin{array}{c} +\ 1 \\ +\ 1 \end{array}$$

$$E1 = P1 + P2 \times 2 + 1$$

$$E2 = P4 + P3 \times 2 + 1$$

Mask (E1, E2)

$$\begin{array}{|c|c|} \hline 0 & 0 \\ \hline 0 & 0 \\ \hline \end{array} \qquad \text{Mask } (1,1)$$

Background Mask

Figure 7. Binary Pixels Arrangement in 2 X 2 Mask

Single Pixel

0	0
1	0

Mask (2,1)

0	1
0	0

Mask (3,1)

0	0
0	1

Mask (1,3)

0	0
1	0

Mask (1,2)

Two Pixels

0	0
1	1

Mask (1,4)

0	1
0	1

Mask (3,3)

1	0
1	0

Mask (2,2)

1	1
0	0

Mask (4,1)

1	0
0	1

Mask (2,3)

0	1
1	0

Mask (3,2)

Figure 7. (Continued)

Three Pixels

1	1
0	1

Mask (4,3)

0	1
1	1

Mask (3,4)

1	0
1	1

Mask (2,4)

1	1
1	0

Mask (4,2)

Solid Mask

1	1
1	1

Mask (4,4)

Figure 7. (Continued)

P(i,j+1), P(i-1,j+1), P(i-1,j), P(i-1,j-1), P(i,j-1), and P(i+1,j-1). The center of the region, P(i,j), is moved around the image row by row from left to right. At every position, the mask value is computed as follows:

$$E1 = P1 * 4 + P2 * 2 + P3 + 1 \quad \text{..........(4-2)}$$

$$E2 = P4 * 4 + P5 * 2 + P6 + 1 \quad \text{..........(4-3)}$$

$$E3 = P7 * 4 + P8 * 2 + P9 + 1 \quad \text{..........(4-4)}$$

where the P1 through P9 is assigned as shown in Figure 6.

The P values represent the binary image value in a three by three neighborhood. The value of E1, E2 or E3 is in the range of 1 to 8 and the mask value may be considered as a three tuple (E1, E2, E3). Several examples of the mask value for a vertex, a horizontal line, a vertical line, and a diagonal line are shown in Figure 8. The mask value for a background region is (1,1,1), while a white region has a mask value of (8,8,8). An ideal "+" vertex would have a mask value of (3,8,3). In the first round of skeletonization, the mask is marked 0 if the mask value is found to be a background region (1,1,1), and is marked 1 otherwise.

For the second pass of the skeletonization process, an extended region is required and only processed by those mask value marked as 1. The extended region is designed to be either 9 x 9 or 11 x 11 region size. In this thesis, both 9 x 9 and 11 x 11 region sizes are used. To illustrate the technique, a 9 x 9 region size is

x 4	x 2	x 1	+ 1
x 1	x 2	x 4	+ 1
x 4	x 2	x 1	+ 1

$$E1 = P1 \times 4 + P2 \times 2 + P3 + 1$$

$$E2 = P4 \times 4 + P5 \times 2 + P6 + 1$$

$$E3 = P7 \times 4 + P8 \times 2 + P9 + 1$$

Mask (E1, E2, E3)

Vertex Mask

0	1	0
1	1	1
0	1	0

Mask (3,8,3)

Figure 8. Binary Pixels Arrangement in 3 X 3 Mask

Background Mask

0	0	0
0	0	0
0	0	0

Mask (1,1,1,)

Solid Mask

1	1	1
1	1	1
1	1	1

Mask(8,8,8)

Figure 8. (Continued)

Horizontal Line Mask

1	1	1
1	1	1
0	0	0

Mask (8,8,1)

0	0	0
1	1	1
1	1	1

Mask (1,8,8)

1	1	1
0	0	0
0	0	0

Mask (8,1,1)

0	0	0
0	0	0
1	1	1

Mask (1,1,8)

0	0	0
1	1	1
0	0	0

Mask (1,8,1)

Figure 8. (Continued)

Vertical Line Mask

1	1	0
1	1	0
1	1	0

Mask (7,4,7)

0	1	1
0	1	1
0	1	1

Mask (4,7,4)

1	0	0
1	0	0
1	0	0

Mask (5,2,5)

0	0	1
0	0	1
0	0	1

Mask (2,5,2)

0	1	0
0	1	0
0	1	0

Mask (3,3,3)

Figure 8. (Continued)

Right Diagonal Line

0	1	1
1	1	0
1	0	0

Mask (4,4,5)

0	0	1
0	1	1
1	1	0

Mask (2,7,7)

0	1	0
1	0	0
0	0	0

Mask (3,2,1)

0	0	0
0	0	1
0	1	0

Mask (1,5,3)

0	0	1
0	1	0
1	0	0

Mask (2,3,5)

Figure 8. (Continued)

Left Diagonal Line

1	1	0
0	1	1
0	0	1

Mask (7,7,2)

1	0	0
1	1	0
0	1	1

Mask (5,4,4)

0	1	0
0	0	1
0	0	0

Mask (3,5,1)

0	0	0
1	0	0
0	1	0

Mask (1,2,3)

1	0	0
0	1	0
0	0	1

Mask (5,3,2)

Figure 8. (Continued)

considered. As Figure 9 shows, the 9 x 9 mask has been assigned to nine 3 x 3 small masks, represented by M1 through M9. Using the same calculation, the mask value can be computed by following equations:

$$G1 = M1 * 4 + M2 * 2 + M3 + 1 \quad \text{...........(4-5)}$$

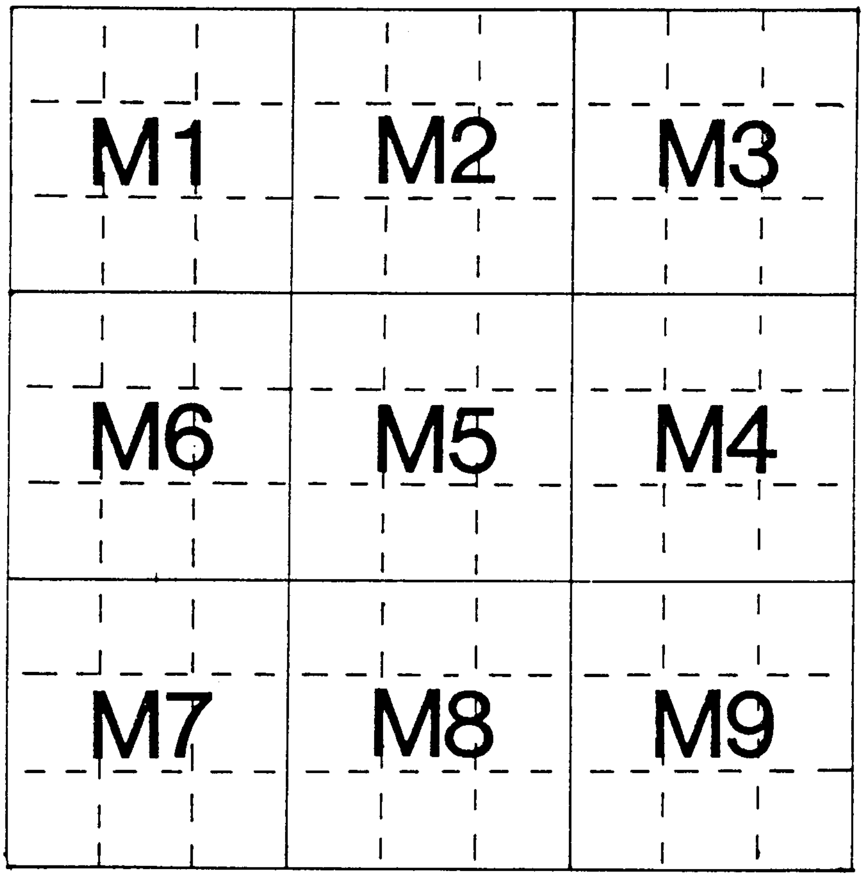
$$G2 = M4 * 4 + M5 * 2 + M6 + 1 \quad \text{...........(4-6)}$$

$$G3 = M7 * 4 + M8 * 2 + M9 + 1 \quad \text{...........(4-7)}$$

where the M values were obtained from the previous computation, that is 0 represented a background and 1 represented a line point.

If the mask value of (G1, G2, G3) is (3,8,3), the expected vertex point is detected. For illustration, Figure 9 shows the different types of arrangements of pixels that exist in a 9 x 9 region. The next section will discuss the technique of using the matching line concept to reduce the false location of vertex points.

Mask (G1, G2, G3)



$$G1 = M1 \times 4 + M2 \times 2 + M3 + 1$$

$$G2 = M4 \times 4 + M5 \times 2 + M6 + 1$$

$$G3 = M7 \times 4 + M8 \times 2 + M9 + 1$$

Figure 9. Pixels Arrangement in a 9 x 9 Mask

0	0	0	0	1	0	0	0	0
0	0	0	1	1	0	0	0	0
0	0	0	0	1	1	0	0	0
0	0	0	0	1	1	1	1	0
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0	1
0	0	0	1	1	1	0	0	0
0	0	0	1	1	0	0	0	0
0	0	0	1	1	0	0	0	0

Example of Confident Vertex

Figure 9. (Continued)

0	0	1	1	0	0	0	0	0
1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	0
0	0	1	1	1	0	0	1	0
0	0	1	1	1	0	0	1	0
0	1	1	1	0	1	1	0	0
0	1	1	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Example of Undetermined Vertex

Figure 9. (Continued)

5. VERTEX ESTIMATION

The vertices of the grid pattern provide information for the surface measurement of an object. Thus, an unambiguity vertex location must be obtained. There are few vertices, located next to the true vertex points and have been detected by using the small mask technique, need to be eliminated. One method to find the true vertex is by using matching line concepts and calculating the minimum distance. An excellent paper describing the method for three-dimensional scene matching is given by Hwang [37]. Figure 10 shows that a point in one image defines a ray along which the corresponding point must lie. In this thesis, the optical axis of the projector is parallel to the optical axis of the camera. Thus, searching for the corresponding vertex of the projected grid pattern can be achieved along a row in the image plane. The lines on which the corresponding points lie are termed matching lines. The equation for a matching line is defined as the matching line equation. The geometrical relationship of the corresponding points must follow a bilinear relation. The bilinear relationship may be represented by [33]

$$f = x_p x_c C_1 + x_p y_p C_2 + x_p y_c C_3 + x_c y_p C_4$$
$$+ x_p C_5 + x_c C_6 + y_p C_7 + y_c C_8 + C_9 = 0 \qquad (5\text{-}1)$$

where C_i, i=1 to 9 are the coefficients which depend on the relative

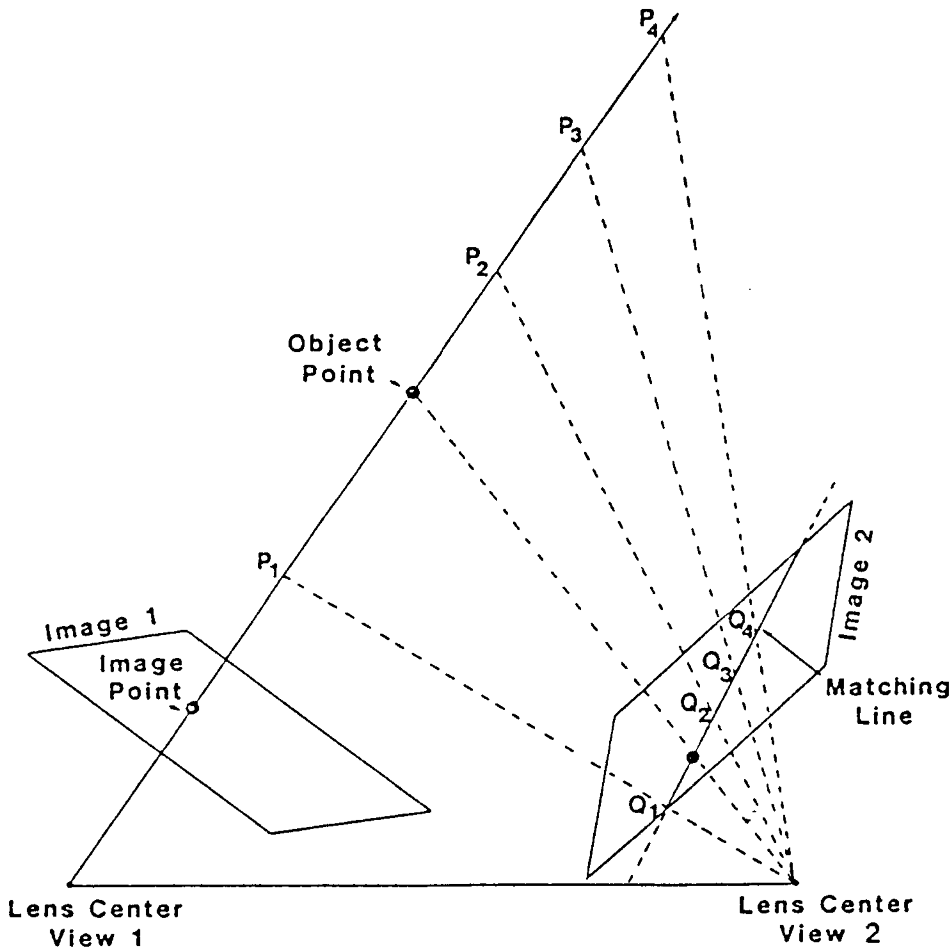


Figure 10. Matching Line Concept

geometry and focal length of the projector and the camera. (x_p, y_p) and (x_c, y_c) are the image points of the projector and camera.

The value of the coefficient C_i's can be computed if the perspective projection transformation matrix of the projector and camera are known. The derivative of the perspective projection transformation matrix of the projector and the camera is described in [38]. The imaging process may be represented by a perspective projection transformation process in homogeneous space.

$$\underline{U}^* = \underline{A}\,\underline{V}^* \qquad (5\text{-}2)$$

where $\underline{U}^*$ is a 4-vector homogeneous representation of the image at the $Z_i = 0$ plane,

$\underline{A}$ is the perspective projection transformation matrix,

$\underline{V}^*$ is the three-dimensional cartesian coordinates with w = 1 as shown in Figure 11.

or

$$\begin{bmatrix} W x_i \\ W y_i \\ W \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \qquad (5\text{-}3)$$

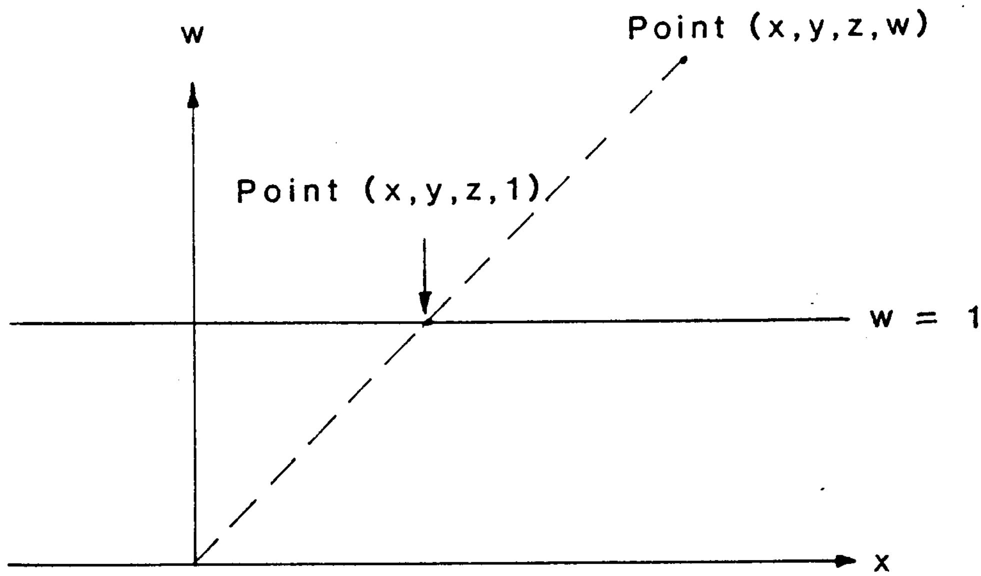


Figure 11. A Homogeneous Coordinate Point

where W is the non-zero factor for homogeneous representation;

x_i, y_i are the image coordinates ;

$a_{11} = \cos\psi \cos\theta$;

$a_{12} = -\sin\psi \cos\phi - \sin\phi \sin\phi \cos\psi$;

$a_{13} = -\sin\psi \sin\phi + \cos\phi \cos\psi \sin\theta$;

$a_{14} = -Nx\,(a_{11}) - Ny\,(a_{12}) - Nz\,(a_{13})$;

$a_{21} = -\sin\psi \cos\theta$;

$a_{22} = -\cos\psi \cos\phi + \sin\phi \sin\psi \sin\theta$;

$a_{23} = -\cos\psi \sin\phi - \cos\phi \sin\psi \sin\theta$;

$a_{24} = -Nx\,(a_{21}) - Ny\,(a_{22}) - Nz\,(a_{23})$;

$a_{31} = \sin\theta / f$;

$a_{32} = \cos\theta \sin\phi / f$;

$a_{33} = -\cos\phi \cos\theta / f$;

$a_{34} = -Nx(\sin\theta) - Ny(\cos\theta \sin\phi) - Nz(-\cos\phi \cos\theta))/f + 1$; and

X, Y, Z are the three-dimensional cartesian coordinates.

As discussed in [37], matrix $\underline{A}$ is a singular matrix, a nonsingular 4 x 4 matrix may be obtained by augmenting the matrix $\underline{A}$ by introducing a row vector of (0, 0, 0, 1). The resulting matrix representation as shown in Figure 12 may be defined as

$$\begin{bmatrix} r & r & r & p \\ r & r & r & p \\ r & r & r & p \\ t & t & t & s \end{bmatrix}$$

r - Rotation and Scaling

p - Perspective

t - Translation

s - W component

Figure 12. Elements of Perspective Projection Transformation Matrix

$$\begin{bmatrix} W x_i \\ W y_i \\ W \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad \ldots\ldots\ldots\ldots\ldots(5\text{-}4)$$

For the projection system, the perspective projection transformation matrices of projector and camera are represented by

$$\begin{bmatrix} W_1 x_p \\ W_1 y_p \\ W_1 \\ 1 \end{bmatrix} = \begin{bmatrix} a_{111} & a_{112} & a_{113} & a_{114} \\ a_{121} & a_{122} & a_{123} & a_{124} \\ a_{131} & a_{132} & a_{133} & a_{134} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad \ldots\ldots\ldots\ldots\ldots(5\text{-}5)$$

and

$$\begin{bmatrix} W_2 x_c \\ W_2 y_c \\ W_2 \\ 1 \end{bmatrix} = \begin{bmatrix} a_{211} & a_{212} & a_{213} & a_{214} \\ a_{221} & a_{222} & a_{223} & a_{224} \\ a_{231} & a_{232} & a_{233} & a_{234} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad \ldots\ldots\ldots\ldots\ldots(5\text{-}6)$$

where x_p, y_p are the image coordinates of projector;

x_p, y_p are the image coordinates of camera;

W_1, W_2 are the non-zero factors of the homogeneous representation for three-dimensional space at the $Z_p = Z_c = 0$.

Similarly, equation (5-5) and equation (5-6) can be rewritten as

$$\underline{U}_p^{**} = \underline{A}_p \, \underline{V}^{*} \qquad \text{...........(5-7)}$$

and

$$\underline{U}_c^{**} = \underline{A}_c \, \underline{V}^{*} \qquad \text{...........(5-8)}$$

where $\underline{U}_p^{**}$ is a 4-vector super homogeneous representation of the projector image coordinates,

$\underline{U}_c^{**}$ is a 4-vector super homogeneous representation of the camera image coordinates,

$\underline{A}_p$ is the perspective projection transformation matrix of the projector,

$\underline{A}_c$ is the perspective projection transformation matrix of the camera,

$\underline{V}^{*}$ is the three-dimensional homogeneous representation matrix.

Since matrix $\underline{A}_p$ is nonsingular, the object coordinates and the projector image coordinates may be expressed by

$$\underline{V}^{*} = \underline{A}_p^{-1} \underline{U}_p^{**} \qquad \text{........(5-9)}$$

Substitute equation (5-9) into equation (5-8)

$$\underline{U}_c^{**} = \underline{A}_c \underline{A}_p^{-1} \underline{U}_p^{**} \qquad \text{........(5-10)}$$

Equation (5-10) can be rewritten as

$$\underline{U}_c^{**} = \underline{M} \; \underline{U}_p^{**} \qquad \text{........(5-11)}$$

and

$$\underline{M} = \underline{A}_c \underline{A}_p^{-1} \qquad \text{........(5-12)}$$

where $\underline{M}$ is the matched images transformation matrix. Rewriting equation (5-11) with the projector and camera image coordinates gives

$$\begin{bmatrix} W_2 x_c \\ W_2 y_c \\ W_2 \\ 1 \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} W_1 x_p \\ W_1 y_p \\ W_1 \\ 1 \end{bmatrix} \qquad \text{....(5-13)}$$

Equation (5-13) may be rewritten into equation (5-14)

$$\begin{bmatrix} x_c \\ y_c \\ 1 \\ 1/W_1 \end{bmatrix} = k \begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_p \\ y_p \\ 1 \\ 1/W_2 \end{bmatrix} \quad \text{....(5-14)}$$

where

$$k = \frac{W_2}{W_1}$$

and W_1 and W_2 can be represented as

$$W_1 = \frac{M_{34}}{1 - kM_{31}x_p - kM_{32}y_p - kM_{33}} \quad \text{.........(5-15)}$$

$$W'_2 = \frac{k\ M_{34}}{1 - kM_{31}x_p - kM_{32}y_p - kM_{33}} \quad \text{.........(5-16)}$$

Using equations (5-15) and (5-16), then

$$x_c = \frac{k}{M_{34}} [(M_{34}M_{11} - M_{14}M_{31}) x_p + (M_{34}M_{12} - M_{14}M_{32}) y_p$$

$$+ (M_{34}M_{13} - M_{14}M_{33})] + \frac{M_{14}}{M_{34}} \quad \text{..........(5-17)}$$

$$y_c = \frac{k}{M_{34}} [(M_{34} M_{21} - M_{24} M_{31}) x_p + (M_{34} M_{22} - M_{24} M_{32}) y_p$$

$$+ (M_{34} M_{23} - M_{24} M_{33})] + \frac{M_{24}}{M_{34}} \quad \ldots\ldots\ldots\ldots(5\text{-}18)$$

Using Equations (5-17) and (5-18) to eliminate k, then the following equations may be derived :

$$(d_1 x_p + d_2 y_p + d_3) x_c + (d_4 x_p + d_5 y_p + d_6) y_c$$

$$+ (d_7 x_p + d_8 y_p + d_9) = 0 , \quad \ldots\ldots\ldots\ldots(5\text{-}19)$$

or

$$(d_1 x_c + d_4 y_c + d_7) x_p + (d_2 x_c + d_5 y_c + d_8) y_p$$

$$+ (d_3 x_c + d_6 y_c + d_9) = 0 , \quad \ldots\ldots\ldots\ldots(5\text{-}20)$$

where

$d_1 = M_{34} M_{21} - M_{24} M_{31}$;

$d_2 = M_{34} M_{22} - M_{24} M_{32}$;

$d_3 = M_{34} M_{23} - M_{24} M_{33}$;

$d_4 = M_{14} M_{31} - M_{34} M_{11}$;

$d_5 = M_{14} M_{32} - M_{34} M_{12}$;

$d_6 = M_{14} M_{33} - M_{34} M_{13}$;

$d_7 = M_{24}M_{11} - M_{21}M_{14}$;

$d_8 = M_{24}M_{12} - M_{22}M_{14}$;

$d_9 = M_{24}M_{13} - M_{23}M_{14}$;

Equation (5-19) is defined as the matching line equation which can be simplified into

$$a\,x + b\,y + c = 0 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots(5\text{-}21)$$

where a, b, c are the coefficients of the matching line.

For each vertex in the projected pattern, the true image vertex must lie on a matching line. Since the equation of the matching line is known, the distance from each possible vertex to this line may be used as a figure of merit for the vertex. The equation for this distance is:

$$d = (a\,x + b\,y + c) \,/\, (a^2 + b^2)^{0.5} \quad \ldots\ldots\ldots(5\text{-}22)$$

where a, b, c are the coefficients of the line and (x, y) is the coordinate of the possible vertex. Since a distance exactly equal to zero is unlikely because of quantization, a minimum distance is used to selected a uniqe vertex.

6. EXPERIMENTAL RESULTS

In this study, coffee cup and football pictures were used as examples to demonstrate the technique of vertex location using the projection method. A conventional slide projector was used to represent the projection system. The location of the camera and the slide projector was arranged as shown in Figure 13. The angle between the location of the camera and the projector is selected arbitrarily provided that the entire grid can be seen from the camera. Other than the slide projector and the camera, the experiment also includes an image processing system which consists of a PDP 11/40 computer, a DE ANZA image display and other computer accessories such as a video camera, joystick, etc. Prior to the digitization, it is important that the distance of the object from the projector and camera is within a reasonable range. As discussed in the section 3, the distance of the object from the projector or camera may affect the quality of the object image for vertex location. One solution is to use a laser or other range finder mounted near the camera so that the computer can adjust the focal length of the camera. For better results, another laser range finder can be mounted near the projector to control the focal length, the light source intensity and the resolution of the projection pattern. In this thesis, all experiments are performed based upon the following conditions: the projector and camera are placed parallel with the plane where the object is placed, the distance of the projector is within one meter such that the grid pattern is visible to the camera. The distance of the camera is also within one meter.

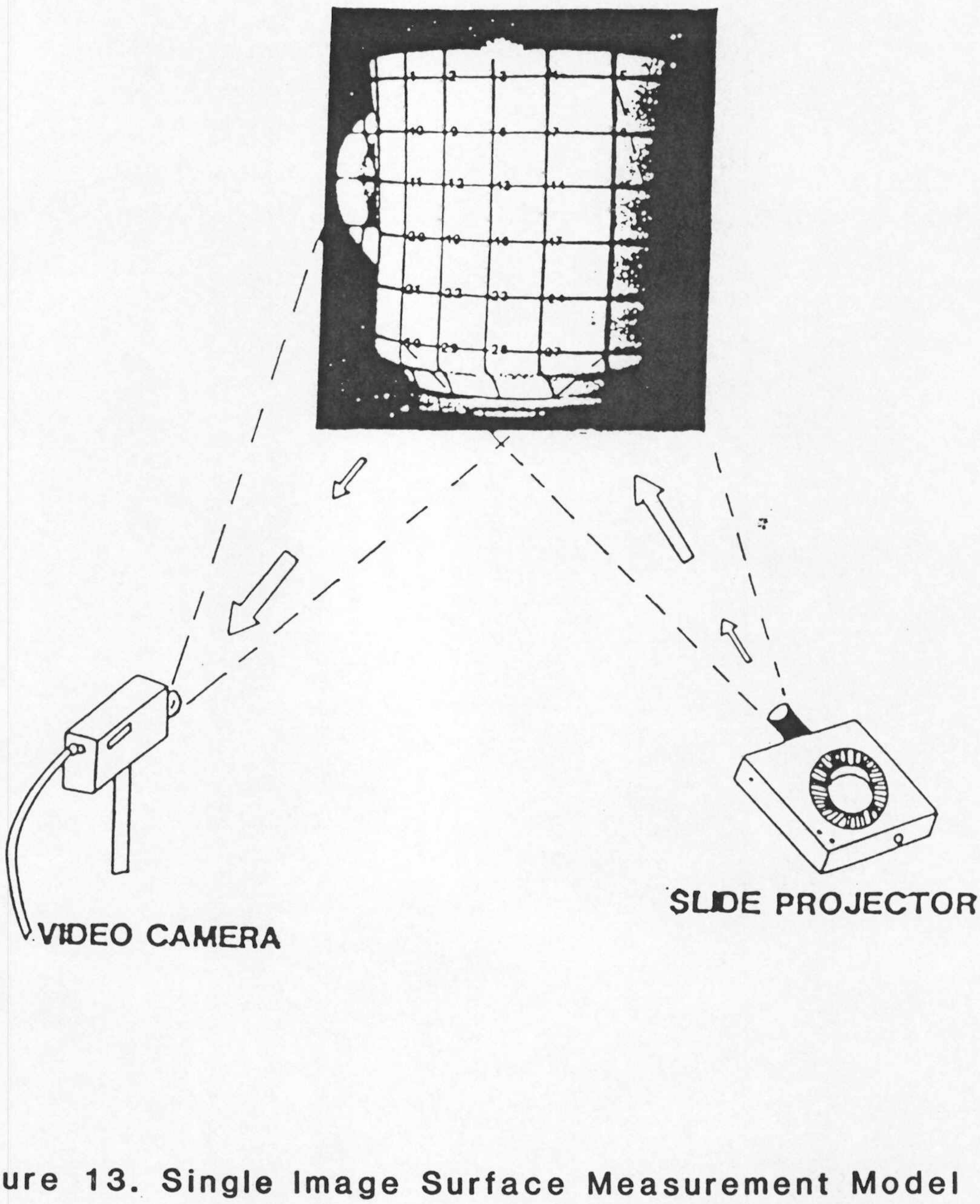
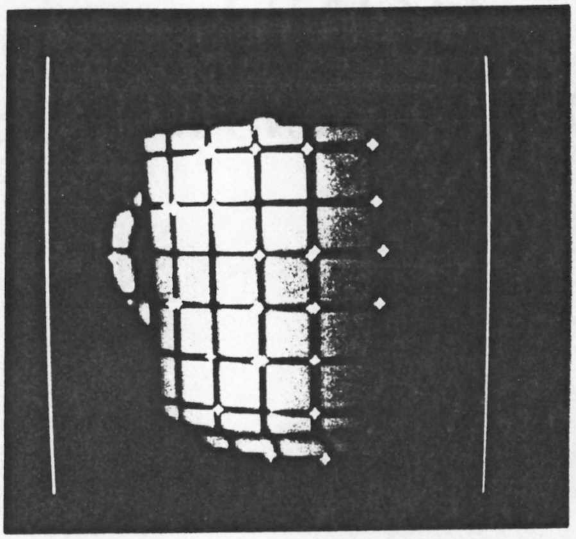


Figure 13. Single Image Surface Measurement Model

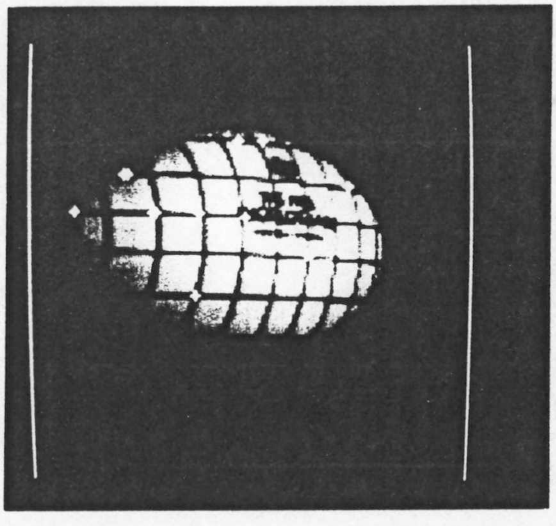
The image of the object with the distorted grid was digitized through the video camera with a resolution of 128 x 128 pixels. The gray level of the digitized image is from 0 to 255 levels as black to white. Figure 2, page 11 shows the original digitized object images which were taken from the display screen. Figure 4, page 18 shows the resulting images of edge detection with a standard gradient operation. Using the standard gradient operator, the distorted grid pattern and the edge of the object are extracted from the original image. The significant change from the original image is that the edge detection process has eliminated the reflection of the projected light source from the object surface.

The detected vertex points using the small mask method are shown in Figure 14 to Figure 21. Figure 14 shows the pictures with a threshold value at 10.0 in a 9 x 9 mask. Figure 15 shows the pictures with threshold value at 20.0 in a 9 x 9 mask. Figure 16 shows the pictures with threshold value at 30.0 in a 9 x 9 mask. Figure 17 shows the pictures with threshold value at 40.0 in a 9 x 9 mask. For the 11 x 11 mask, Figure 18 shows the pictures with threshold value at 10.0, Figure 19 shows the pictures with threshold value at 20.0, Figure 20 shows the pictures with threshold value at 30.0, and Figure 21 shows the pictures with threshold value at 40.0. As the pictures show, a threshold value at 20.0 seems to present a better result. The accuracy of detected vertices is about 70%.

A list of the original grid image and object image pixel coordinates of a cup are shown in Tables 1 and 2. Table 3 shows the detected vertex coordinates of a cup with a threshold value at 20.0.



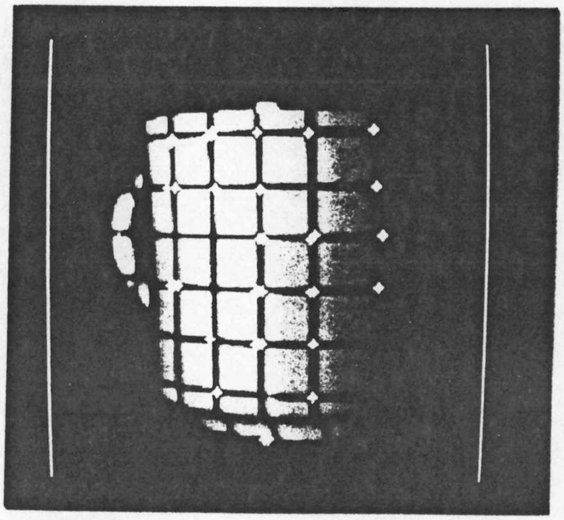
a. Coffee Cup Picture



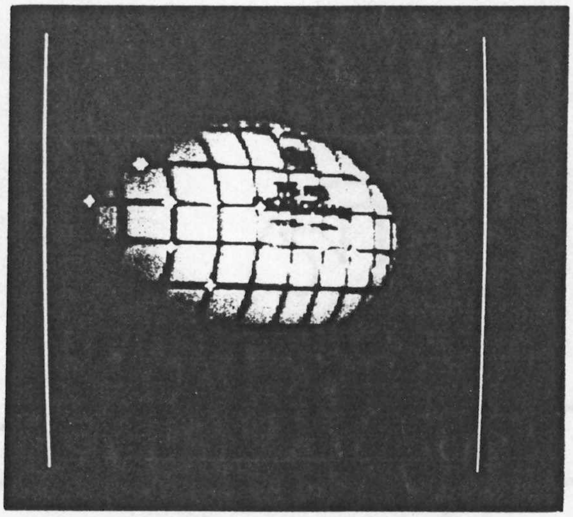
b. Football Picture

Figure 14. Detected Vertices with

Threshold Value : 10.0 in 9 x 9 Mask



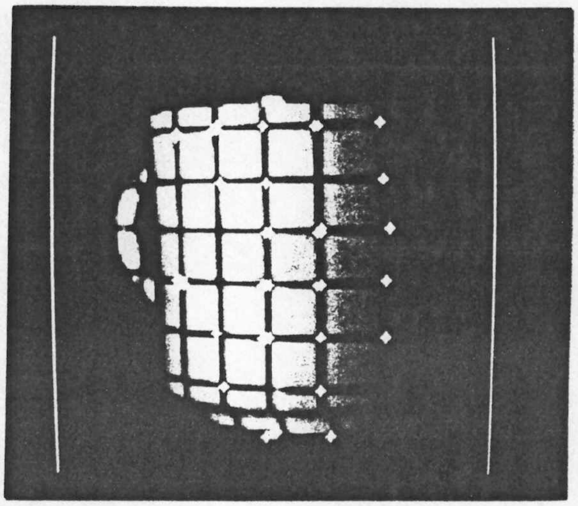
a. Coffee Cup Picture



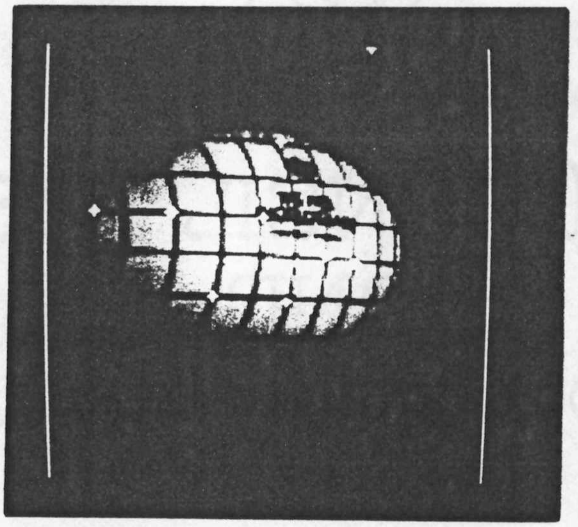
b. Football Picture

Figure 15. Detected Vertices with

Threshold Value : 20.0 in 9 x 9 Mask



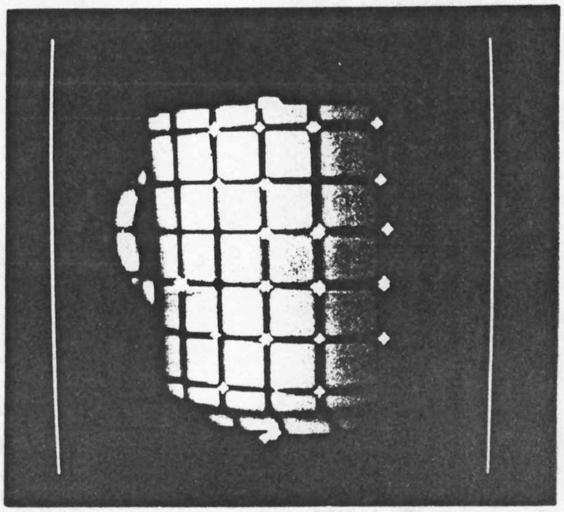
a. Coffee Cup Picture



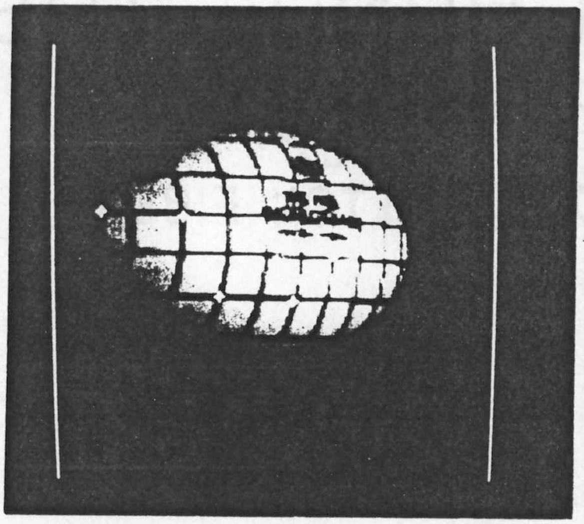
b. Football Picture

Figure 16. Detected Vertices with

Threshold Value : 30.0 in 9 x 9 Mask



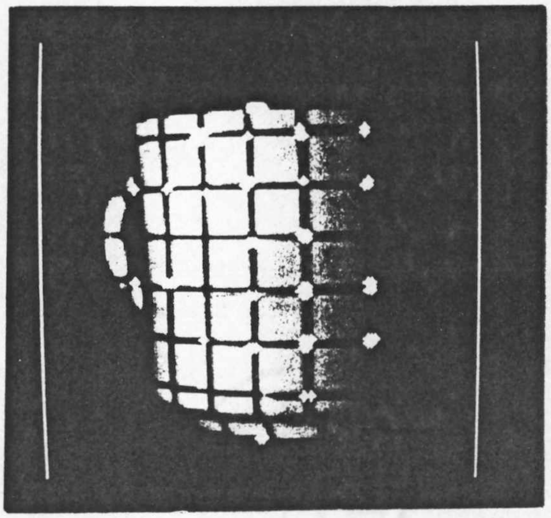
a. Coffee Cup Picture



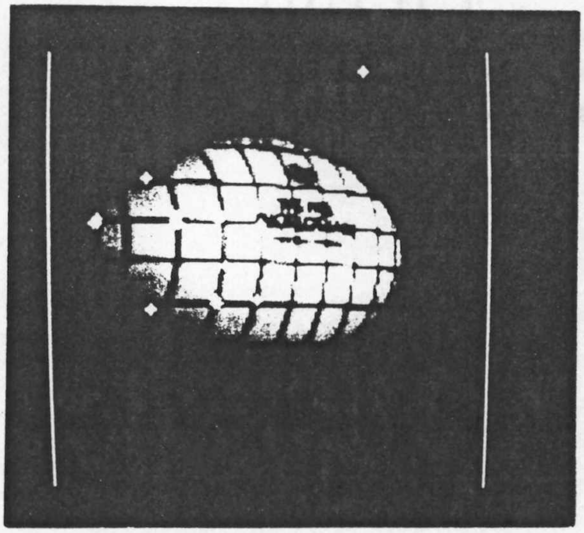
b. Football Picture

Figure 17. Detected Vertices with

Threshold Value : 40.0 in 9 x 9 Mask

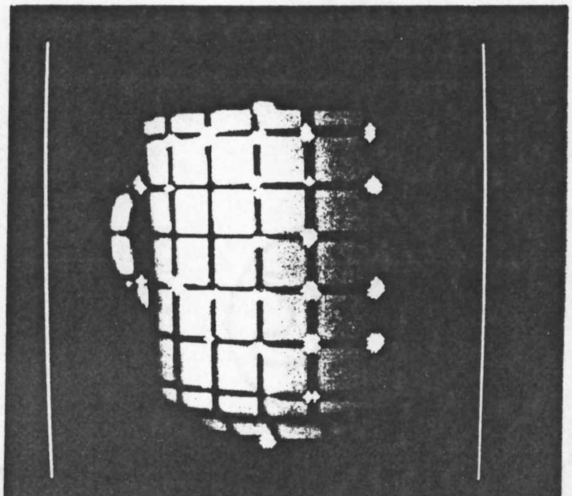


a. Coffee Cup Picture

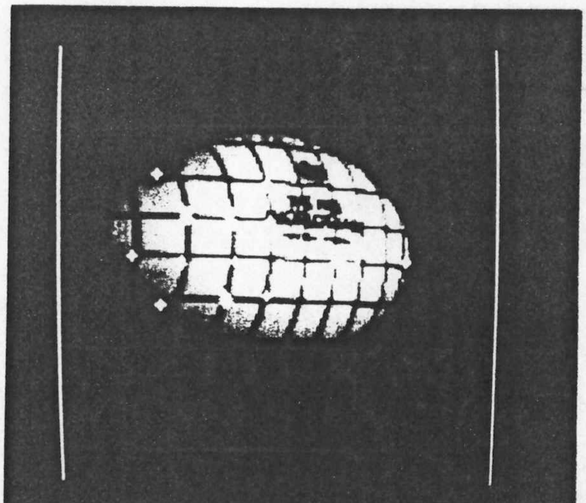


b. Football Picture

Figure 18. Detected Vertices with Threshold Value : 10.0 in 11 x 11 Mask



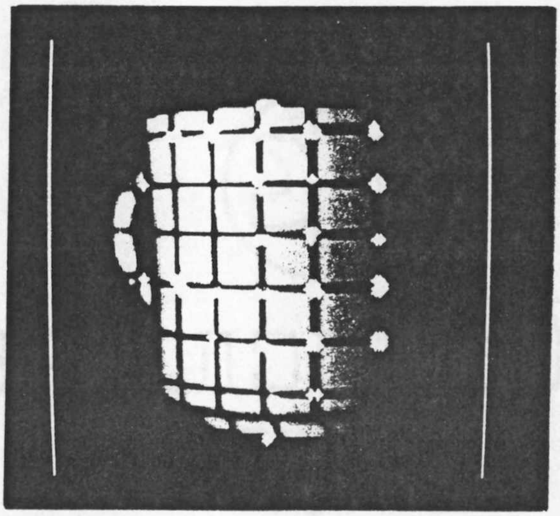
a. Coffee Cup Picture



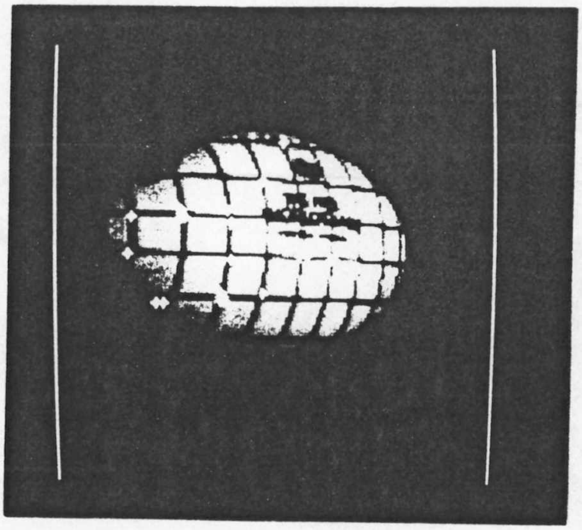
b. Football Picture

Figure 19. Detected Vertices with

Threshold Value : 20.0 in 11 x 11 Mask

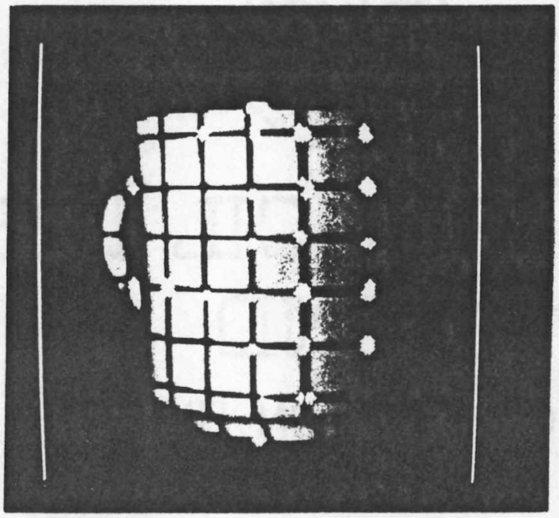


a. Coffee Cup Picture

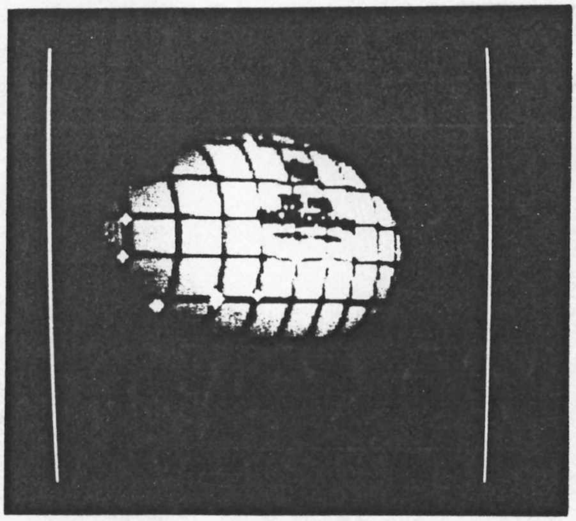


b. Football Picture

Figure 20. Detected Vertices with Threshold Value : 30.0 in 11 x 11 Mask



a. Coffee Cup Picture



b. Football Picture

Figure 21. Detected Vertices with

Threshold Value : 40.0 in 11 x 11 Mask

TABLE 1

ORIGINAL GRID VERTEX COORDINATES

Number	X_p	Y_p
1	46.00	51.00
2	52.00	51.00
3	58.00	51.00
4	64.00	51.00
5	70.00	51.00
6	70.00	57.00
7	64.00	57.00
8	58.00	57.00
9	52.00	57.00
10	46.00	57.00
11	46.00	63.00
12	52.00	63.00
13	58.00	63.00
14	64.00	63.00
15	70.00	63.00
16	70.00	69.00
17	64.00	69.00
18	58.00	69.00
19	52.00	69.00
20	46.00	69.00
21	46.00	75.00
22	52.00	75.00
23	58.00	75.00
24	64.00	75.00
25	70.00	75.00
26	70.00	81.00
27	64.00	81.00
28	58.00	81.00
29	52.00	81.00
30	46.00	81.00

TABLE 2

ORIGINAL IMAGE VERTEX COORDINATES FOR THE CUP

Number	X_p	Y_p
1	46.00	49.00
2	52.00	49.00
3	58.00	49.00
4	65.00	49.00
5	73.00	49.00
6	73.00	55.00
7	65.00	55.00
8	58.00	55.00
9	52.00	55.00
10	45.00	55.00
11	45.00	62.00
12	52.00	62.00
13	58.00	62.00
14	65.00	62.00
15	73.00	62.00
16	73.00	69.00
17	65.00	69.00
18	58.00	69.00
19	52.00	69.00
20	47.00	69.00
21	47.00	75.00
22	53.00	75.00
23	59.00	75.00
24	66.00	75.00
25	74.00	75.00
26	74.00	82.00
27	66.00	82.00
28	59.00	82.00
29	53.00	82.00
30	47.00	82.00

TABLE 3

DETECTED VERTEX COORDINATES FOR THE CUP

Number	X	Y
1	46.00	49.00
2	52.00	49.00
3	58.00	49.00
4	65.00	49.00
5	73.00	49.00
6	73.00	55.00
7	58.00	55.00
8	52.00	55.00
9	45.00	55.00
10	58.00	62.00
11	65.00	62.00
12	73.00	62.00
13	73.00	69.00
14	65.00	69.00
15	58.00	69.00
16	47.00	69.00
17	53.00	75.00
18	59.00	75.00
19	66.00	75.00
20	66.00	82.00
21	53.00	82.00

The original image vertex coordinates of the football are shown in Table 4. The detected vertex coordinates of the football with a threshold value of 20.0 are shown in Table 5. The perspective, projective transformation matrix for the projector and camera of a cup are shown in Table 6 and Table 7. The perspective, projective transformation matrix for the projector and camera of the football are shown in Table 8 and Table 9. The results of computing the matching line coefficients and minimum distance for both cup and football are shown in Table 10 and Table 11 respectively. The confusion matrices for vertex location of both cup and football are shown in Table 12 and Table 13. Finally, the summary of percent agreement of detected vertex points is shown in Table 14.

As the results show, not all of the vertices on the object surface are detected. One reason is due to the selection of the threshold value when converting the edge detection image to binary image. Another reason is because of using low resolution images, such as a 128 x 128 picture. However, it should be noted that for surface fitting, not all vertices are necessary since a least square fit would normally be used. The results indicate that this technique demonstrates the capability of locating vertices for projection method.

TABLE 4

ORIGINAL IMAGE VERTEX COORDINATES FOR THE FOOTBALL

Number	X_p	Y_p
1	28.00	49.00
2	38.00	49.00
3	48.00	49.00
4	58.00	49.00
5	68.00	49.00
6	78.00	49.00
7	88.00	49.00
8	88.00	59.00
9	78.00	59.00
10	68.00	59.00
11	58.00	59.00
12	48.00	59.00
13	38.00	59.00
14	28.00	59.00
15	28.00	69.00
16	38.00	69.00
17	48.00	69.00
18	58.00	69.00
19	68.00	69.00
20	78.00	69.00
21	88.00	69.00
22	88.00	79.00
23	78.00	79.00
24	68.00	79.00
25	58.00	79.00
26	48.00	79.00
27	38.00	79.00
28	28.00	79.00

TABLE 5

DETECTED VERTEX COORDINATES FOR THE FOOTBALL

Number	X	Y
1	28.00	49.00
2	38.00	49.00
3	48.00	49.00
4	58.00	49.00
5	58.00	59.00
6	48.00	59.00
7	38.00	59.00
8	28.00	59.00
9	38.00	69.00
10	48.00	69.00
11	78.00	69.00
12	78.00	79.00
13	48.00	79.00
14	28.00	79.00

TABLE 6

PERSPECTIVE, PROJECTIVE TRANSFORMATION MATRIX
FOR THE PROJECTOR OF A CUP

$$\begin{bmatrix} 0.153369 & 0.000015 & 0.011294 & -0.891651 \\ 0.003964 & 0.143151 & 0.000397 & -0.035050 \\ 0.002378 & -0.005735 & -0.003973 & 1.000000 \end{bmatrix}$$

TABLE 7

PERSPECTIVE, PROJECTIVE TRANSFORMATION MATRIX
FOR THE CAMERA OF A CUP

$$\begin{bmatrix} 0.035459 & -0.001301 & 0.005646 & -0.439592 \\ -0.000910 & 0.035949 & 0.000960 & -0.075792 \\ -0.004524 & -0.001974 & -0.010727 & 1.000000 \end{bmatrix}$$

TABLE 8

PERSPECTIVE, PROJECTIVE TRANSFORMATION MATRIX
FOR THE PROJECTOR OF A FOOTBALL

$$\begin{bmatrix} 0.297924 & 0.007178 & 0.041483 & -1.822266 \\ -0.014936 & 0.298462 & -0.001833 & 0.122259 \\ -0.012367 & 0.018968 & -0.009337 & 1.000000 \end{bmatrix}$$

TABLE 9

PERSPECTIVE, PROJECTIVE TRANSFORMATION MATRIX
FOR THE CAMERA OF A FOOTBALL

$$\begin{bmatrix} -0.889162 & 0.002621 & 0.000054 & -0.008949 \\ 0.001285 & -0.885912 & -0.001109 & 0.048116 \\ -0.003775 & 0.001914 & -0.060910 & 1.000000 \end{bmatrix}$$

TABLE 10

MATCHING LINE AND MINIMUM DISTANCE COMPUTATION OF A CUP

#	Image Pts. at Image 1		Image Pts. at Image 2		Matching Line Coefficients			Minimum Distance
	X	Y	X	Y	a	b	c	d
1	-0.06	1.00	-0.06	1.00	0.260223	0.060365	0.090868	0.018
2	-0.30	1.00	-0.30	1.00	0.255442	-0.005825	0.082282	0.001
3	0.00	1.00	0.00	1.00	0.250662	-0.072016	0.250662	0.006
4	0.30	1.00	0.35	1.00	0.245881	-0.138106	0.065110	0.046
5	0.60	1.00	0.75	1.00	0.241100	-0.204396	0.056524	0.104
6	0.60	0.70	0.75	0.70	0.165900	-0.207495	0.031789	0.041
7	0.30	0.70	0.35	0.70	0.170681	-0.141304	0.040375	0.005
8	0.00	0.70	0.00	0.70	0.175462	-0.075114	0.048961	0.019
9	-0.30	0.70	-0.30	0.70	0.180242	-0.008924	0.057547	0.015
10	-0.60	0.70	-0.65	0.70	0.185023	0.057266	0.066113	0.073
11	-0.60	0.35	-0.65	0.35	0.109823	0.054167	0.041398	0.090
12	-0.30	0.35	-0.30	0.35	0.105042	-0.012023	0.032812	0.028
13	0.00	0.35	0.00	0.35	0.100262	-0.078213	0.024226	0.025
14	0.30	0.35	0.35	0.35	0.095481	-0.144403	0.015640	0.009
15	0.60	0.35	0.75	0.35	0.090700	-0.210593	0.007054	0.006
16	0.60	0.00	0.75	0.00	0.015500	-0.213692	-0.017680	0.028
17	0.30	0.00	0.35	0.00	0.020281	-0.147502	-0.009094	0.013
18	0.00	0.00	0.00	0.00	0.025062	-0.081312	-0.000508	0.006
19	-0.30	0.00	-0.30	0.00	0.029842	-0.015122	0.008078	0.026
20	-0.60	0.00	-0.55	0.00	0.034623	0.051069	0.016664	0.039
21	-0.60	-0.30	-0.55	-0.30	-0.040577	0.047970	-0.008071	0.002
22	-0.30	-0.30	0.25	-0.30	-0.045358	-0.018220	-0.016657	0.461
23	0.00	-0.30	0.05	-0.30	-0.050138	-0.084410	-0.025243	0.025
24	0.30	-0.30	0.40	-0.30	-0.054919	-0.150601	-0.033829	0.066
25	0.60	-0.30	0.80	-0.30	-0.059700	-0.216791	-0.042415	0.112
26	0.60	-0.05	0.80	-0.65	-0.134900	-0.219890	-0.067149	0.125
27	0.30	-0.65	0.40	-0.65	-0.130119	-0.153699	-0.058563	0.053
28	0.00	-0.65	0.05	-0.65	-0.125338	-0.087509	-0.049977	0.004
29	-0.30	-0.65	-0.25	-0.05	-0.120558	-0.021319	-0.041391	0.021
30	-0.60	-0.65	-0.55	-0.65	-0.115777	0.044871	-0.032805	0.014

TABLE 11

MATCHING LINE AND MINIMUM DISTANCE COMPUTATION OF A FOOTBALL

#	Image Pts. at Image 1		Image Pts. at Image 2		Matching Line Coefficients			Minimum Distance
	X	Y	X	Y	a	b	c	d
1	-1.85	1.00	-1.50	1.00	-2.845001	-1.944942	-4.308182	0.576
2	-1.15	1.00	-1.00	1.00	-2.941489	0.270009	-4.425313	0.410
3	-0.60	1.00	-0.50	1.00	-2.868505	2.028762	-4.293084	0.236
4	-0.05	1.00	0.00	1.00	-2.795522	3.787517	-4.160855	0.079
5	0.35	1.00	0.50	1.00	-2.701864	5.071639	-4.003526	0.049
6	0.75	1.00	1.00	1.00	-2.756999	6.337325	-4.070458	0.070
7	1.10	1.00	1.50	1.00	-2.656449	7.463236	-3.904763	0.053
8	1.20	0.50	1.50	0.50	-1.182290	7.964023	-1.678888	0.065
9	0.85	0.50	1.00	0.50	-1.282840	6.838111	-1.844583	0.041
10	0.45	0.50	0.50	0.50	-1.227705	5.572425	-1.777651	0.069
11	0.00	0.50	0.00	0.50	-1.314472	4.130093	-1.926613	0.031
12	-0.50	0.50	-0.50	0.50	-1.245552	2.547986	-1.842948	0.018
13	-1.05	0.50	-1.00	0.50	-1.318535	0.789232	-1.975177	0.170
14	-1.65	0.50	-1.50	0.50	-1.235832	-1.109297	-1.874779	0.346
15	-1.70	0.00	-1.50	0.00	0.407797	-0.064706	0.600457	0.009
16	-1.05	0.00	-1.00	0.00	0.318202	0.992033	0.491692	0.166
17	-0.50	0.00	-0.50	0.00	0.242390	2.732351	0.399661	0.101
18	0.00	0.00	0.00	0.00	0.173471	4.314458	0.315996	0.073
19	0.45	0.00	0.50	0.00	0.260238	5.756791	0.464958	0.103
20	0.85	0.00	1.00	0.00	0.205102	7.022477	0.398026	0.085
21	1.20	0.00	1.50	0.00	0.156859	8.129951	0.339460	0.070
22	1.10	-0.50	1.50	-0.50	1.807380	8.016332	2.823062	0.185
23	0.75	-0.50	1.00	-0.50	1.706829	6.890420	2.657367	0.129
24	0.35	-0.50	0.50	-0.50	1.761964	5.624734	2.724699	0.134
25	-0.10	-0.50	0.00	-0.50	1.823992	4.200839	2.799598	0.152
26	-0.60	-0.50	-0.50	-0.50	1.892911	2.618731	2.883263	0.194
27	-1.15	-0.50	-1.00	-0.50	1.819928	0.859977	2.751034	0.248
28	-1.85	-0.50	-1.50	-0.50	1.916415	-1.354973	2.868165	0.285

TABLE 12

COMPUTER RESULTS FOR VERTEX LOCATION OF A CUP

True Vertex	Computer D	Computer M
D	19	11
M	3	0

Threshold = 10 in 9 x 9 mask

True Vertex	Computer D	Computer M
D	21	9
M	1	0

Threshold = 20 in 9 x 9 mask

True Vertex	Computer D	Computer M
D	21	9
M	2	0

Threshold = 30 in 9 x 9 mask

True Vertex	Computer D	Computer M
D	20	10
M	1	0

Threshold = 40 in 9 x 9 mask

True Vertex	Computer D	Computer M
D	20	10
M	3	0

Threshold = 10 in 11x11 mask

True Vertex	Computer D	Computer M
D	21	9
M	3	0

Threshold = 20 in 11x11 mask

True Vertex	Computer D	Computer M
D	21	9
M	3	0

Threshold = 30 in 11x11 mask

True Vertex	Computer D	Computer M
D	19	11
M	3	0

Threshold = 40 in 11x11 mask

D - represents detected

M - represents missed

TABLE 13

COMPUTER RESULTS FOR VERTEX LOCATION OF A FOOTBALL

	Threshold = 10 in 9 x 9 mask		Threshold = 20 in 9 x 9 mask		Threshold = 30 in 9 x 9 mask		Threshold = 40 in 9 x 9 mask	
True Vertex	Computer D	Computer M	Computer D	Computer M	Computer D	Computer M	Computer D	Computer M
D	7	21	9	19	8	20	5	23
M	3	0	2	0	2	0	1	0

	Threshold = 10 in 11x11 mask		Threshold = 20 in 11x11 mask		Threshold = 30 in 11x11 mask		Threshold = 40 in 11x11 mask	
True Vertex	Computer D	Computer M	Computer D	Computer M	Computer D	Computer M	Computer D	Computer M
D	6	22	12	16	12	16	8	20
M	4	0	2	0	1	0	1	0

D - represents detected

M - represents missed

TABLE 14

SUMMARY OF PERCENT AGREEMENT OF DETECTED VERTEX POINTS

#	Mask size	Threshold	Cup(%)	Football(%)
1.	9 x 9	10.0	63.3	25.0
2.	9 x 9	20.0	70.0	32.1
3.	9 x 9	30.0	70.0	28.6
4.	9 x 9	40.0	67.8	17.9
5.	11 x 11	10.0	67.8	21.4
6.	11 x 11	20.0	70.0	42.9
7.	11 x 11	30.0	70.0	42.9
8.	11 x 11	40.0	63.3	28.6

7. IMPLEMENTATION

An example design to implement the algorithm of this thesis into a three-dimensional robot vision system has been studied. The IBM personal computer is chosen as the general-purpose computer for several reasons. The reasons are the availability of the supporting hardware and software, the cost of the computer, and the cost of interfacing it with the projector, the camera and the robot arm. The system configuration for the three-dimensional robot vision system is shown in Figure 22. The integration of the hardware and software can be accomplished in several phases. The first phase involves the direct connection of the IBM personal computer with the monitor, keyboard, printer, communication, double sided high density floppy disk drive and a 10 Mega-byte Winchester disk drive.

Phase 2 involves the connection of the image and graphic processing system which includes the Charge-Couple-Device (CCD) camera, image digitizer, frame grabber, image and graphics processor and a high resolution image display monitor. Most of the hardware and software for phase 1 and 2 are off the shelf items. No attempt has been made to select the image processing system in this thesis. However, a light weight, low power camera with 256 x 256 image resolution should be sufficient for normal applications. The system should be able to perform an ordinary image processing function when phase 1 and 2 are completed.

Phase 3 involves the implementation and connection of the

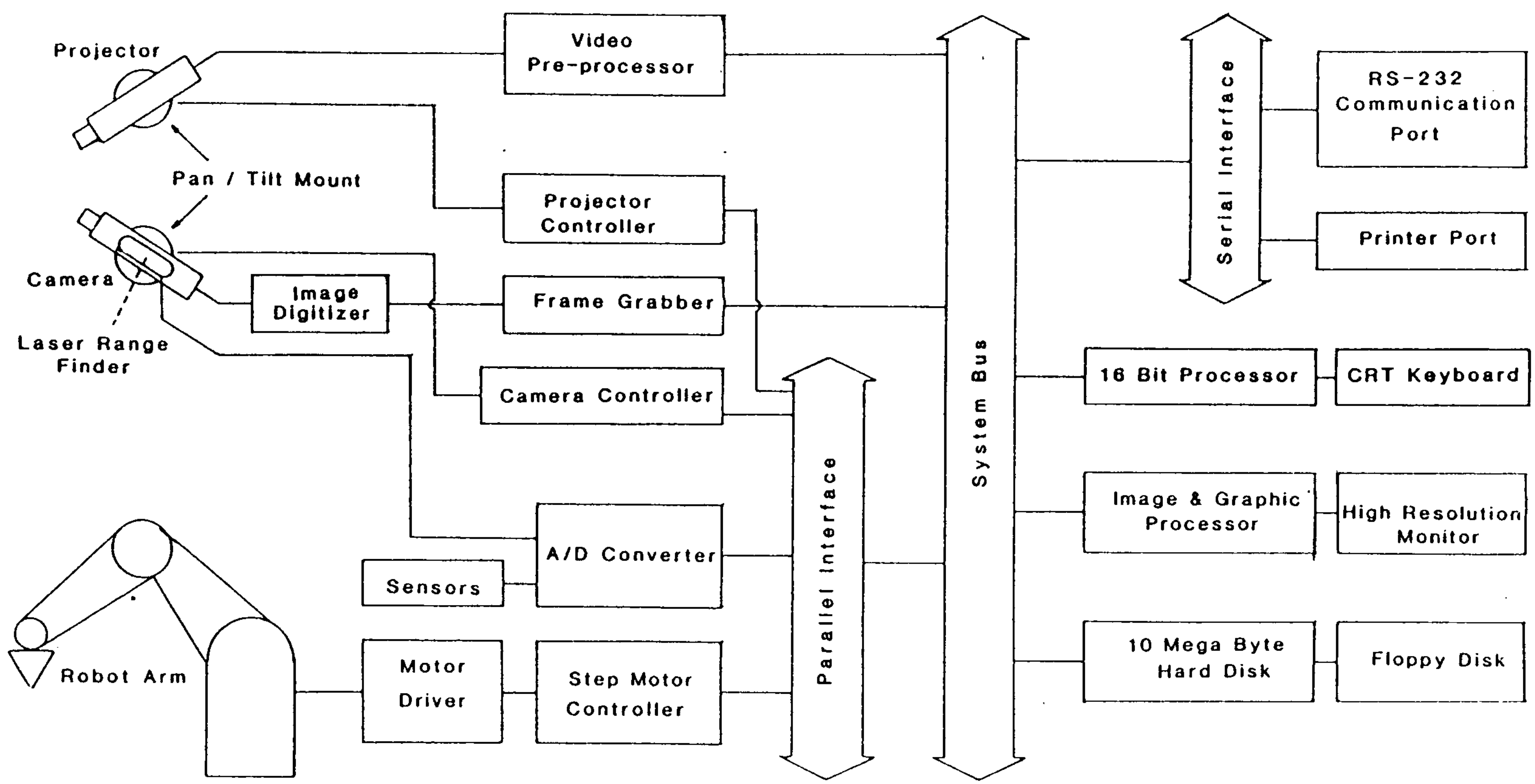


Figure 22. Three-Dimensional Robot Vision System

projection system and the camera controller to the system. The projection system consists of a high density video projector, a projector controller and a video pre-processor. The high density video projector is chosen mainly due to the ease of projecting various patterns for different applications. The projection pattern can be created or edited using the image and graphics processor. The resulting projection pattern is transferred to the video pre-processor and remains in the video pre-processor until a new or different projection pattern is requested by the operator. The video pre-processor will provide a refresh routine for the projector at a specified scan rate. The main advantage of this projection system is that the projection pattern can be stored as a picture file on the system disk. The operating system can recall the projection pattern whenever it is needed for various types of objects. For robotics applications, it is recommended that the projector and the camera be mounted on the robot arm. Figure 23 shows the projector and the camera mounted on a robot arm, which has six degrees of freedom. With the projector and camera mounted on the robot arm, the robot is able to observe the object within a reasonable distance in three-dimensional space. Another advantage of this projection system is that the orientation and position of the projector and camera can be determined. A good example of such a system is the vision system developed at the National Bureau of Standards [27]. This vision system is capable of providing the depth and orientation information to the robot control system. The depth information is acquired by mounting the camera and the projection light source at a fixed angle so that a triangular relationship can be established. In this thesis, the projector and

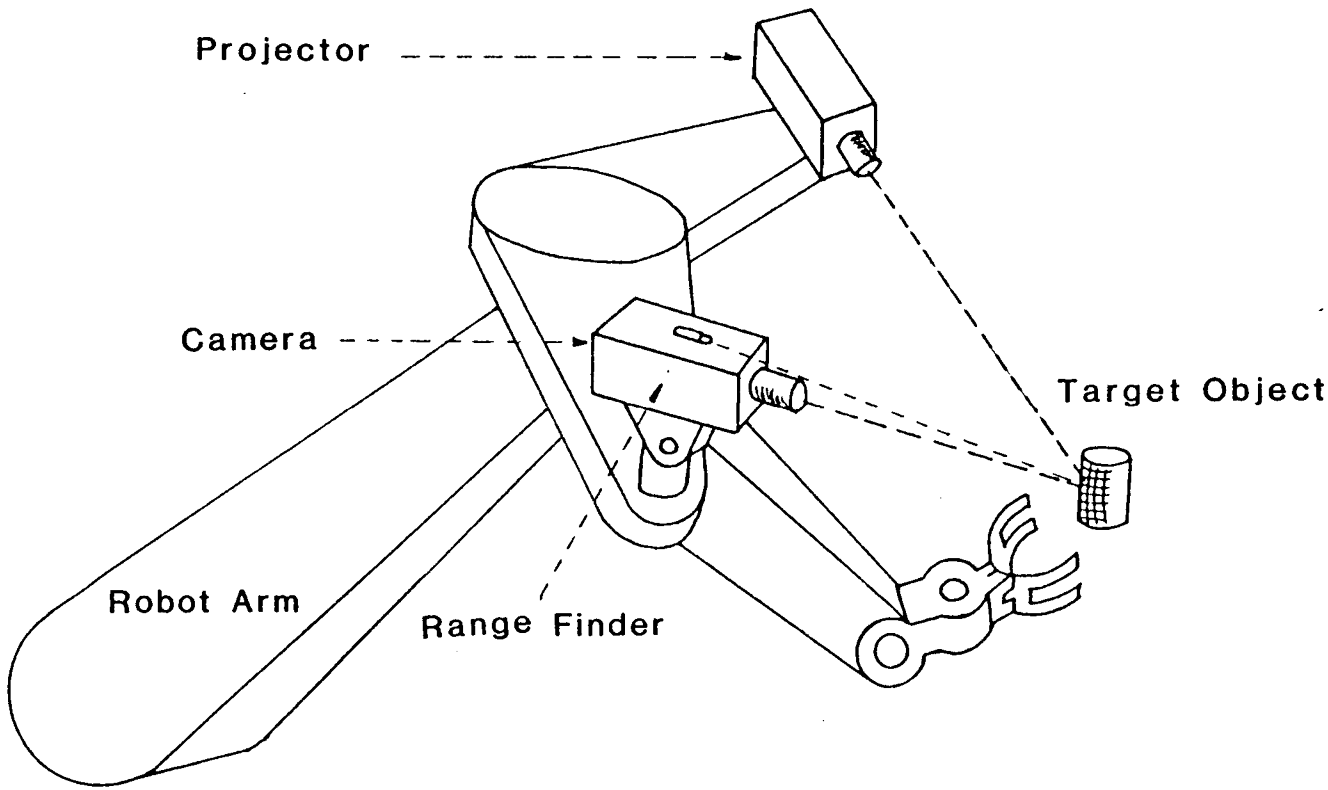


Figure 23. A Robot Arm With A Projector and Camera

camera are mounted on a fixed position with pan and tilt adjustment capability. The pan and tilt controller for the projector and camera is shown in Figure 24. Also, the controller for the robot arm is included in Figure 24 although no attempt has been made to discuss the robot arm controller. The robot arm controller is another off the shelf item which is available from the market. All the controllers are connected to the parallel interface of the IBM personal computer. The system uses a parallel interface which is connected to an intelligent positioning stepper motor controller (CY512) manufactured by Cybernetic Micro Systems. The output signals from the stepper motor controller can be used to control several motors through the use of demultiplexer such as a 74LS155 TTL chip. The output signals from the demultiplexer are connected to the pan and tilt driver of the projector and camera stepper motors. The use of an intelligent positioning stepper motor controller (CY512) is recommended because no special software is then required to communicate with the stepper motor controller. The stepper motor controller is equipped with commands which can be accessed by using ASCII, decimal, or binary communication. Because the projector and the camera are mounted on the robot arm, the stepper motor must be light-weight and small. The permanent magnet stepper motor (M65 series), which is manufactured by General Scanning, Inc., is recommended. The M65GS05 is chosen for several reasons : there are the wight, low power consumption and the small angle of tolerance.

Phase 4 involves the connection of the ranger finder to the analog to digital converter board of the IBM personal computer. The purposes of having a range finder is to provide overall depth informa-

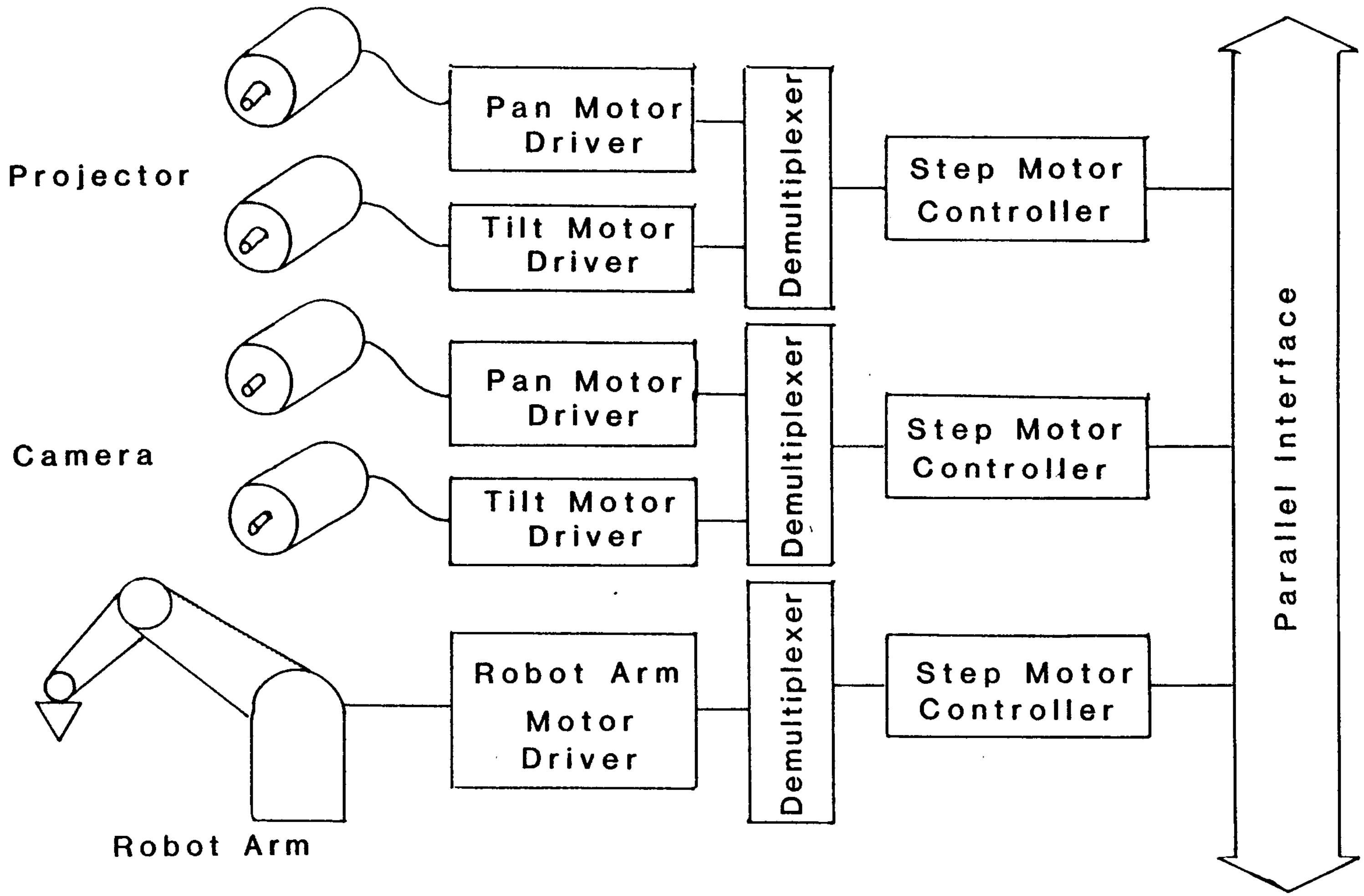


Figure 24. Projector and Camera Controller

tion of the object for the camera. Similarly, the ranger finder is another off the shelf item which is available.

Phase 5 involves the connection of the robot arm controller to the robot arm. Although no attempt has been made to select the type of the robot arm, the minimum lifting weight requirement must be able to support the projector and camera unit.

Phase 6 involves the connection of sensors for the projector, camera and robot arm so that the safety of the system and robot arm can be measured and controlled. Another advantage of having sensors on the projector and camera is that it can be used to maintain the angles of pan and tilt. This concludes the hardware implementation of the three-dimensional robot vision system.

Phase 7 involves the development of software modules for the vision system. These includes calibration, data acquisition, vertex location, measurement, three-dimensional calculation, display and control. Since most of the software is written in the FORTRAN language, the modules can be used with minimum modification.

8. APPLICATIONS AND CONCLUSIONS

A new technique for vertex location using a small mask method has been studied. Vertex location is an important step for surface measurement using projection pattern. The technique is designed for robot control applications since it requires only simple computations. All procedures in the vertex location can be automated. Using the matching line constraint, each of the true image vertices can be uniquely located. However, the accuracy of detecting the true image vertices is highly reliant on the quality of the digitized images. This can be accomplished by using computer controlled pan and tilt mounts for the camera and projector. As discussed in section 7, a low cost three-dimensional robot vision system can be easily implemented. However, it is recommended that a high speed processor be used for this system, since a large number of computations is required. For higher resolution images, such as a 256 x 256 picture, it is highly recommended that a 32-bit processor such as Motorola 68000 processor be used. For robotics applications, it is important that the image processing and all transformations are performed in real time. One way to achieve a real time solution is to use multiple processors to perform individual tasks such as edge detection, vertex location, projection and camera control, concurrently.

These algorithms have direct applications in many fields of technology. In the field of robotics, future industrial robots will be able to observe the object more accurately by using a refined projection grid pattern. Another area which extensively requires computer vision is automatic visual inspection for quality control. Automating the vertex location technique may be the solution to accuracy measurements since most of the vertices can be located. Another application which requires a large number of vertices is in radiotheraphy treatment to measure the surface of the patient's body to deliver a uniform dose.

In conclusion, the vertex location technique is not limited to robotics, since the feature extraction can be easily modified. Future development of vertex location is recommended in the hardware implementation of the technique.

LIST OF REFERENCES

LIST OF REFERENCES

[1] E. L. Hall, J. B. K. Tio, C. A. McPherson and F. A. Sadjadi, "Curved Surface Measurement and Recognition For Robot Vision," The International Workshop on Industrial Applications of Machine Vision, May, 1982.

[2] C. A. McPherson, J. B. K. Tio, F. A. Sadjadi and E. L. Hall, "Curved Surface Representation For Image Recognition," Proceedings of the Pattern Recognition and Image Processing Conference 1982, Las Vegas, June, 1982, pp. 363-369.

[3] H. C. Lee and K. S. Fu, "A Computer Vision System For Generating Object Description," Proceedings of the Pattern Recognition and Image Processing Conference 1982, Las Vegas, June, 1982, pp. 466-472.

[4] G. J. Agin, "Computer Vision Systems for Industrial Inspection and Assembly," Computer, IEEE, May, 1980, pp. 11-20.

[5] D. A. Bourne, "On Automatically Generating Programs For Real Time Computer Vision," Pattern Recognition Proceedings 1980, Miami, December, 1980, pp. 759-764.

[6] E. L. Hall, J. Lesac, W. E. Woodrick and D. J. Soukup, "An Educational Computer Vision and Robotics System," Proceedings of the Pattern Recognition and Image Processing Conference 1982, Las Vegas, June, 1982, pp. 647-649.

[7] G. B. Porter and J. L. Mundy, "Visual Inspection System Design," Computer, IEEE, May, 1980, pp. 40-48.

[8] R. Paul, "Robots, Models, and Automation." Computer, IEEE, July, 1979, pp. 19-27.

[9] J. F. Jarvis, "Visual Inspection Automation," Computer, IEEE, May, 1980, pp. 32-38.

[10] G. J. Gleason and G. J. Agin, "A Modular System for Sensor-Controlled Manipulation and Inspection," Proceedings of Ninth International Symposium Industrial Robots, Society of Manufacturing Engineers and Robot Institute of America, Washington, D.C., March, 1979, pp. 57-70.

[11] G. J. Agin and R. O. Duda, "SRI Vision Research for Advanced Automation," Proceedings of Second USA-Japan Computer Conference, Tokyo, Japan, August, 1975, pp. 113-117.

[12] R. R. Schreiber, "Robot Vision: An Eye to the Future," Robotics Today, The Society of Manufacturing Engineers, June, 1983, pp. 53-57.

[13] B. Bhanu, "Surface Representation and Shape Matching of 3-D Objects," Proceedings of the Pattern Recognition and Image Processing Conference 1982, Las Vegas, June, 1982, pp. 349-354.

[14] J. B. K. Tio, J. J. Hwang, C. A. McPherson and E. L. Hall, "Surface location in Scene Content Analysis," Proceedings of the SPIE Conference 1981, San Diego, August, 1981.

[15] F. I. Parke, "Measuring Three-Dimensional Surface With A Two-Dimensional Data Tablet," Computer & Graphics, Vol. 1, 1975, pp. 5-7.

[16] J. B. K. Tio, C. A. McPherson and E. L. Hall, "Curved Surface Measurement For Robot Vision," Proceedings of the Pattern Recognition and Image Processing Conference 1982, Las Vegas, June, 1982, pp. 370-378.

[17] D, Marr and H. K. Nishihara, "Representation and Recognition of Three-Dimensional Shapes," Al Memo 416, Massachussetts Institute of Technology Laboratory, 1977.

[18] W. D. Renner, et al., "The Use of Photogrammetry in Tissue Compensation Design," Radiology, Vol. 125, pp. 505-510, November, 1977.

[19] B. Cernuschi-Frias, D. B. Cooper, and R. M. Bolle, "Estimation of Location and Orientation of 3-D Surfaces Using A Single 2-D Image," Proceedings of the Pattern Recognition and Image Processing Conference 1982, Las Vegas, June, 1982, pp. 605-610.

[20] R. Rajarshi, "Eying The Assembly Line," Photonics Spectra, January, 1982, pp. 59-60.

[21] M. Oshima and Y. Shirai, "Representation of Curved Objects Using Three-Dimensional Information," Proceedings of the Second USA-JAPAN Computer Conference, 1975, pp. 108-112.

[22] R. J. Woodham, "A Cooperative Algorithm For Determing Surface Orientation From A Single View," Proceedings of the fifth International Joint Conference on Artificial Intelligence 1977, pp. 635-641.

[23] S. Berman, P. Parikh and G. Lee, "Computer Recognition of Overlapping Parts Using A Single Camera," Proceedings of the Pattern Recognition and Image Processing Conference 1982, Las Vegas, June, 1982, pp. 650-655.

[24] P. G. Mulgaonkar, L. G. Shapiro and R. M. Haralick, "Recognizing Three-Dimensional Objects From Single Perspective Views Using Geometric and Relational Reasoning," Proceedings of the Pattern Recognition and Image Processing Conference 1982, Las Vegas, June, 1982, pp. 479-484.

[25] R. P. Kruger and W. B. Thompson, "A Technical and Economic Assessment of Computer Vision for Industrial Inspection and Robotics Assembly," Proceedings of the IEEE, Vol. 69, No. 2, December, 1981, pp.1524-1538.

[26] M. R. Ward et al., "CONSIGHT: A Practical Vision Based Robot Guidance System," Proceedings of Ninth International Symposium Industrial Robots, Society of Manufacturing Engineers and Robot Institute of America, Washington, D.C., March, 1979, pp. 195-212.

[27] J. S. Albus, Brains, Behavior and Robotics, BYTE/McGraw-Hill, N. Y., 1981.

[28] P. M. Will and K. S. Pennington, "Grid Coding : A Novel Technique for Image Processing," Proceedings of the IEEE, Vol. 60, No.6. June 1972, pp.669-680.

[29] J. L. Posdamer and M. D. Altschuler, "Surface Measurement by Space-encoded Projected Beam Systems," Computer Graphics and Image Processing, Vol. 18, 1982, pp. 1-17.

[30] R. P. Paul, Robot Manipulators, Massachussetts Institute of Technology Press, Massachussetts, 1981.

[31] W. Myers, "Industry Begins To Use Visual Pattern Recognition," Proceedings of IEEE, Computer, May, 1980, pp. 21-31.

[32] M. Yachida and S. Tsuji, "Industrial Computer Vision in Japan," Proceedings of IEEE, Computer, May, 1980, pp. 50-63.

[33] J. J. Hwang, "Computer Stereo Vision for Three Dimensional Object Location," Ph.D Dissertation, Department of Electrical Engineering, The University of Tennessee, August, 1980.

[34] Robinson, G.S., "Detection and Coding of Edges Using Directional Mask", Optical Engineering, Vol.16, No.6, Nov.-Dec.,1977.

[35] Davies, D. L., "Edge Detection in Digital Images Using Small Mask and Vector Operators and The Method of Polar Histograms", M. S. Thesis, Department of Electrical Engineering, The University of Tennessee, December, 1978.

[36] E. L. Hall, Computer Image Processing and Recognition, Academic Press, New York, 1979.

[37] J. J. Hwang and E. L. Hall, "Matching of Feature Objects Using Relational Tables From Stereo Images," Computer Graphics and Image Processing, Academic Press, Vol. 20, 1982, pp. 22-42.

[38] J. B. K. Tio, "Single Image Surface Measurement for Robotics Application," Master Thesis, University of Tennessee, August, 1982.

APPENDIX

```
C                           *=*=*    EDGE.FOR    *=*=*
C
C*******************************************************************************
C                                                                              *
C       Subject : Edge detection                                               *
C       Programmer : Min-Ching Chiang                                          *
C       Date : Aug. 1, 1983.                                                   *
C       program description : This program is using the standard gradient     *
C               operator to perform the edge detection.                        *
C                                                                              *
C       Input : File name must be " *.PIC "                                    *
C               The input file is a 128 x 128, 40 blocks image picture file    *
C                                                                              *
C       Output : File name must be " *.PIC "                                   *
C               The input file is a 128 x 128, 40 blocks image picture file    *
C                                                                              *
C*******************************************************************************
C
C
        INTEGER*2       FILSPC(39),DEFEXT(4)
        LOGICAL*1       LPIC1(16384),LPIC2(16384)
        LOGICAL*1       LP1(128,128),LP2(128,128)
        LOGICAL*1       FILNAM(28)
        DATA DEFEXT/4*3RPIC/
C
C       *** INPUT IMAGE FILE ***
C
        TYPE 10
10      FORMAT(10X,'ENTER THE INPUT IMAGE FILE ; ',$)
        ACCEPT 20,J,(FILNAM(I),I=1,J)
20      FORMAT(Q,28A1)
        CALL ASSIGN (1,FILNAM,J)
        DEFINE FILE 1(40,256,U,IBLKI)
C
C       *** ENTER OUTPUT FILE NAME ***
C
        TYPE 11
11      FORMAT(10X,'ENTER THE OUTPUT FILENAME ; ',$)
        ACCEPT 20,J,(FILNAM(I),I=1,J)
        CALL ASSIGN (2,FILNAM,J)
        DEFINE FILE 2(40,256,U,IBLKO)
C
C       *** READ AN IMAGE TO AN ARRAY ***
C
        IBKLI=9
        IBKLO=1
        DO 40 N=0,31
        IS1=N*512+1
        IS2=IS1+511
40      READ(1'IBLKI)(LPIC1(I),I=IS1,IS2)
C
C       *** PIXEL MANIPULATION ***
C
        DO 50 IX=0,126
        DO 50 IY=0,126
        MAG=(LPIC1(IY*128+IX+1)-LPIC1(IY*128+IX+2))
        1 +(LPIC1((IY+1)*128+IX+1)-LPIC1((IY+1)*128+IX=2))
        RMAG=FLOAT(MAG/2)**2.0
        MAG=(LPIC1((IY+1)*128+IX+1)-LPIC1(IY*128+IX+1))
        1 +(LPIC1((IY+1)*128+IX+2)-LPIC1(IY*128+IX+2))
        RMAG=RMAG+FLOAT(MAG/2)**2.0
```

```
         RMAG=ABS(RMAG)**2.0
         MAG=RMAG
        LPIC2(IY*128+IX=1)=ITRUN(MAG)
50      CONTINUE
C
C       *** ZERO THE BORDERS ***
C
        DO 60 I=1,127
        LPIC2(I*128+128)=-128
        LPIC2(127*128+I+1)=-128
60      CONTINUE
C
C       *** OUTPUT THE IMAGE ***
C
        IBLKO=9
        DO 70 N=0,31
        IS1=N*512+1
        IS2=IS1+511
70      WRITE(2'IBLKO)(LPIC2(I),I=IS1,IS2)
        CALL CLOSE(1)
        CALL CLOSE(2)
C
C       *** END OF PROGRAM ***
C
        STOP 'NORMAL'
        END
```

```
C
C
C                         ***   TRACE.FOR   ***
C
C***********************************************************************
C                                                                      *
C       Subject : Vertex extraction                                    *
C       Programmer : Min-Ching Chiang                                  *
C       Date : Aug. 1, 1983.                                           *
C       program description : This program is using the small mask     *
C               technique to extract the vertices of the grid pattern  *
C                                                                      *
C       Input : File name must be " *.PIC "                            *
C               The input file is a 128 x 128, 40 blocks image picture file  *
C                                                                      *
C       Output : File name must be " *.DAT "                           *
C                The output contains the detected vertex points coordinstes  *
C                for the object image.                                 *
C                                                                      *
C***********************************************************************
C
C
        INTEGER*2       IBOUND(500),JBOUND(500)
        INTEGER*2       ICOUNT,ICIR(9),JCIR(9)
        INTEGER*2       ICIR2(9),JCIR2(9)
        INTEGER*2       MASK(8,8,8),MBUF(9)
        REAL*4          BACKGR,BUFAVG,THRES,CALTHR
        LOGICAL*1       LPIC(16384),LP(128,128)
        LOGICAL*1       FILNAM(28)
        EQUIVALENCE     (LPIC(1),LP(1,1))
        DATA ICOUNT/1/
        DATA ICIR/-3,0,3,3,0,-3,-3,0,3/
        DATA JCIR/-3,-3,-3,0,0,0,3,3,3/
        DATA ICIR2/-4,0,4,4,0,-4,-4,0,4/
        DATA JCIR2/-4,-4,-4,0,0,0,4,4,4/
C
C       *** INPUT IMAGE FILE ***
C
        TYPE 10
10      FORMAT(10X,'ENTER THE EDGE IMAGE FILE ; ',$)
        ACCEPT 20,J,(FILNAM9I),I=1,J)
20      FORMAT(Q,28A1)
        CALL ASSIGN (1,FILNAM,J)
        DEFINE FILE 1(40,256,U,IBLK1)
C
C       *** OPEN A FILE TO STORE COORDINATES ***
C
        TYPE 30
30      FORMAT(10X,'ENTER THE OUTPUT FILENAME FOR COORDINATES : ',$)
        ACCEPT 20,J,(FILNAM(I),I=1,J)
        OPEN(UNIT=3,NAME=FILNAM,TYPE='NEW')
C
C       *** READ AN IMAGE TO AN ARRAY ***
C
        IBKL1=9
        DO 40 N=0,31
        IS1=N*512+1
        IS2=IS1+511
40      READ(1'IBLK1)(LPIC(I),I=IS1,IS2)
        CLOSE(UNIT=1)
C
C       *** INITIALIZES MASK TABLE FOR IDENTIFICATION PURPOSE ***
C
```

```
      DO 70 L1=1,8
       DO 60 L2=1,8
        DO 50 L3=1,8
         MASK(L1,L2,L3)=50
50      CONTINUE
60     CONTINUE
70    CONTINUE
C
C     *** ASSIGN THE MASK VALUE ***
C
      MASK(1,1,1)=20
      MASK(3,8,3)=2
      MASK(1,8,8)=3
      MASK(1,8,1)=4
      MASK(8,8,1)=5
      MASK(1,1,8)=6
      MASK(8,8,1)=7
      MASK(7,4,7)=8
      MASK(4,7,4)=9
      MASK(4,8,7)=10
      MASK(7,8,4)=11
      MASK(3,3,3)=12
      MASK(5,3,2)=13
      MASK(2,3,5)=14
      MASK(6,1,6)=15
      MASK(8,1,8)=16
      MASK(8,8,8)=17
C
C     *** SEARCH FOR THE BACKGROUND WINDOW ***
C
      CALL AVERAG(LP,3,3,CTHRES)
      TYPE 80
80    FORMAT(5X,' THRESHOLD VALUE IS : ',$)
      ACCEPT 90,THRES
90    FORMAT(F6.2)
      DO 130 ISTART=1,128
      DO 130 JSTART=1,128
      CALL BKGRD(LP,ISTART,JSTART,THRES,N1,N2,N3)
      IF(MASK(N1,N2,N3) .EQ. 20) GO TO 130
C
C     *** CONTINUATION FOR EXTENDED REGION SIZE ***
C
      DO 110 L=1,9
      INEXT=ISTART+ICIR(L)
      JNEXT=JSTART+JCIR(L)
      CALL AVERAG(LP,INEXT,JNEXT,CALTHR)
      IF(CALTHR .GT. THRES) GO TO 100
      MBUF(L)=0
      GO TO 110
100   MBUF(L)=1
110   CONTINUE
      CALL SEARCH(MBUF,M1,M2,M3)
      IF(MASK(M1,M2,M3) .NE. 2) GO TO 130
C
C     *** THE VERTEX HAS BEEN EXTRACTED ***
C
      IBOUND(ICOUNT)=ISTART
      JBOUND(ICOUNT)=JSTART
      TYPE 120,ICOUNT,JBOUND(ICOUNT),IBOUND(ICOUNT)
120   FORMAT(5X,'$ ',I3,4X,'X= ',I6,10X,'Y= ',I6)
      ICOUNT=ICOUNT+1
130   CONTINUE
C
```

```
C         *** OUTPUT THE COORDINATES TO A FILE ***
C
          DO 150 L5=1,ICOUNT
          WRITE(3,140)JBCUND(L5),IBOUND(L5)
140       FORMAT(2I5)
150       CONTINUE
C
C         *** END OF MAIN PROGRAM ***
C
          CLOSE(UNIT=3)
          STOP 'NORMAL'
          END
```

```
C
C
C                          ****   AVERAGE.FOR   ****
C
C***************************************************************************
C                                                                          *
C       Subject : Averaging the gray level of the small window            *
C       Programmer : Min-Ching Chiang                                      *
C       Date : Aug. 1, 1983.                                               *
C       Program description : This subroutine calculates the average      *
C               gray level of the small mask.                              *
C                                                                          *
C       Input : The coordinates of each pixel in the image.               *
C                                                                          *
C       output : The average gray level of each small window.             *
C                                                                          *
C***************************************************************************
C
C
        SUBROUTINE AVERAG(LP,I,J,THRES)
        REAL*4 JBUF(9),BUFF,THRES
        LOGICAL*1 LP(128,128)
C
C       *** PIXEL MANIPULATION ***
C
        JBUF(1)=FLOAT(LP(I-1,J-1))
        JBUF(2)=FLOAT(LP(I-1,J))
        JBUF(3)=FLOAT(LP(I-1,J+1))
        JBUF(4)=FLOAT(LP(I,J+1))
        JBUF(5)=FLOAT(LP(I,J))
        JBUF(6)=FLOAT(LP(I,J-1))
        JBUF(7)=FLOAT(LP(I+1,J-1))
        JBUF(8)=FLOAT(LP(I+1,J))
        JBUF(9)=FLOAT(LP(I+1,J+1))
C
C       *** CALCULATE THE AVERGE OF THE WINDOW ***
C
        ICNT=0
        BUFF=0.0
        DO 10 K=1,9
        IF(JBUF(K) .LT. THRES) GO TO 10
        BUFF=BUFF+JBUF(K)
        ICNT=ICNT+1
10      CONTINUE
        IF(BUFF .NE. 0.0) GO TO 20
        THRES=0.0
        GO TO 30
20      THRES=BUFF/THRES
C
C       *** END OF SUBROUTINE ***
C
30      RETURN
        END
```

```
C
C
C                          ****   BACKGROUND.FOR   ****
C
C***********************************************************************
C                                                                      *
C       Subject : Convert original image to binary image subroutine    *
C       Programmer : Min-Ching Chiang                                  *
C       Date : Aug. 1, 1983.                                           *
C       Program description : This is a subroutine to convert the image *
C               into a binary image.                                   *
C                                                                      *
C       Input : Gray level of each pixel.                              *
C                                                                      *
C       Output : Mask value of the small mask.                         *
C                                                                      *
C***********************************************************************
C
C
        SUBROUTINE BKGRD(LP,I,J,THRES,N1,N2,N3)
        REAL*4 JBUF(9),BUFF,THRES
        INTEGER*2 NBUF(9)
        LOGICAL*1 LP(128,128)
C
C       *** PIXEL MANIPULATION ***
C
        JBUF(1)=FLOAT(LP(I-1,J-1))
        JBUF(2)=FLOAT(LP(I-1,J))
        JBUF(3)=FLOAT(LP(I-1,J+1))
        JBUF(4)=FLOAT(LP(I,J+1))
        JBUF(5)=FLOAT(LP(I,J))
        JBUF(6)=FLOAT(LP(I,J-1))
        JBUF(7)=FLOAT(LP(I+1,J-1))
        JBUF(8)=FLOAT(LP(I+1,J))
        JBUF(9)=FLOAT(LP(I+1,J+1))
        DO 10 K=1,9
        IF(JBUF(K) .GT. THRES) GO TO 20
        NBUF(K)=0
        GO TO 10
20      NBUF(K)=1
10      CONTINUE
        CALL SEARCH(NBUF,M1,M2,M3)
        N1=M1
        N2=M2
        N3=M3
C
C       *** END OF THE SUBROUTINE ***
C
        RETURN
        END
```

```
C
C
C                         *==*   SEARCH.FOR   *==*
C
C***************************************************************************
C                                                                          *
C       Subject : Search for mask value subroutine.                        *
C       Programmer : Min-Ching Chiang                                      *
C       Date : Aug. 1, 1983.                                               *
C       Program description ; This subroutine computes the mask vale       *
C              of each window.                                             *
C                                                                          *
C       Input : Binary gray level of each pixel.                           *
C                                                                          *
C       Output : The mask value of each window.                            *
C                                                                          *
C***************************************************************************
C
C
        SUBROUTINE SEARCH(MBUF,M1,M2,M3)
        INTEGER*2 MBUF(9)
C
C       *** APPLY THE WEIGHT FACTOR TO EACH PIXEL IN 3 X 3 WINDOW ***
C           WHERE M1 - MASK VALUE FOR ROW 1 IN 3 X 3 WINDOW
C                 M2 - MASK VALUE FOR ROW 2 IN 3 X 3 WINDOW
C                 M3 - MASK VALUE FOR ROW 3 IN 3 X 3 WINDOW
C
        M1=MBUF(1)*4+MBUF(2)*2+MBUF(3)+1
        M2=MBUF(4)*4+MBUF(5)*2+MBUF(6)+1
        M1=MBUF(7)*4+MBUF(8)*2+MBUF(9)+1
        RETURN
        END
```

```
C
C
C                          ****    PLOT.FOR    ****
C
C***************************************************************************
C                                                                          *
C       Subject : Display the x, y coordinates on the screen.              *
C       Programmer : Min-Ching Chiang                                      *
C       Date : Aug. 5, 1983.                                               *
C       Program description : This program is using the graphics           *
C               technique to display the x, y coordinates on the sreen.    *
C                                                                          *
C       Input : File name must be " *.dat "                                *
C               Input file contains the x, y coordinates of the detected   *
C               vertices.                                                  *
C                                                                          *
C       Output : Displayed on the screen.                                  *
C                                                                          *
C***************************************************************************
C
C
        EXTERNAL ICSR
        INTEGER*2 AX(500,2),A(500,2)
        INTEGER*2 IB(16),ITTA(256),ITTB(16,16)
        LOGICAL*1 FNAME(28)
        EQUIVALENCE (ITTA,ITTB)
        DATA IB/0,2,8,10,32,34,40,42,128,130,136,138,160,162,168,170/
C
C       *** ENTER THE INPUT DATA FILE ***
C
        TYPE 10
10      FORMAT(1X,' ENTER THE INPUT DATA FILE')
        ACCEPT 20,J,(FNAME(I),I=1,J)
20      FORMAT90,28A1)
        FNAME(J+1)=0
        OPEN(UNIT=1,NAME=FNAME,FORM='FORMATTED',TYPE='OLD')
        TYPE 30
30      FORMAT(1X,' ENTER THE NUMBER OF POINTS')
        ACCEPT 40,N
40      FORMAT(I3)
C
C       *** READ IN THE DATA FILE ***
C
        READ(1,50) ((A(I,J),J=1,2),I=1,N)
50      FORMAT(2I5)
        DO 60 I=1,N
        DO 60 J=1,2
        AX(I,J)=A(I,J)*4
60      CONTINUE
C
C       *** SETUP THE DE ANZA IMAGE PROCESSOR ***
C
        CALL IP5INI
        CALL SETUP
        JCSR=IADDR(ICSR)
        DO 70 I=0,1999
        I1=I
        CALL WRANNO(I1,0,1,0)
70      CONTINUE
        CALL IPOKE(JCSR,"61040)
        DO 90 I=1,16
        DO 80 J=1,16
        ITTB(J,I)=IB(J)
```

```
80      CONTINUE
90      CONTINUE
C
C       *** DRAW THE BOARDER LINE OF THE IMAGE ***
C
        CALL LOADIT(16,ITTA)
        CALL VECTOR(4,0,2,511,2,2,255)
        CALL VECTOR(4,511,511,2,255)
        CALL VECTOR(4,0,511,2,255)
        CALL VECTOR(4,0,2,2,255)
C
C       *** DRAW THE VERTICES ON THE SCREEN ***
C
        DO 100 K=1,N
        CALL VECTOR(4,AX(K,1),(512-AX(K,2)),0,255)
        IX=AX(K,1)
        IY=512-AX(K,2)
        CALL VECTOR(4,IX,IY+5,IX,IY-5,3,255)
        CALL VECTOR(4,IX-5,IY,IX+5,IY,3,255)
        CALL VECTOR(4,IX+2,IY+2,IX+2,IY-2,3,255)
        CALL VECTOR(4,IX-2,IY+2,IX-2,IY-2,3,255)
        CALL VECTOR(4,IX-2,IY+2,IX+2,IY+2,3,255)
        CALL VECTOR(4,IX-2,IY-2,IX+2,IY-2,3,255)
100     CONTINUE
        STOP
        END
```

```
C
C
C                        *==*   PROGRAM MATLIN.FOR   *==*
C
C***********************************************************************
C                                                                      *
C       Subject : Matching line algorithm                              *
C       Programmer : Min-Ching Chiang                                  *
C       Date : Nov. 12, 1983.                                          *
C       Program description : This program is used to match a pair of points *
C               by using the matching line equation.                   *
C                                                                      *
C       Input : A pair of Matching points.                             *
C                                                                      *
C       Output : Filename must be " *.dat "                            *
C                The output contains the line equation and the minimum *
C                distance of two matching points.                      *
C                                                                      *
C***********************************************************************


        LOGICAL*1 FNAME
        LOGICAL*4 SING
        REAL*4 A1(3,4),A2(3,4),C1(4,4),C2(4,4)
        REAL*4 V1(1,2),V2(1,2)
        COMMON /PMATCH/C3
C
C       *** ENTER PROJECTOR AND CAMERA PERSPECTIVE PROJECTION ***
C               TRANSFORMATION MATRICES DATA FILE
C
        TYPE 10
10      FORMAT(1X,'ENTER THE INPUT MATRIX DATA FILE')
        ACCEPT 20,J,(FNAME(I),I=1,J)
20      FORMAT(Q,28A1)
        FNAME(J+1)=0
        OPEN(UNIT=1,NAME=FNAME,FORM='FORMATTED',TYPE='OLD')
C
C       *** ENTER GRID IMAGE AND OBJECT IMAGE COORDINATES DATA FILE ***
C
30      TYPE 40
40      FORMAT(1X,'ENTER X,Y COORDINATES DATA FILE')
        ACCEPT 2,J,(FNAME(I),I=1,J)
        FNAME(J+1)=0
        OPEN(UNIT=3,NAME=FNAME,FORM='FORMATTED',TYPE='OLD')
C
C       *** ENTER NUMBER OF OBJECT COORDINATES TO BE CALCULATED ***
C
        TYPE 50
50      FORMAT(1X,'ENTER NUMBER OF POINTS : ',$)
        ACCEPT 60, N
60      FORMAT(I6)
        DO 80 I=1,30
        READ(3,70)(U1(I,J),J=1,2)
70      FORMAT(2F12.6)
80      CONTINUE
        DO 90 I=1,N
        READ(3,70)(U2(I,J),J=1,2)
90      CONTINUE
        TYPE 70,((U1(I,J),J=1,2),I=1,30)
        TYPE 70,((U2(I,J),J=1,2)I=1,N)
C
C       *** OUTPUT MATCHING LINE EQUATION COEFFICIENT TO A FILE ***
C
```

```
      TYPE 100
100   FORMAT(1X,'ENTER THE OUTPUT DATA FILE')
      ACCEPT 2,J,(FNAME(I),I=1,J)
      FNAME(J+1)=0
      OPEN(UNIT=2,NAME=FNAME,FORM='FORMATTED',TYPE='NEW')
C
C     *** ENTER MINIMUM ACCEPTABLE DISTANCE FROM THE MATCHING LINE ***
C
      TYPE 110
110   FORMAT(1X,'ENTER THE MINIMUM ACCEPTABLE DISTANCE: ',$)
      ACCEPT 120,THRES
120   FORMAT(F6.4)
      DO 130 I=1,3
130   READ(1,150)(A1(I,J),J=1,4)
      DO 140 I=1,3
140   READ(1,150)(A2(I,J),J=1,4)
150   FORMAT(4F6.4)
C
C     *** REARRANGE THE DATA FILE TO MATICES ***
C
      DO 160 I=1,3
      DO 160 J=1,4
      C1(I,J)=A1(I,J)
      C2(I,J)=A2(I,J)
160   CONTINUE
      DO 170 I=1,3
      C1(4,I)=0.
      C2(4,I)=0.
170   CONTINUE
      C1(4,4)=1.
      C2(4,4)=1.
C
C     *** CALL DOUBLE PRECISION INVERSE MATRIX SUBROUTINE ***
C
      CALL DPINVT(C1,4,4,SING)
      TYPE 180
180   FORMAT(1X,'THE INVERTED MATRIX IS')
      TYPE 190,((C1(I,J),J=1,4),I=1,4)
190   FORMAT(4F15.6)
      WRITE(2,180)
      WRITE(2,190)((C1(I,J),J=1,4),I=1,4)
C
C     *** CALL MATRIX MULTIPLICATION SUBROUTINE ***
C
      CALL MULTI(C2,4,4,4,C1,C3)
      TYPE 200
200   FORMAT(1X,'THE TRANSFORMATIOIN MATRIX IS')
      TYPE 190,((C3(I,J),J=1,4),I=1,4)
      WRITE(2,200)
      WRITE(2,190)((C3(I,J),J=1,4),I=1,4)
C
C     *** CALCULATE MATCHING LINE EQUATION ***
C
      DO 220 K=1,30
      DO 210 L1=1,N
      V1(1,1)=U1(K,1)
      V1(1,2)=U1(K,2)
      V2(1,1)=U2(L1,1)
      V2(1,2)=U2(L1,2)
      CALL CMATCH(V1,V2,L,TT1,S8,S13,S112,THRES)
      IF (L.NE.1) GO TO 210
      TYPE 201,U1(K,1),U1(K,2)
201   FORMAT(1X,'THE POINTS AT IMAGE #1 IS',2F12.6)
```

```
      TYPE 202,U2(L1,1),U2(L1,2)
202   FORMAT(1X,'THE POINTS AT IMAGE #2 IS',2F12.6)
      WRITE(2,201)U1(K,1),U1(K,2)
      WRITE(2,202)U2(L1,1),U2(L1,2)
      TYPE 203,S8,S13,S12
203   FORMAT(1X,'THE LINE EQUATION IS',3F15.6)
      WRITE(2,203)S8,S13,S12
      TYPE 204,L,TT1
      WRITE(2,204)L,TT1
204   FORMAT(1X,'L=',I4,4X,'TT1=',F12.6)
210   CONTINUE
220   CONTINUE
      CLOSE(UNIT=1)
      CLOSE(UNIT=2)
      CLOSE(UNIT=3)
      STOP 'NORMAL END'
      END
```

```
C
C
C                        *===    DPINVT.FOR     *==*
C
C*****************************************************************************
C                                                                            *
C       Subject : Double precision inverse matrix                            *
C       Programmer : Min-Ching Chiang                                        *
C       Date : Nov. 12, 1983.                                                *
C       Program description : This subroutine computes the inverse matrix    *
C               of a M x N matrix.                                           *
C                                                                            *
C       Input : A set of M x N matrix data                                   *
C                                                                            *
C       Output : A set of N x M matrix data                                  *
C                                                                            *
C*****************************************************************************
C
C
        SUBROUTINE DPINVT(A,ORDER,DIM,SING)
        INTEGER ORDER,DIM
        DIMENSION A(ORDER,DIM),B(40),C(40)
        LOGICAL SING
        INTEGER IP(40),IQ(40)
        N=ORDER
        DO 1 K=1,N
C
C       *** DETERMINE PIVOT POINT ***
C
        PIVOT=0.
        DO 100 I=K,N
        DO 100 J=K,N
        IF(ABS(A(I,J))-ABS(PIVOT))100,100,101
101     PIVOT=A(I,J)
        IP(K)=I
        IQ(K)=J
100     CONTINUE
        IF(PIVOT)102,900,102
102     IPK=IP(K)
        IQK=IQ(K)
C
C       *** EXCHANGE THE PIVOT ROW WITH KTH ROW ***
C
        IF(IPK-K)200,299,200
200     DO 201 J=1,N
        Z=A(IPK,J)
        A(IPK,J)=A(K,J)
201     A(K,J)=Z
299     CONTINUE
C
C       *** EXCHANGE THE PIVOT ROW WITH THE KTH COLUMN ***
C
        IF(IQK-K)300,399,300
300     DO 301 I=1,N
        Z=A(I,IQK)
        A(I,IQK)=A(I,K)
301     A(I,K)=Z
399     CONTINUE
C
C       *** JORDAN ELIMINATION STEP ***
C
        DO 400 J=1,N
        IF(J-K)403,402,403
```

```
402     B(J)=1./PIVOT
        C(J)=1.
        GO TO 404
403     B(J)=-A(K,J)/PIVOT
        C(J)=A(J,K)
404     A(K,J)=0.
400     A(J,K)=0.
        DO 405 I=1,N
        DO 405 J=1,N
405     A(I,J)=A(I,J)+C(I)*B(J)
1       CONTINUE
C
C       *** THE REMAINING MATRIX ***
C
        K=N
        DO 500 KDUM=1,N
        IPK=PI(K)
        IQK=IQ(K)
        IF(IPK-K)501,502,501
501     DO 503 I=1,N
        Z=A(I,IPK)
        A(I,IPK)=A(I,K)
503     A(I,K)=Z
502     IF(IQK-K)504,500,504
504     DO 506 J=1,N
        Z=A(IQK,J)
        A(IQK,J)=A(K,J)
506     A(K,J)=Z
500     K=K-1
        SING =.FALSE.
        RETURN
900     SING=.TRUE.
        RETURN
        END
```

```
C
C
C                               *==*     MULTI.FOR     *==*
C
C*****************************************************************************
C                                                                            *
C       Subject : Matrix multiplication subroutine                           *
C       Programmer : Min-Ching Chiang                                        *
C       Date : Nov. 12, 1983.                                                *
C       Program description : This subroutine computes the product of       *
C               two matrices.                                                *
C                                                                            *
C       Input : Two sets of matrices data.                                   *
C                                                                            *
C       Output : A set of matrix data.                                       *
C                                                                            *
C*****************************************************************************
C
C
        SUBROUTINE MULTI(A,B,M,L,N,C)
        REAL*4 A(M,L),B(L,N),C(M,N)
        DO 300 I=1,M
          DO 200 J=1,N
            C(I,J)=0.0
              DO 100 INDEX=1,L
                C(I,J)=C(I,J)+A(I,INDEX)*B(INDEX,J)
100           CONTINUE
200       CONTINUE
300     CONTINUE
        RETURN
        END
```

```
C
C
C                         *==*   CMATCH.FOR   *==*
C
C*************************************************************************
C                                                                        *
C      Subject : Matching line equation subroutine.                      *
C      Programmer : Min-Ching Chiang                                     *
C      Date : Nov. 12, 1983.                                             *
C      Program description : This subroutine computes the matching line  *
C              equation for a pair of points, and also calculates the    *
C              minimum distance of these two points.                     *
C                                                                        *
C      Input : A matched images transformation matrix and a set of a     *
C              pair of points.                                           *
C                                                                        *
C      Output : Matching line equation and minimum distance of each      *
C               pair of points.                                          *
C                                                                        *
C*************************************************************************
C
C
      SUBROOUTINE CMATCH(V1,V2,L,TT1,S8,S13,S12,THRES)
      REAL*4 C3(4,4),THRES
      REAL*4 V1(1,2),V2(1,2)
      COMMON /PMATCH/C3
      L=0
      S1=(C3(1,1)-C3(1,4)*C3(3,1)/C3(3,4))*V1(1,1)
      S2=(C3(1,2)-C3(1,4)*C3(3,2)/C3(3,4))*V1(1,2)
      S3=C3(1,3)-C3(1,4)*C3(3,3)/C3(3,4)
      S4=S1+S2+S3
      S5=(C3(2,1)-C3(2,4)*C3(3,1)/C3(3,4))*V1(1,1)
      S6=(C3(2,2)-C3(2,4)*C3(3,2)/C3(3,4))*V1(1,2)
      S7=C3(2,3)-C3(2,4)*C3(3,3)/C3(3,4)
      S8=S5+S6+S7
      S9=C3(1,4)/C3(3,4)
      S0=C3(2,4)/C3(3,4)
      S10=S8*S9
      S11=S4*S0
      S12=S11-S10
      S13=-S4
      S14=SQRT(S8**2+S13**2)
      TT=S8*V2(1,1)-S4*V2(1,2)+S12
      TT=TT/S14
      TT1=ABS(TT)
      IF9TT1.LT.THRES)L=1
      RETURN
      END
```

VITA

Min Ching Chiang was born on December 17, 1957, at Tainan, Republic of China. She graduated from National Junior College, Singapore, in 1976. In the spring of 1976, she was admitted to The University of Tennessee, Knoxville and began study toward a Bachelor of Science degree. She graduated from The University of Tennessee with a Bachelor of Science in ELectrical Engineering in March of 1980. In the fall of 1980, she was admitted to the graduate school at The University of Tennessee and began study toward a Master of Science degree in Electrical Engineering. She has been a research assistant in Image and Pattern Analysis Laboratory at The University of Tennessee where she participated in research on image processing, pattern recognition, computer graphics, medical image processing, and computer vision and robotics.

She is married to James B. K. Tio of Indonesia.