THREE DIMENSIONAL TRACKING OF FOUR POINT PLANAR PATTERNS USING CORNERS

By

© Michael J. Frendo, B.Sc., M. Eng.

A Thesis

Submitted to the School of Graduate Studies

in Partial Fulfillment of the Requirements

for the Degree

Doctor of Philosophy

McMaster University

1989

THREE DIMENSIONAL TRACKING OF FOUR POINT PLANAR PATTERNS

DOCTOR OF PHILOSOPHY (1989)
(Electrical and Computer Engineering)

McMASTER UNIVERSITY
Hamilton, Ontario

TITLE: Three-dimensional Tracking of Four Point Planar Patterns Using Corners

AUTHOR: Michael J. Frendo, B.Sc. (University of Western Ontario)
M.Eng (McMaster University)

SUPERVISORS: Dr. R. Kitai and Dr. D. Capson

NUMBER OF PAGES: xiii, 154

ABSTRACT

This thesis concerns tracking of planar shapes using a single camera view. Two dimensional silhouette projections of planar shapes randomly oriented and positioned in the camera field of view are processed to determine their position and rotational orientation with respect to the camera. This information can then be provided to robots or automated guided vehicles for automated assembly, docking, shape identification and retrieval.

Silhouettes are initially encoded in the system as a linked list of vertex points representing changes in the direction of the contour. This contour is employed to get an initial estimate of the shape's position. Corner feature points of the shape are then more accurately located using a fast corner location algorithm. A photogrammetric solution is then applied to the projection points of the corners to determine the three dimensional position of the shape. The photogrammetric technique employed does not require a least squares fit as does the traditional photgrammetric approach. This new approach is contrasted with the traditional method in terms of accuracy and computational complexity.

ACKNOWLEDGEMENTS

The author is indebted to Reuven Kitai and David Capson for guidance and encouragement throughout their supervision of this work. Gratitude is also extended to the Natural Sciences and Engineering Research Council and McMaster University for financial assistance. The members of the image analysis lab, Randy Tsang, Sai-Kit Eng, Michael Anderson, and Rick Maludzinski are also acknowledged for the affable atmosphere created during the course of this work.

Finally, a very special thank you to my wife whose patience and love during the good times and the bad have enabled this work to reach fruition.

Table of Contents

LIST OF FIGURES

Page

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

AGV	-	Automated guided vehicle
NRC	-	National Research Council
3-D	-	Three dimensional
DSP	-	Digital signal processor
PC	-	Personal computer
CCD	-	Charge coupled device
RLSE	-	Run length segment encoder
FIFO	-	First-in first-out
RAM	-	Random access memory
I/O	-	Input/output
ms	-	millisecond(s)
M_{ij}	-	i,j moment of inertia
x_c, y_c	-	coordinates of the centre of area of a shape
X_a	-	x moment of the shape area in the window
Y_a	-	y moment of the shape area in the window
A	-	area of the shape within the window
A_a	-	shape area intersecting the left edge
A_b	-	shape area intersecting the top edge
A_c	-	shape area intersecting the right edge
A_d	-	shape area intersecting the bottom edge
Y_a	-	y moment of shape area intersecting the left edge
X_b	-	x moment of shape intersecting the top edge
Y_c	-	y moment of shape intersecting the right edge

X_d - x moment of shape intersecting the bottom edge

X_j, Y_j, Z_j - coordinates of a control point j on the object

X_L, Y_L, Z_L - coordinates of the perspective centre or focal point w.r.t. object plane coordinate system

x_j, y_j - coordinates of the projection of an object control point j on the image plane

Ω, θ, Φ - pitch yaw and roll of camera axes w.r.t. the object axes

f_e - camera effective focal length

m_{11} - $\cos(\Phi)\cos(\Omega)$

m_{12} - $\sin(\theta)\sin(\Omega)\cos(\Phi) + \cos(\theta)\sin(\Phi)$

m_{13} - $\sin(\theta)\sin(\Phi) - \cos(\theta)\sin(\Omega)\cos(\Phi)$

m_{21} - $-\cos(\Omega)\sin(\Phi)$

m_{22} - $\cos(\theta)\cos(\Phi) - \sin(\theta)\sin(\Omega)\sin(\Phi)$

m_{23} - $\sin(\theta)\cos(\Phi) + \cos(\theta)\sin(\Omega)\sin(\Phi)$

m_{31} - $\sin(\Omega)$

m_{32} - $-\sin(\theta)\cos(\Omega)$

m_{33} - $\cos(\theta)\cos(\Omega)$

B_{ij} - are the partial derivatives of the collinearity equations calculated at the initial estimate point.

(x_{pi}, y_{pi}) - coordinates of a shape's projection point on the image plane.

CHAPTER 1

INTRODUCTION

Robotics continue to play an increasingly important role in automated manufacturing. The incorporation of non-contact vision systems in robotic stations increases the flexibility with which parts can be presented to them. Early machine vision systems were capable of locating parts on a conveyor at a known distance from the camera. Systems such as those developed by Fahim and Cheng [1.1], Capson [1.2] and Holland, Rossol, and Ward [1.3] report the part's position and rotation within the camera frame: however these were strictly two dimensional measurements. In order to integrate vision systems in a dynamic work environment such as those utilizing automated guided vehicles (AGV) or robots required to place and remove parts hung from an overhead conveyor, vision systems capable of providing three dimensional information are required.

Development in three dimensional part position determination and tracking has proceeded along a number of courses. The system described by Williams [1.4] considers a stationary three dimensional environment and moves the camera into known positions to "snap" individual photographs. Selected feature points in the two photographs are then matched and triangulation is used to determine the distance to the feature points. Stereo vision systems use a similar technique. Mitiche et al [1.5] also

used a moving camera to estimate the position of a rigid body by considering a small number of corresponding points in the two projections of the scene. This method assumes that the point correspondences are known. The technique proposed by Mutch [1.6] uses two cameras and a sequence of stereo images to determine the movement of objects in the scene. This is referred to as stereoscopic motion and differs from the traditional stereo approach because it does not employ single snapshots. This technique derives change vectors for points in the successive images and uses these to detect movement of rigid bodies.

Another approach described by Huang [1.7] assumes a dynamic three dimensional environment and determines the motion between successive images by matching corresponding pixels in consecutive frames to develop a flow diagram of the sequence. From the flow diagram an attempt is made to discern the movement and identity of rigid bodies. Jain and Sethi [1.8] also proposed a technique which uses image flow. A notable feature of their approach is that rigid bodies are not assumed. The point correspondence problem is solved by assuming a smoothness of motion in the detectible image feature points (in this case corners). Several successive images are used and points are joined according to the smoothness of the motion they define. Groups of points displaying similar motion are then assumed to belong to the same object.

Augusteijn and Dyer [1.9] found the three dimensional position of an object by using geometric characteristics of the object. Their paper describes a technique to determine the orientation of a planar shape in

space through an iterative process in which the feature points of a known shape are matched to those derived from a shape in the image. The main drawback to this technique is the assumption that the projections of the shapes are orthographic whereas the shapes contained in a camera image are generally perspective projections. Figure 1.1 contrasts orthographic

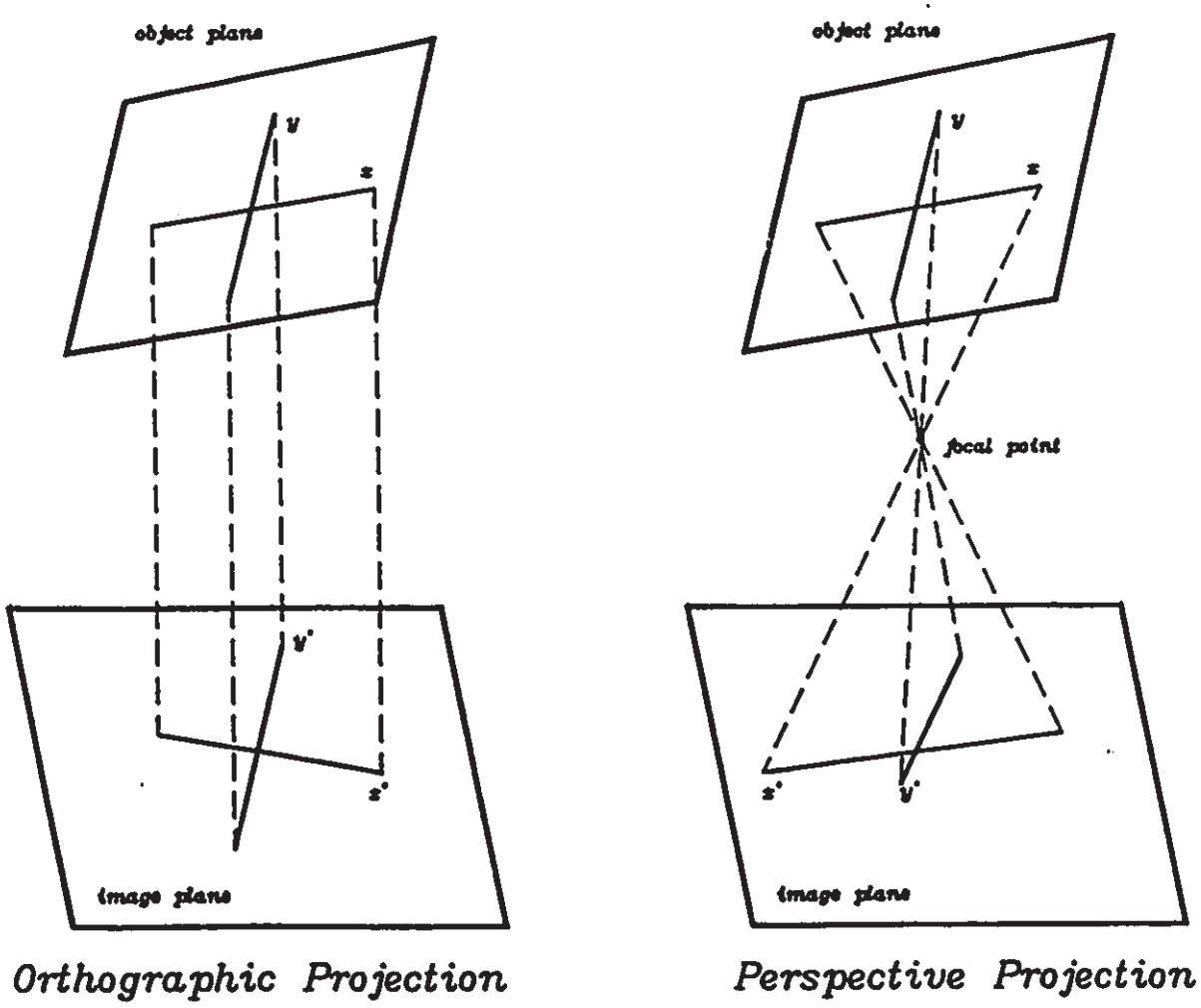


Figure 1.1. Orthographic vs Perspective Projection

with perspective projections. The orthographic projection of the shape will be the same regardless of its distance from the camera; thus the use of orthographic projections results in the loss of information about the z direction.

Perspective object projections are assumed by Horaud [1.10]. Using objects which contain straight edges and planar surfaces, this technique matches image junctions formed by intersecting edges with those of previously learned shapes. Once a possible correspondence is found, the object's estimated position is used to predict where other junctions should be found in the image. If enough corresponding junctions are found, the object position is considered correct, otherwise a different correspondence is attempted.

Another approach which assumes perspective projections is that of Topa and Schalkoff [1.11]. The method employed in this case is limited to finding the pitch and yaw orientation of a plane and accomplishes the task by projecting a number of circles or ellipses onto the plane and then determining orientation from the deformation of the projected image. For example, if circles are projected onto a pitched surface the characteristics of the ellipses in the resulting image are used to determine the pitch. An illustration of the projector camera setup is given in Figure 1.2 where the camera and projector are modelled with parallel optical axes.

Gordon et al [1.12] also use a projection technique to determine the position of an object. In this case a laser light line is projected

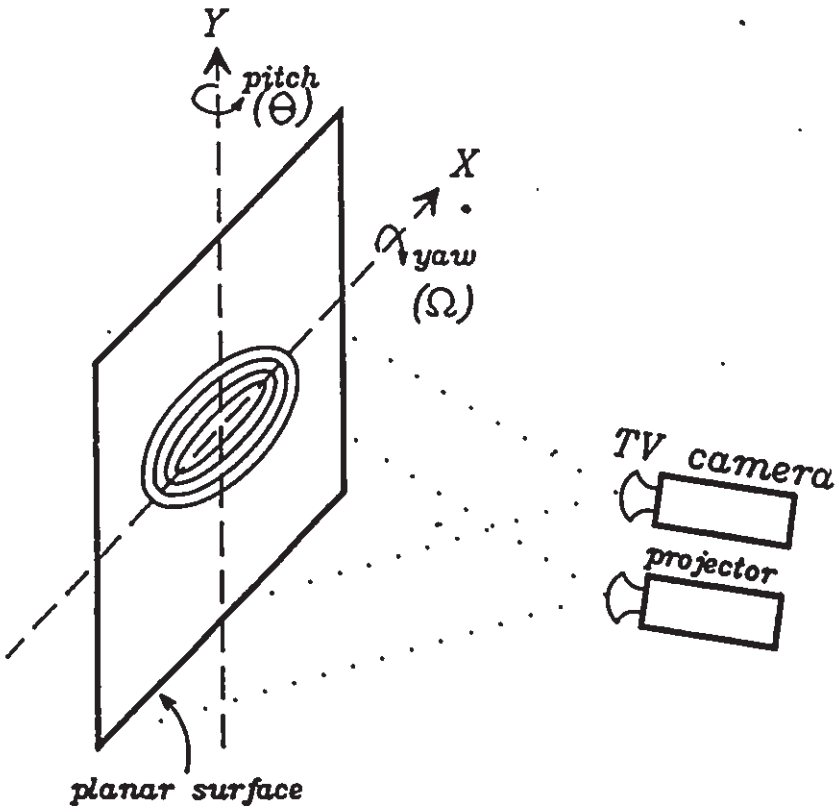


Figure 1.2. Illustration of Camera Projector Setup for Technique Described in [1.11]

onto a part whose position is approximately known. By processing a single image in this highly constrained environment the reflected line of light is used to accurately determine the position of the object.

A fast (30 Hz) video photogrammetry system for robot guidance in a dynamic environment has been developed at the National Research Council (NRC) of Canada and is described by Pinkney and Perrat [1.13],[1.14] and Kratky [1.15]. This system uses a target comprising four dots attached to

the object to be tracked as depicted in Figure 1.3. This four dot planar

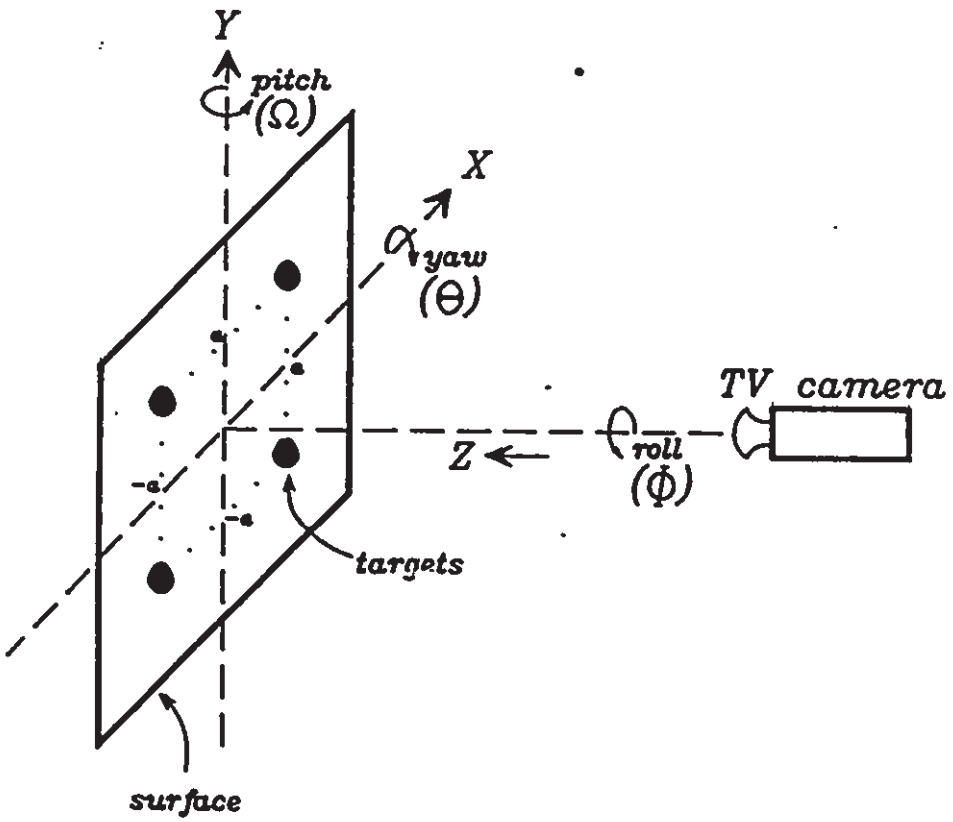


Figure 1.3. Real Time Photogrammetry System Developed by NRC

target simplifies two of the classical problems associated with object tracking. First, the point correspondence problem is solved because the four dots are homogeneous and arranged in a square (this limits the determination of roll to 0 - 90 degrees). Second, since a standard target is used, the derivation of the equations to determine the X,Y,Z position and the pitch, yaw and roll angle from the perspective projection need only be determined once. A set of equations to estimate the three

dimensional target position have been developed using the standard least squares approach for single image resection described by Thompson [1.16] and elsewhere.

An important aspect in the determination of object position and orientation is the measurement of the shape of the perspective projection which changes as the object's pitch and yaw are varied. The 3-D tracking system described by Wallace [1.17] uses the shape of the projection to determine the position of the object (in this case aircraft). This is accomplished by deriving a set of Fourier shape descriptors from the object contour and comparing them to a data base containing sets of descriptors for several perspective projections of the object. The actual orientation in space of the object relative to the camera is then estimated by interpolating amongst a set of stored projections.

These examples have been chosen to illustrate some of the current techniques in practice for three dimensional object position and orientation determination. Many other techniques have been described (for example several recent works are listed in a survey paper by Rosenfeld [1.18]). A recurring theme in many systems is the separation of the tracking problem into two tasks; first, the point correspondence problem which involves either matching image points in a series of images or matching image points in a single image to those of a known shape; second, the mathematical analysis necessary to determine position (this involves triangulation for image sequences or stereo images to determine depth and photogrammetric resection for single image solutions).

In the context of the NRC system a complete tracking system for robot guidance consists of 5 stages:

1) The raw image pixels must be classified. This might involve separating object from background, detecting object edges or grouping pixels into colour regions.
2) Objects within the image must be identified and features used for tracking must be isolated. (For the NRC system these were target dots.)
3) A method of fast feature location within the image frame must be used to track at frame rates.
4) The object position must be calculated based on the feature projections.
5) The feedback loop for the robot must be developed.

The scope of this thesis is the three intermediate stages of this process.

This thesis is devoted to a new technique for the determination of the three dimensional position and orientation of planar shapes using four arbitrary corners. Two-dimensional perspective projections of translated and rotated planar shapes are considered on a frame by frame basis from a single camera view. Determining the three dimensional position is performed in two stages. First, descriptors of the image projection of the shape are used to get an initial estimate of the object's position. Next, a set of four shape corners are more accurately located on the projection by positioning feature windows over the estimated corner locations (Figure 1.4). Data within the windows is collected and the position and orientation of the shape are calculated

using the projected corner positions. Operation of the system can be divided into four tasks:

1) Training phase: during this task a shape is taught to the system. This includes computing sets of shape descriptors for the shape at a large number of orientations (i.e. for various pitch and yaw angles). The corners to be used for shape tracking are also selected at this stage.

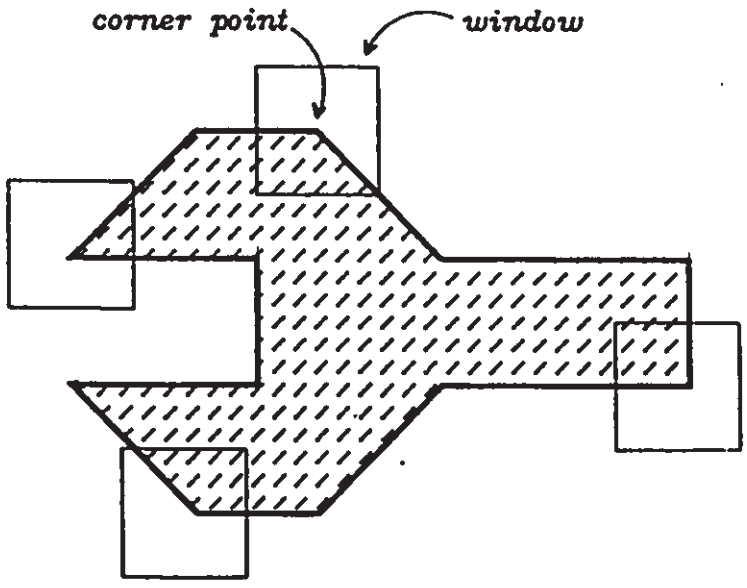


Figure 1.4. Example of Feature Window Placement on Shape Corners

2) Feature lock-on: When a shape is presented to the system at an unknown orientation, the shape descriptors are computed and compared to the sets of descriptors computed during the training phase. The closest match is found and a set of feature windows is placed on the object at positions where the corners are expected.
3) Feature window processing: Data is collected for the areas within the feature windows to more accurately locate the projections of the corners.
4) Photogrammetric solution: The corner projections are used to compute the three dimensional position of the object.

Within this context, contributions have been made in the following areas:

1. Feature Lock-On System

A distinguishing feature of this system is the use of planar shape descriptors in order to locate and lock-on to object features. A number of techniques to derive descriptors from shape contours are investigated to find a suitable method to estimate the location of appropriate features to be used in tracking. A suitable technique employing Fourier descriptors is used to determine the pitch, yaw and roll of the observed shape.

2. Fast Corner Location Algorithm

When the approximate positions of the corner features have been determined, windows are placed at the estimated locations and a fast corner location algorithm is employed. A technique which exploits similar

data as that collected for locating the centres of target dots for the NRC tracking system has been developed. This technique is suitable for real-time implementation.

3. Three Dimensional Position Determination

A set of equations have been developed to determine the three dimensional position of a planar shape given four non-collinear projection points in a single camera view. Earlier techniques employed a set of non-linear equations which were solved with a least squares estimation technique. The new technique uses a set of 11 linear equations together with one non-linear equation which can be solved directly to determine the three dimensional position. This analytic solution has been verified experimentally.

Chapter 2 of the thesis constitutes a description of the two dimensional shape identification process employed. A brief description of a fast contour derivation technique is included as well as a description of hardware developed to perform this task. This is followed by a review of a number of shape descriptors. Through this review process a number of advantages gained by Fourier descriptors are illustrated. The results from a number of experiments using Fourier descriptors are reported and a number of the descriptor characteristics are demonstrated.

Implementation of the tracking system is described in Chapter 3. Details of the projection identification and feature lock-on technique are provided together with experimental results. The window placement

strategy as well as the fast corner location algorithm are described. This is followed by an outline of the least squares solution for position determination as well as a detailed description of the new analytic solution. Some of the practical aspects of a tracking system are also discussed in this chapter. These include the effects of lighting and the need for fast algorithms.

Chapter 4 provides the results of experimentation to evaluate tracking performance. First the analytic solution is compared with the NRC least squares solution developed for a square four dot target. Further experimental results are shown for several arbitrary planar patterns, followed by results obtained from tracking using corner features. The effects of camera specifications and distance are also investigated.

Conclusions and comments for further work are provided in Chapter 5 including an analytic solution for four point non-planar patterns.

Chapter 2

CONTOUR BASED SHAPE PROJECTION RECOGNITION

Identification of a shape projection within an image can be achieved using the shape contour; the method employed to extract the contour must be fast to operate at video rates. An aspect of a suitable algorithm that makes it possible to achieve this goal is that it be sequential in operation (i.e. the information derived from the image should be processed as it is received in raster fashion). A number of sequential algorithms have been developed by Rosenfeld and Kak [2.1], Cederberg [2.2], and Agrawala and Kulkarni [2.3]. A hardware implementation of the algorithm developed by Capson [2.4] was carried out in order to evaluate its suitability for high speed implementation. This implementation is further described in section 2.1.

Once the shape contours in a scene have been acquired the next problem to be solved is that of shape classification. Since connectivity analysis provides a polygonal approximation of the object boundaries, descriptors derived from this boundary are preferable. A suitable set of shape descriptors must satisfy four requirements as set forth by Ma, Wu and Lu [2.5]:

1) The descriptors should be independent of translation, rotation, scale of objects and cyclic shift of the starting point.

2) The distinction between the descriptors of objects of different shapes should be as large as possible.
3) The number of descriptors used in classification and recognition should be as small as possible.
4) The computational time and the storage requirements should be as short and small as possible respectively.

For object tracking, two additional properties are desirable:

1) The descriptors should be derived from a unique sampling of the object contour thereby ensuring non-ambiguous results.
2) The ability to determine the rotational orientation of the object would be desirable.

A number of different shape descriptor schemes have been used in the past. The simplest of these calculate shape statistics such as area, perimeter, perimeter-squared/area etc. These statistics are quickly calculated but fail to give unique accurate results for a wide range of shapes. Other methods use a series of statistics such as moments or Fourier descriptors which can be extended to achieve the degree of shape discrimination required. These include moment calculations, Walsh transform descriptors, Rapid transform descriptors and Fourier descriptors. Each of these techniques and their related advantages and disadvantages are further described in section 2.2.

Fourier descriptors were chosen for the various advantages they offer when compared to the other methods. Justification for this selection and the results of a number of experiments carried out using Fourier

descriptors is included in section 2.3.

2.1 High Speed Connectivity Analysis

A vision workstation has been constructed to implement and test the connectivity analysis algorithm. It consists of a CCD camera, a pixel digitizer, specialized hardware (including the TMS32010 processor (DSP)) and a personal computer (PC). The PC provides a facility for program development, permanent storage, and high-level language algorithm development. The specialized hardware permits fast execution of developed programs on the TMS32010. Further developments in DSP processors since this system was implemented would result in improvements in execution times (for example use of the TMS320C25 with its hardware extensions and increased operating speed is expected to improve overall performance by 3 - 4 times). However, the TMS32010 system provides a useful benchmark.

The system implemented [2.6] utilizes the connectivity algorithm developed by Capson because of its data structure and improved efficiency with respect to other known algorithms.

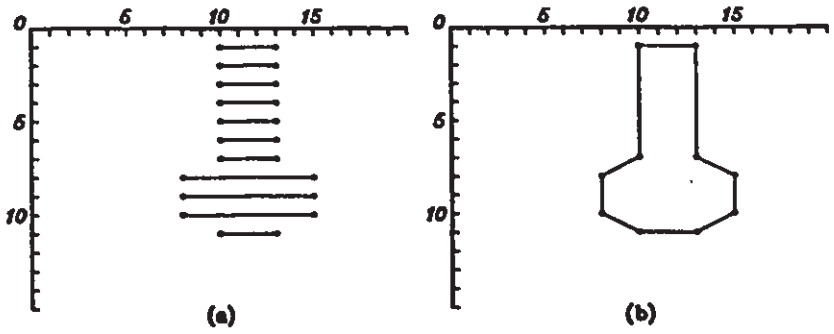
2.1.1 Connectivity Algorithm

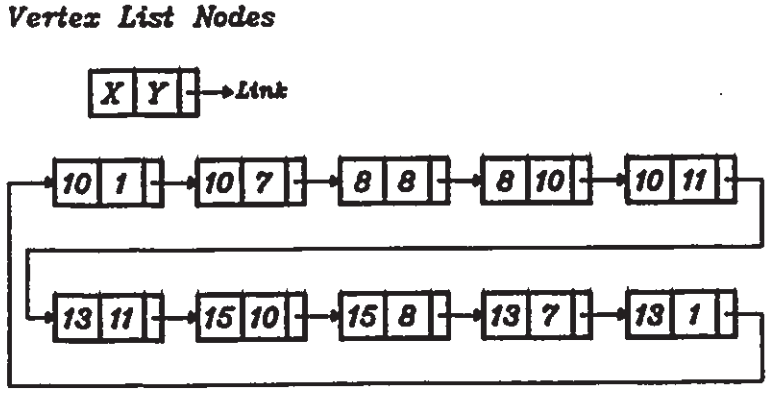
The connectivity analysis algorithm extracts polygonal approximations of the objects and their holes from a raster image. It assumes that pixels have been classified previously as either background or object. In its simplest form, when illumination is uniform and the scene has high contrast, this is achieved using a global threshold. For

uneven illumination low-contrast images, objects may be separated from background using local thresholds or edge operators as described in the papers of Weszka [2.7], and Rosenfeld [2.8]. Linked lists of the vertices of the polygons approximating each object and their associated holes are produced. No restrictions are imposed on the complexity of the image: any number of objects and holes may be featured in the frame and these may be intertwined, or situated within holes of other objects, or intersect with the frame edges.

The input to the algorithm consists of the coordinates of the transition that define segments in an image (Figure 2.1 (a)). These transition points are produced in real time during each line of the raster. The efficiency of the algorithm stems from the design of the data structures that are used to represent the image as it is being processed. Three structures accomplish this task: the active list; the complete list; and the vertex list.

Whenever any portion of an object appears in the current raster line it is deemed "active", e.g., objects 3 and 4 at line 90 of Figure 2.2 (a). The active-list data structure is a linked list of nodes that represent each active object and is updated at each raster line. When an object is first detected in the raster, a new node is entered in the active list. Each node contains a pointer to the boundary coordinates found on the previous raster line (points w,x,p, and q of Figure 2.2 (a)) and a pointer to subsequent nodes of the active list. This is shown in Figure 2.2 (b) for line 90 of the image in Figure 2.2 (a). Then, as the transition points of the current line are obtained (points y,z,r, and s)





(c)

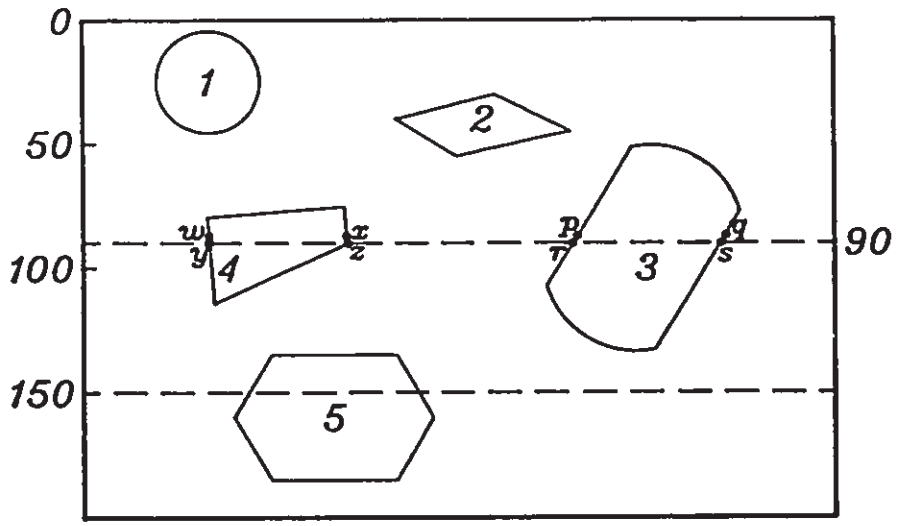
Figure 2.1. a) Sample of transition points derived from a simple image.
b) Polygonal approximation of object in (a).
c) Vertex list of object in (a).

a contour of each object is constructed.

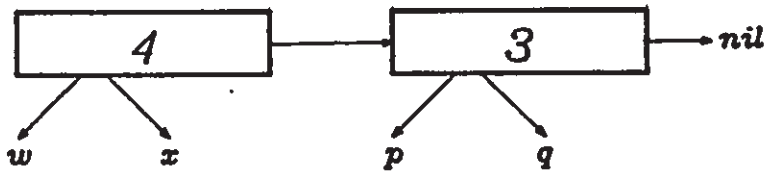
When an active-list node no longer intersects with the current line, that object is deemed "complete", its corresponding node in the active list is removed, and an entry is made in the complete-list data structure. For example, at line 150 of Figure 2.2 (a) the complete list would contain the four elements of Figure 2.2 (c) representing the completed object #'s 1,2,3, and 4. Once an object is entered in the complete list, its contour is available and may be processed even if the entire raster scan has not been completed. At the conclusion of the raster scan the number of objects contained in the image is the number of entries in the complete list.

The vertex list contains linked nodes of x,y coordinates that comprise the polygonal approximation to the contours. The representation of the object of Figure 2.1 (a) is that of Figure 2.1 (b) and (c). As the raster proceeds, a separate vertex list is constructed for each object and hole contour.

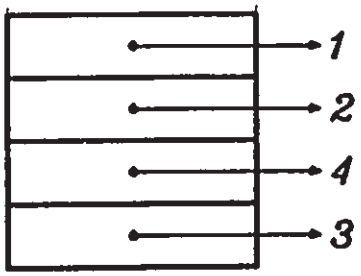
Transition points from each raster line are appended to the appropriate vertex list by comparing the overlap conditions of the segments of the current and previous lines. Referring again to Figure 2.2 (a), the points y and z are known to be connected to the same contour as points w and x because the two segments defined by these endpoints overlap. This is depicted in Figure 2.3 (a), which also shows the updating that occurs in the active list. After updating, entries would be made in the vertex lists of the two objects connecting w to y, z to x and p to r, s to q. When checking overlapping segments two other cases arise:



(a)



(b)



(c)

Figure 2.2. Example of One Image Frame.
a) Sample image with five objects.
b) Active list at line 90.
c) Complete list at line 150.

1) a segment on the current line overlaps two or more segments from the previous line (Figure 2.3 (b)); and
2) two or more segments on the current line are overlapped by one segment on the previous line (Figure 2.3 (e)).

Case 1 indicates either a merging of two or more objects that until this point were taken as separate objects, or the completion of a hole or holes within an object. In case 2, the object is splitting, and this may be the start of a hole. These situations are represented in the active list with pointers. The complete format of an active list node is shown in Figure 2.3 (f). Whenever a split is encountered (case 2) a new node is inserted in the active list and joined to the split node via the "forward" and "back" pointers (Figure 2.3 (e)). Whenever a merge occurs (case 1) The values of these pointers are checked; if the forward and back pointers are set (indicating a previous split), then the object becomes a hole by setting the hole and last hole pointers (Figure 2.3 (d)). Otherwise, a merge of two previously separate objects is taking place and a node is deleted from the active list (Figure 2.3 (c)).

A sample object and its active list progression at several lines is given in Figure 2.4. The structure of the active list is seen to reflect the structure of the object. Consider for example the configuration at line i of the raster scan: it appears that there are two distinct objects 1 and 2 with object 2 having been split due to the hole 3. At line j, object 3 merges with 2 and becomes a hole in 2 as it is

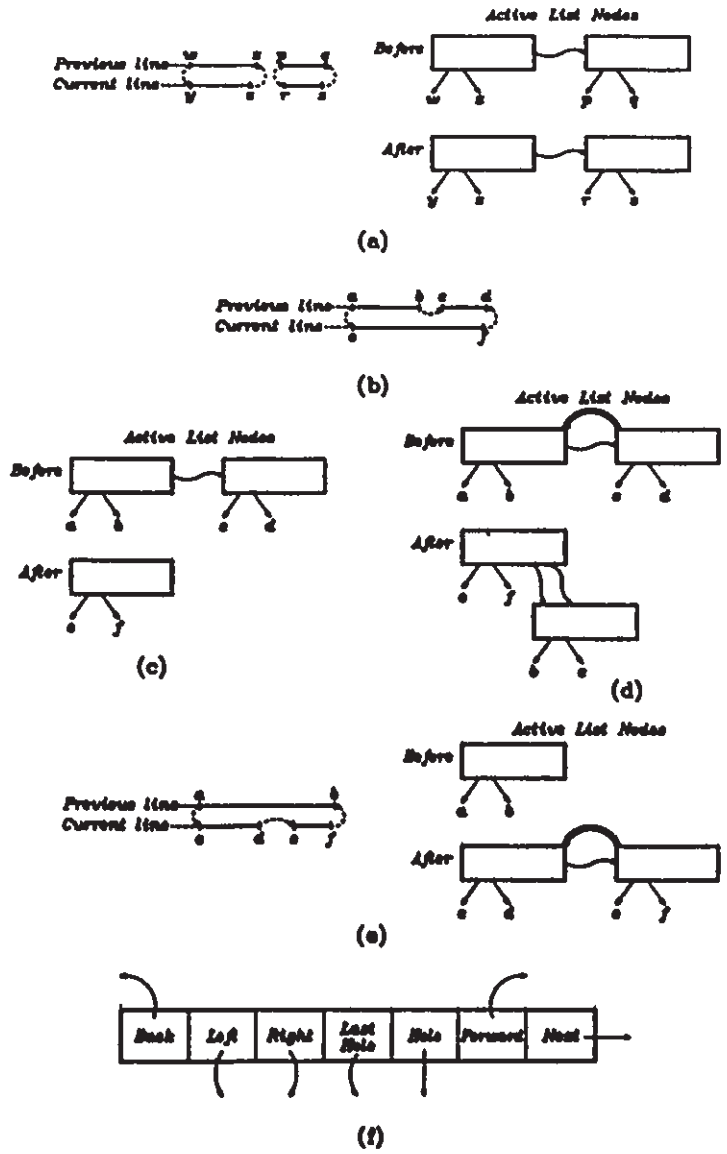


Figure 2.3. Overlap Conditions For Successive Raster Lines.

a) Single segment overlaps previous single segment.
b) Single segment overlaps two previous segments.
c) Active list if two previously unconnected segments are merged.
d) Active list if two previously split objects are merged.
e) Two new segments overlap with a single previous segment.
f) Complete format of an active list node: Back - pointer to object from which object was split; left - pointer to left vertex point; right - pointer to right vertex point; last_hole - pointer to last hole in hole list; hole - pointer to first hole in hole list; forward - pointer to object previously split; next - pointer to next object in object list.

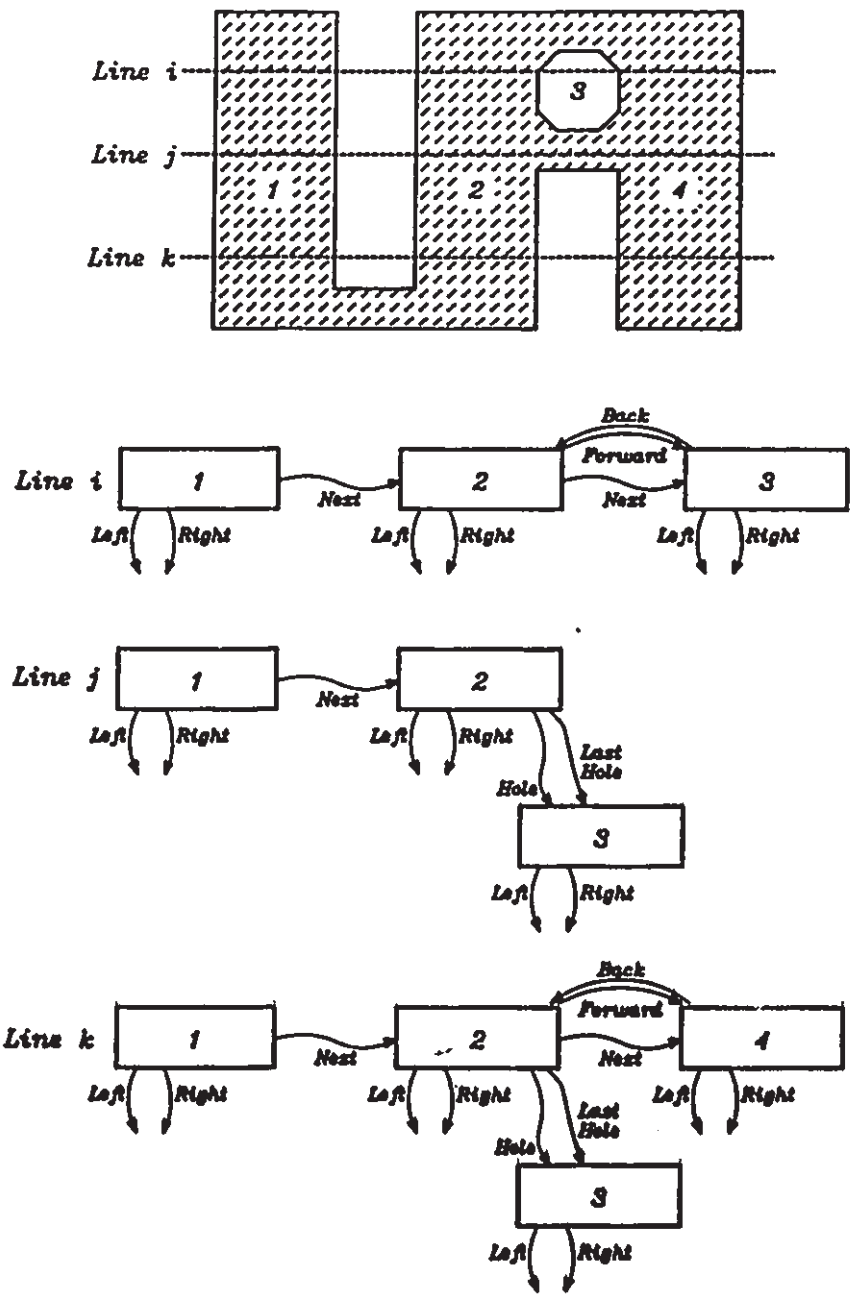


Figure 2.4. Active List Progression for a Sample Object.

known that 3 previously split from 2. At line k, object 2 has split resulting in object 4. When objects 1 and 2 merge, object 2 will be deleted because it was never connected to object 1. The object 2 hole will be appended to the object 1 hole list.

The algorithm is a completely sequential process, yielding contours of segmented images during the raster scan. Efficient execution is attained by manipulating pointers in the dynamic data structures, thereby making it suitable for high speed implementation.

At the end of the frame the active list is empty and the complete list points to the objects within the frame. These polygonal approximations can then be used to identify the objects.

2.1.2 Hardware Implementation

The hardware can be divided into three modules, each implemented on a separate circuit board (Figure 2.5): a fast silhouette breakpoint device capable of producing the coordinates of the transition points in the image (run length segment encoder (RLSE)), an intermediate FIFO buffer to store the breakpoints in real time and supply them as requested (queue buffer) and a fast microprocessor-based (DSP) unit that produces the polygonal approximations of the objects within the image (connectivity analysis processor).

The RLSE receives digitized images from the camera together with a series of control signals (horizontal sync, vertical sync, and pixel clock). This image is binarized through a global thresholding operation, the threshold being set by a controlling PC. The board produces

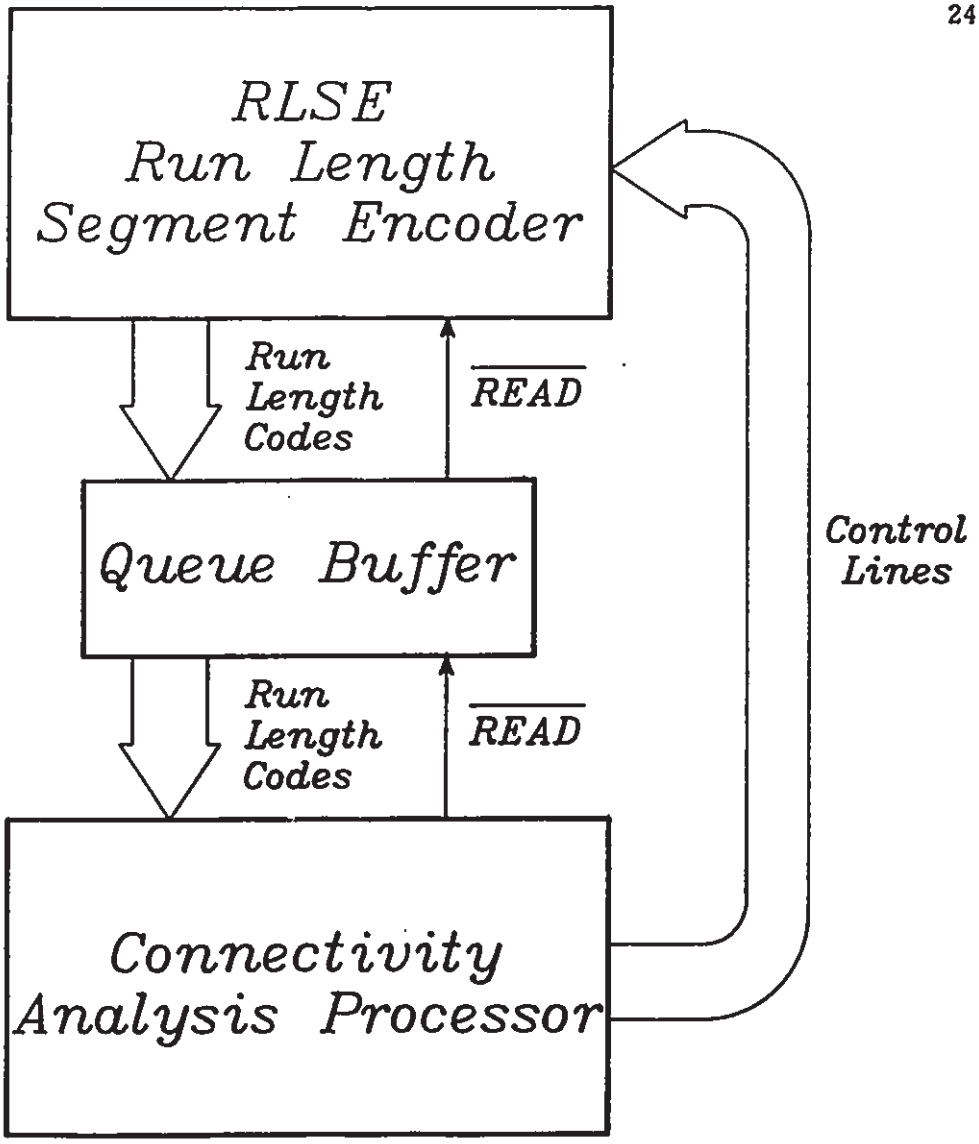


Figure 2.5. Block Diagram of Connectivity Hardware

coordinates of the transitions from background to object and vice versa for each line in the image in real time.

The queue buffer stores the breakpoints at video rates and accesses them at the rate at which the connectivity analysis processor requests them. The FIFO operation of this buffer eliminates the need for the RLSE board and the connectivity analysis processor to be synchronized (i.e. the connectivity processor is not required to read each line of

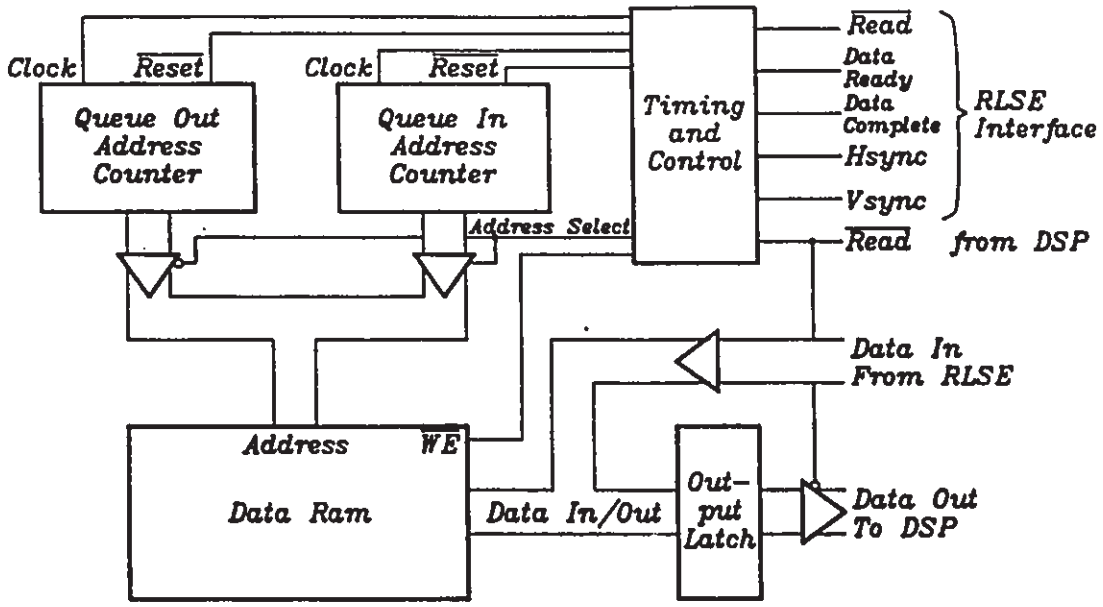


Figure 2.6. Block Diagram of Queue Buffer

breakpoints from the RLSE board as it is produced). The board consists of 8K x 16 bits of RAM (expandable to 64K x 16 bits) for the storage of breakpoints. A block diagram of the device is provided in Figure 2.6.

The connectivity processor board uses the TMS32010 digital signal processing chip as the central processing unit along with four I/O ports; two ports are used for storage and retrieval of connectivity analysis data from a 16K x 16 bit RAM; one port is used for communication with the PC; and one further port is used for communication with the RLSE and the queue buffer. A block diagram of the processor board is given in Figure 2.7.

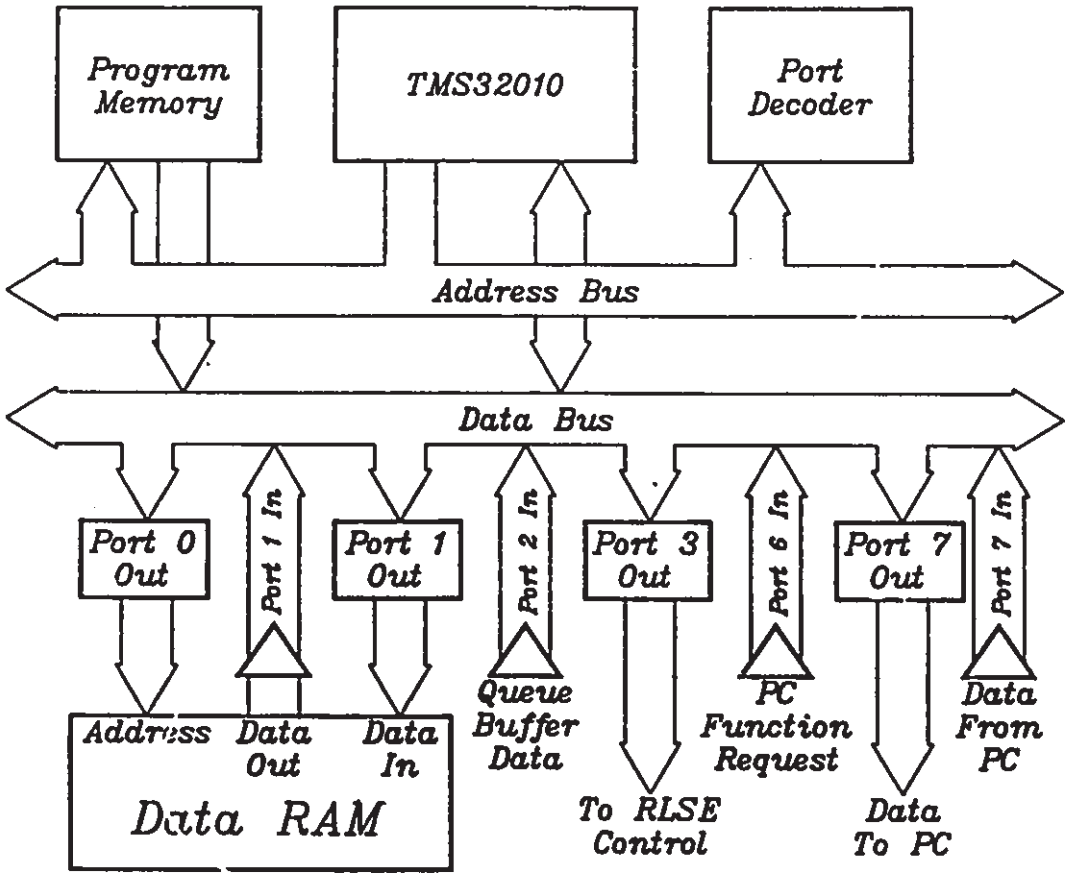


Figure 2.7. Connectivity Analysis Processor Board Block Diagram

2.1.3 Software

The software controlling the connectivity analysis processor can be divided into a number of subtasks. The processor is programmed to perform the connectivity analysis, and communicate with the PC. The PC requests vision functions to be performed by the TMS32010 through the communications link; these include an analyze frame command (i.e. grab,

threshold and perform connectivity analysis), a download vertex points command, and a set threshold command.

Two functions by which the connectivity processor controls the RLSE board are: start data collection, and set threshold. The start data collection command causes the board to collect the breakpoints for the frame starting at the next vertical sync pulse. As breakpoints are accumulated they are dumped into the queue buffer automatically. The threshold command sets the grey level at which the object and background are separated.

The PC software includes an interface that allows the user to execute the vision functions performed by the connectivity processor. It also allows the user to display graphically the vertex points resulting from the connectivity analysis. A facility to store data lists (i.e. the complete list, the active and the vertex list) in disk files on the PC for future use was also implemented.

2.1.4 Results of High Speed Connectivity Implementation

The polygon extraction algorithm and the run length code generator were implemented previously in software on an Intel SBC86/20 single board microcomputer. To produce run-length segments, it was necessary to search the entire frame for breakpoints, and record the x,y coordinates of each breakpoint found. The time required, therefore, depended on the number of breakpoints found, and ranged typically from 1 to 2 seconds. The hardware implementation allows this time to be reduced to a fixed 60ms due to the real time capability of the RLSE board.

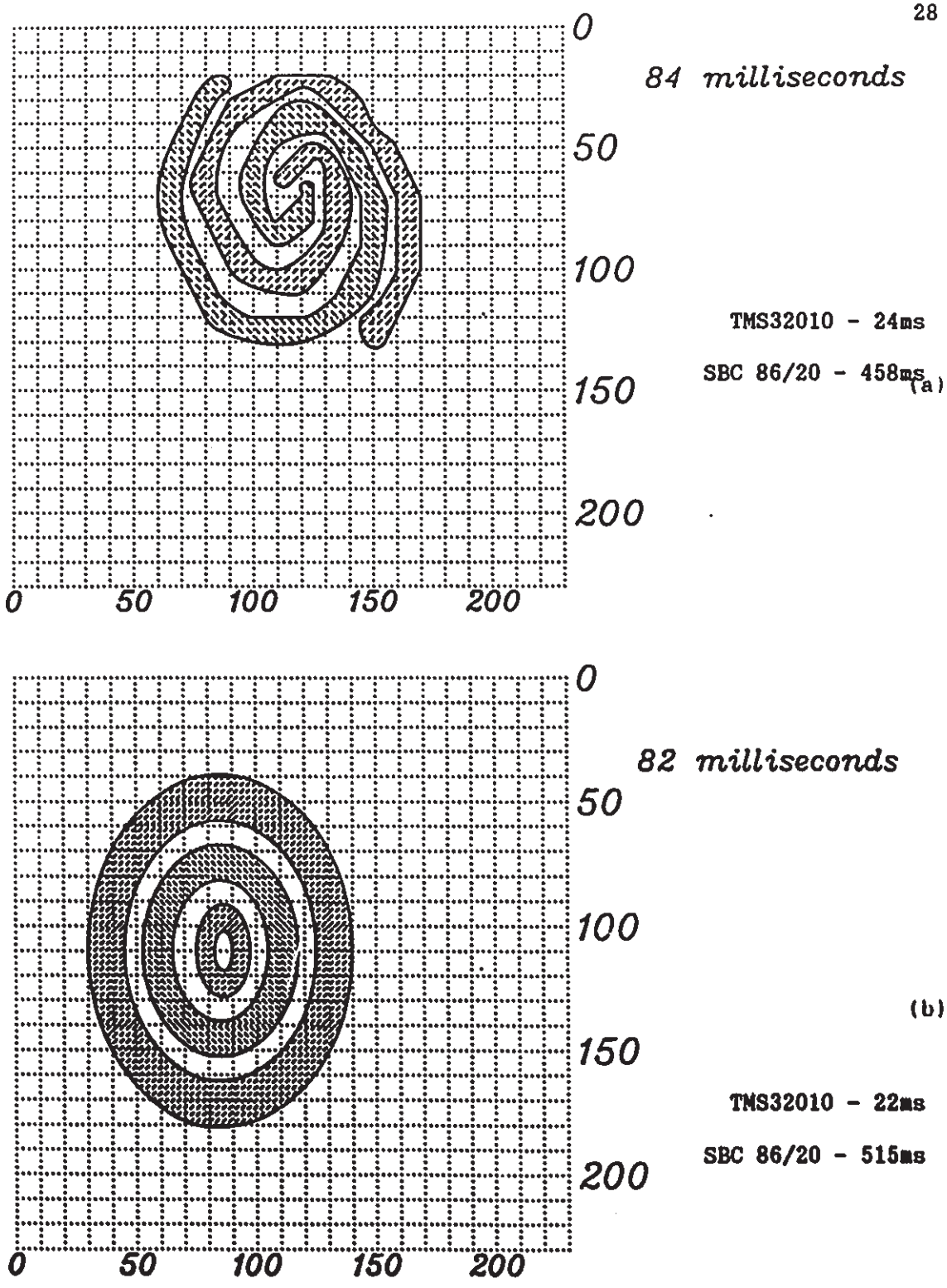


Figure 2.8. Hardware Connectivity Analysis Performance Examples

A number of boundary extraction examples that had previously been used to test the software on the SBC86/20 were chosen to test the DSP based system to compare its performance to that of its predecessor. Figure 2.8 (a) and (b) shows the sample images from the vertex lists provided by the DSP after executing the connectivity program. In each case the time required to grab the image and perform the connectivity analysis is displayed in the upper right-hand corner. From this the fixed 60ms time delay to get the run length codes is subtracted to obtain the time required for connectivity analysis. The resulting connectivity analysis time for the DSP and for the SBC86/20 implementations are also displayed; a speed increase of approximately 20 times has been achieved.

2.2 Shape Identification

The shape identification techniques can be divided into two classes, the moment calculation method and the transform methods. The moment calculations use information about the entire shape or the entire contour whereas the transform methods employ a sampling of the contour. The moment calculation technique is first described. Different methods of contour sampling are then described and the transform techniques employing these methods are compared.

2.2.1 Moment Calculations

Given a polygonal object as depicted in Figure 2.9, the moments are defined in equation 2.1:

$$M_{mn} = \sum_i \sum_j x_{im} * y_{jn} * f(x_i, y_j) \qquad [2.1]$$

where the x_i and y_j are the (x,y) coordinate of each pixel enclosed by the boundary and the function f(x,y) is 1 for all pixels within the boundary and 0 elsewhere. The number of moments used to describe the objects is a function of the desired accuracy of the descriptor set. A larger number of descriptors implies a greater ability to discern objects. The first

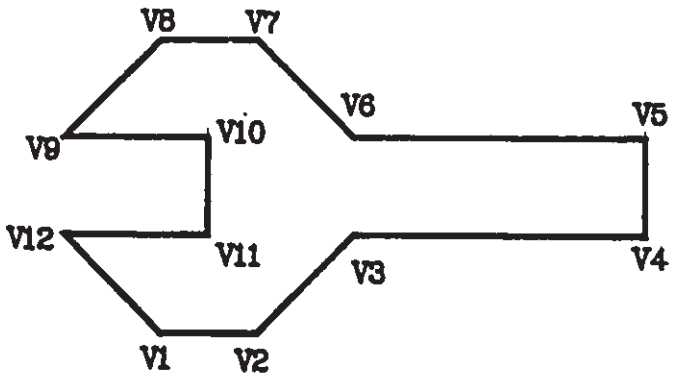


Figure 2.9. Example Polygonal Shape

three moments of the shape can be used to normalize the other moments with respect to size and translation within the image plane. The first moment M_{00} is the area of the object. The next two moments M_{10} and M_{01} are normalized by dividing them by the object area to provide the centre of area of the shape.

Thus the first three statistics are defined as:

$$M_{00} = \sum_i \sum_j f(x_i, y_j) \qquad \text{Area}$$

$$M_{10} = \sum_i \sum_j [\, x_i * f(x_i, y_j)] \,/\, M_{00} \qquad x_c$$

$$M_{01} = \sum_i \sum_j [\, y_j * f(x_i, y_j)] \,/\, M_{00} \qquad y_c$$

Division of each subsequent moment by M_{00} results in normalization with respect to size and movement of the origin to (x_c, y_c) results in normalization of subsequent components with respect to translation.

Moments can be normalized with respect to rotation by rotating the points by an angle which can be determined using the next three moments. Specifically, if M_{11} is set to 0 and ($M_{20} >= M_{02}$) and ($M_{30} >= 0$) then subsequent moments can be compared without regard to rotation. Setting M_{11} to 0 ensures that the object is uniquely oriented with respect to an axis. Ensuring that $M_{20} >= M_{02}$ eliminates 90 degree rotations that would result in the M_{11} still being 0. Setting the $M_{30} >= 0$ ensures that 180 degree rotations are handled. M_{11} is set to 0 in the following way:

$$M_{11} = \sum_i \sum_j x_i * y_j * f(x_i, y_j)$$

where:

$f(x_i, y_j) = 1$ for object points

In order to be set to zero the points must be rotated by an angle theta :

Therefore:

$$M_{11}' = \sum_i \sum_j x_i' * y_j' = 0$$

where:

$$[x_i' \; y_j'] = \begin{bmatrix} \sin\theta & \cos\theta \\ \cos\theta & -\sin\theta \end{bmatrix} \begin{bmatrix} x_i \\ y_j \end{bmatrix}$$

$$M_{11}' = \sum_i \sum_j (x_i*\sin\theta + y_j*\cos\theta)(x_i*\cos\theta - y_j*\sin\theta)$$

$$= 1/2(\sin(2\theta)(M_{20} - M_{02})) - \cos(2\theta)(M_{11}) = 0$$

$$2\theta = \text{Tan}^{-1}((2*M_{11})/(M_{20} - M_{02})) \qquad [2.2]$$

If $M_{20} < M_{02}$ then the x and y axis are interchanged. If $M_{30} < 0$ then the X-axis is reversed.

Once the angle of rotation has been found, subsequent moments are calculated in the following way. Each point in the object is first translated to take into account the shift of the centroid to the origin. The points are then rotated by the angle theta. Finally all values are scaled according to the area. It has been shown by Reeves et al. [2.9] that the use of moments as shape descriptors results in a good classification system. When using moments based on object contours as shape descriptors, the individual pixels making up the shape are not available. Instead the object is approximated by a one pixel wide closed polygon of its boundary. In this case a number of simplifying assumptions regarding the contours can be made. The moments for the shape are approximated as follows:

$$M_{00} = \sum_i \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \qquad \text{perimeter}$$

$$M_{10} = \sum_i ((x_{i+1} + x_i)/2) \,/\, \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}$$

$$M_{01} = \sum_i ((y_{i+1} + y_i)/2) \,/\, \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}$$

$$M_{11} = \sum_{i} \sum_{j=1}^{p} (1/m*j + x_i)(m*j + y_i)$$

where:

$m = (y_{i+1} - y_i)/(x_{i+1} - x_i)$ if $x_{i+1} <> x_i$, $y_{i+1} <> y_i$

$m = 1$, $1/m = 0$ if $x_{i+1} = x_i$

$m = 0$, $1/m = 1$ if $y_{i+1} = y_i$

$$p = \text{int}(\sqrt{(y_{i+1} - y_i)^2 + (x_{i+1} - x_i)^2})$$

$$M_{20} = \sum_{i} \sum_{j=1}^{p} (1/m*j + x_i)^2$$

$$M_{02} = \sum_{i} \sum_{j=1}^{p} (m*j + y_i)^2 \quad \text{etc...}$$

Subsequent moments are then calculated using normalized vertex points.

While this system of moment calculation provides an adequate shape classifier, the drawback is the large amount of computation required. For

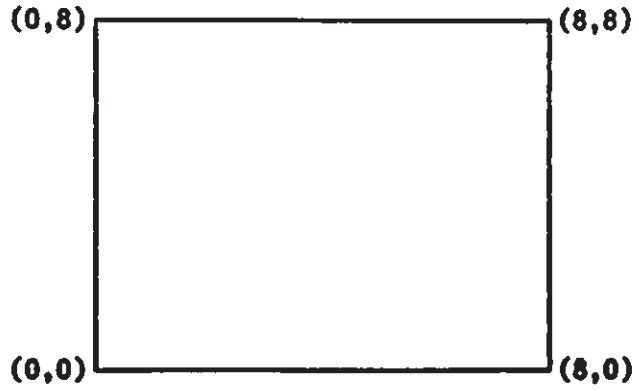


Figure 2.10. A Simple Boundary Example

a simple object as depicted in Figure 2.10 the computations necessary to obtain the first sixteen moments are given in Table 2.1.

	Multiplications	Additions	Square Root	Divisions
M_{00}	8	16	4	-
M_{10}	6	4	-	-
M_{01}	6	4	-	-
M_{11}	96	96	-	4
M_{20}	32	32	-	-
M_{02}	32	32	-	-
M_{30}	32	32	-	-
M_{21}	32	32	-	-
M_{12}	32	32	-	-
M_{03}	32	32	-	-
M_{40}	32	32	-	-
M_{31}	32	32	-	-
M_{22}	32	32	-	-
M_{13}	32	32	-	-
M_{04}	32	32	-	-
M_{50}	32	32	-	-
Totals	500	504	4	4

Table 2.1. Moment Calculations for 16 Moments

These values assume that intermediate results are used to their full advantage (i.e. the calculation of M_{30} results from the summation of single multiplications of each of the M_{20} summation values) and that the square is represented by the optimum number of vertices.

2.2.2 Transform Descriptors

In order to use transform descriptors, the polygonal approximation of the object must be sampled in such a way that it will provide a unique input for the transform. Several methods of sampling the contour have been used including those by Persoon and Fu [2.10], Zahn and Roskies [2.11], Crimmins [2.12], and Chellappa and Bagdazian [2.13]. These methods include, sampling the contour angle at equal distances along the perimeter, sampling the contour radius from the centroid at equal angles, sampling the contour radius from the centroid at equal distances along the perimeter and deriving a sequence of complex values at equal distances along the perimeter. A brief description of each of these methods follows.

A closed polygon can be described by an angular direction function f(x) where x is the distance travelled along the perimeter. Figure 2.11 depicts an object and its corresponding angular direction function f(x). The object contour is navigated in a counter clockwise direction beginning at vertex V_1. In this case the angular direction can be mapped onto the $-\pi$ to 2π range.

To normalize the function it is written:

$$f^*(x) = f((L*x)/(2*\pi)) - x \qquad [2.3]$$

where L is the perimeter of the object

This has the effect of mapping the previous function onto the range $[0,2\pi]$ and normalizing with respect to translations and rotations. The resultant shape sampling function (profile) for the shape in Figure 2.11 is depicted in Figure 2.12.

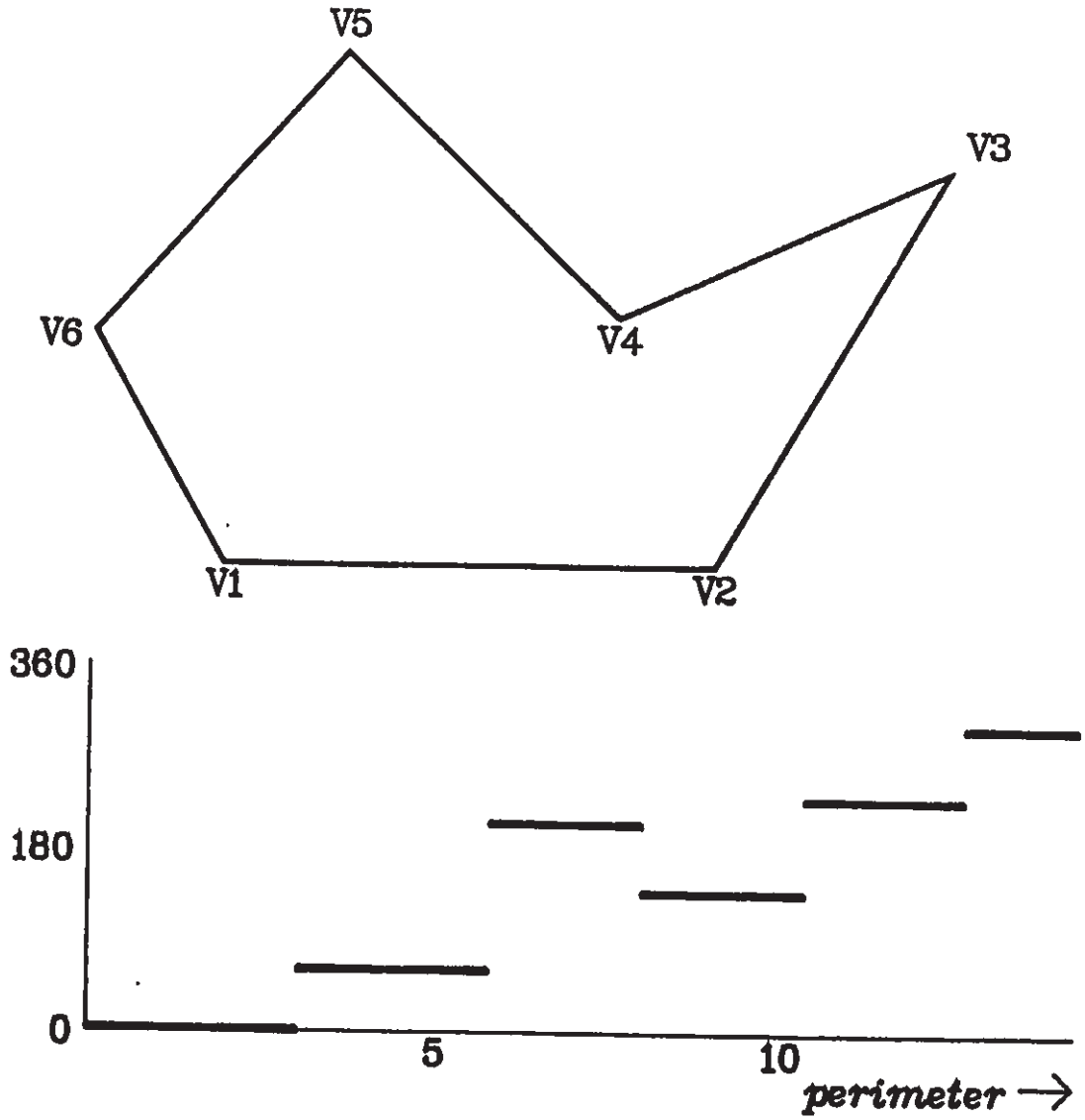


Figure 2.11. An Shape and its Corresponding Angular Direction Function

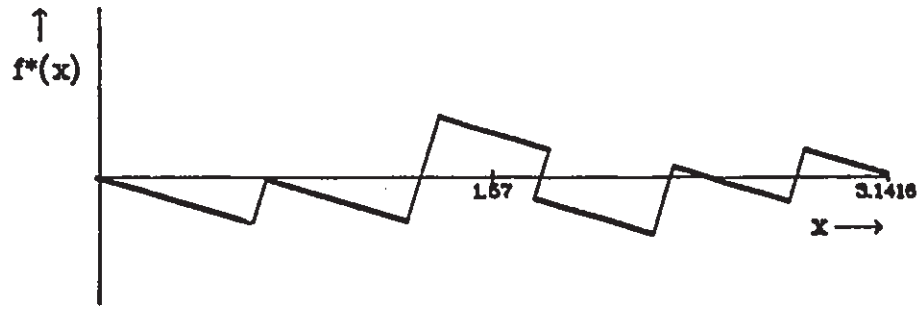


Figure 2.12. Normalized Angular Direction Function for the Shape in Figure 2.11.

Sampling of the contour radius at equal angles is carried out by finding the intersection point of lines drawn from the centroid to the boundary. (An example is given in Figure 2.13 (a)).

The samples derived from the contour are the magnitudes of the radius lines drawn from the centroid to the boundary. The shape in Figure 2.13 (a) would result in the profile given in Figure 2.13 (b). There are two factors that make this method unattractive for practical application. First, finding the intersection points of the boundary and the radius vectors involves significant computation. An intersection point between each of the radius vectors and the object contour would have to be found. This would involve searching through the contour sides to determine if an edge intersects the radius vector. Second, it is possible for a shape contour to have two intersection points with a single radius vector. Since a fixed number of samples are input into the transform multiple intersection points pose a significant problem.

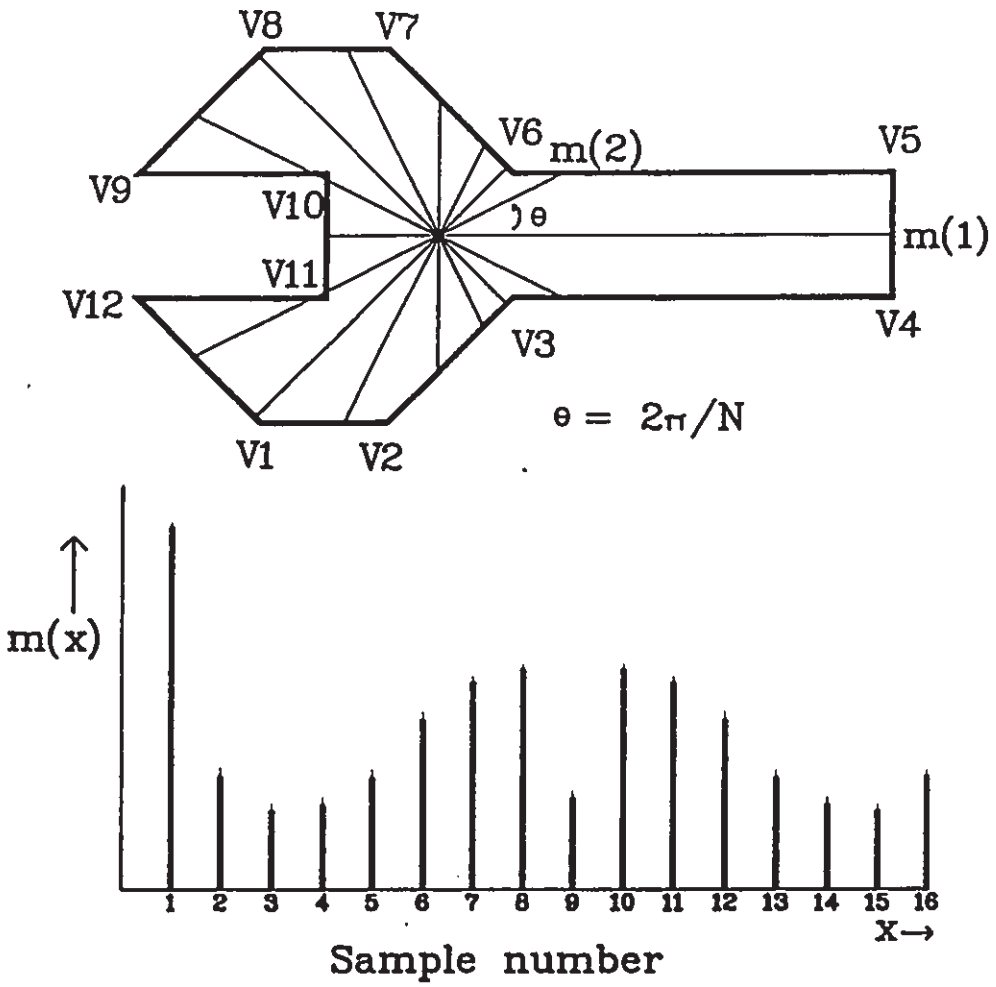


Figure 2.13. Example of Contour Sampling at Equal Angles.

Sampling of the contour radius at equal distances is similar to the previous method except that the perimeter is divided into N equal lengths instead of N angles. An example for the object in Figure 2.13 is given in Figure 2.14.

This method has the advantage that it is not possible to encounter two points at a given sampling angle. However, this method can lead to

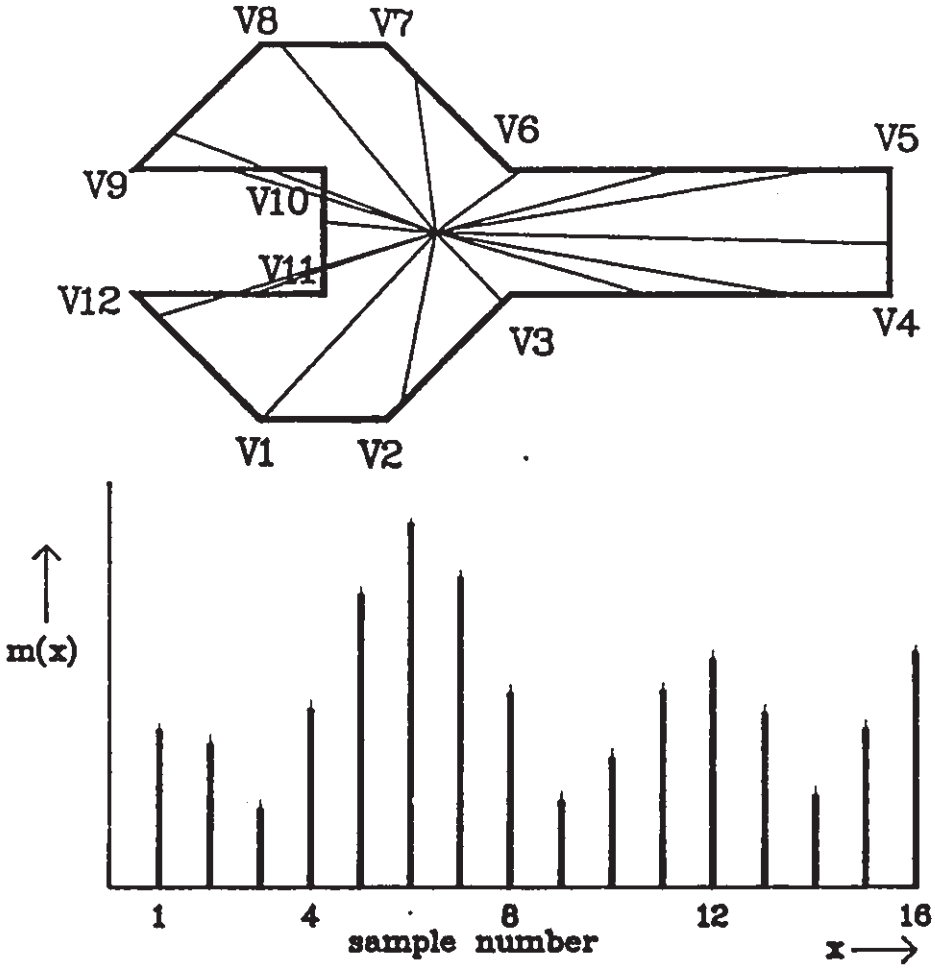


Figure 2.14. Sampling of the Contour Radius at Equal Distances along the Perimeter.

ambiguous results since different shapes can produce the same object profile as depicted in Figure 2.15.

The final method derives a set of complex values from the object contour. To accomplish this the object is mapped onto the complex plane; the centroid of the object is mapped onto the origin. The complex sequence describing the contour is then derived by first dividing the perimeter

into N equal parts, then taking the N points on the perimeter as the sequence. An example of the sequence derived using this method is given in Figure 2.16. This method has the advantage that the sample value has intrinsic magnitude and phase information. This results in unique

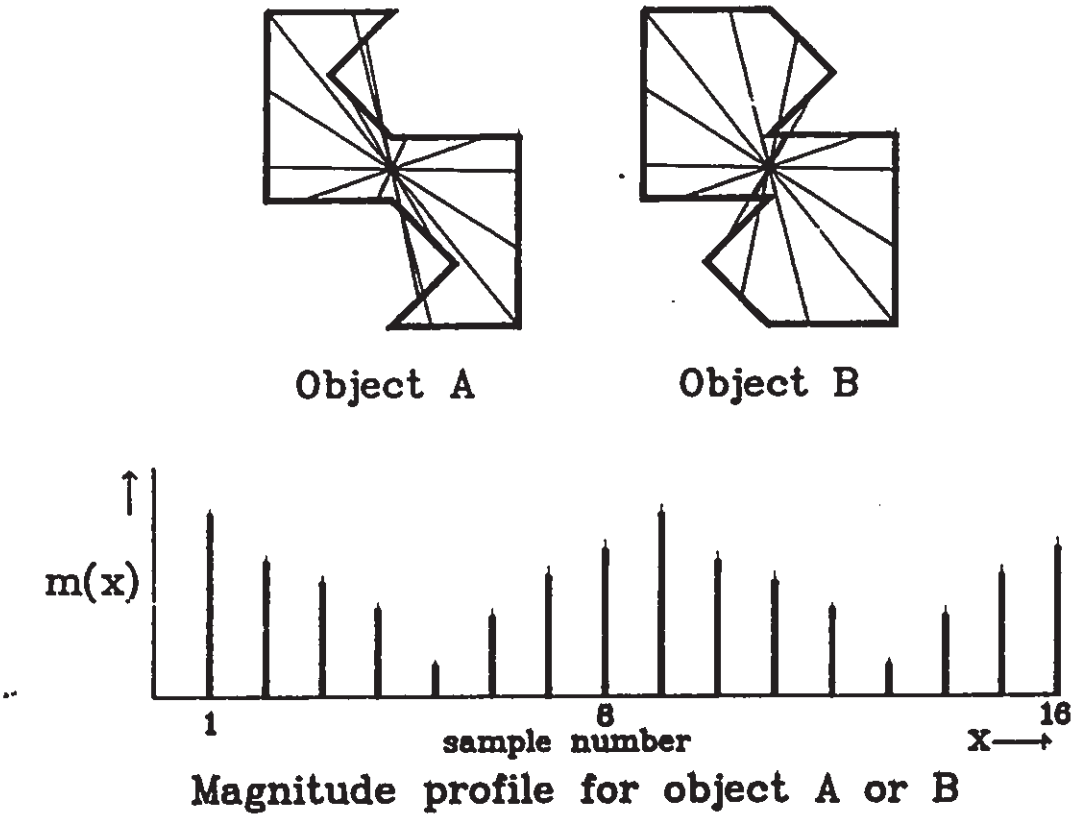
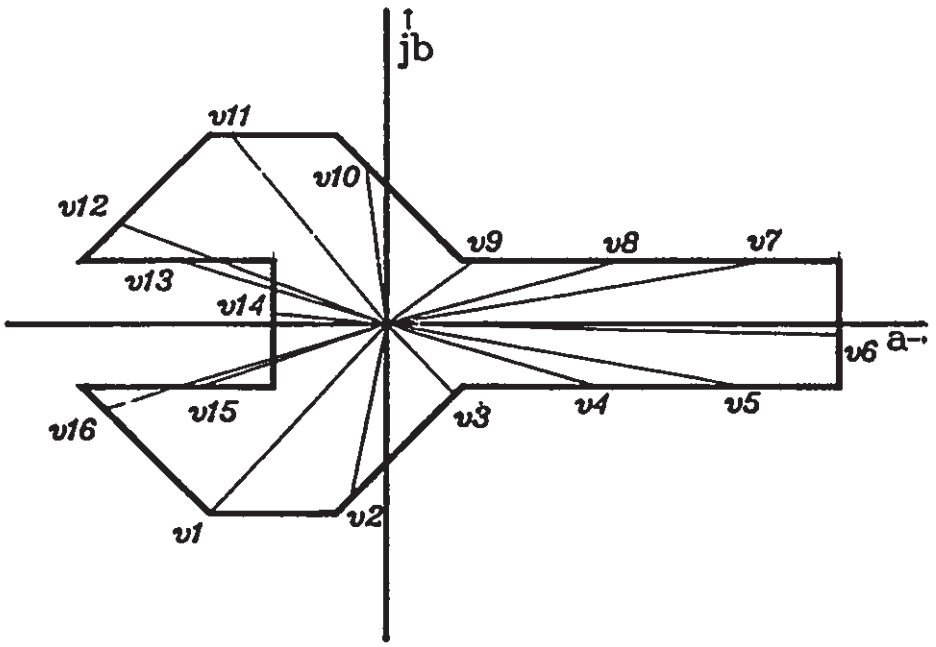


Figure 2.15. Two Shapes Producing the Same Radius Magnitude Function.

sequences for unique objects. This method also decreases the amount of computation necessary to compute the samples.



Complex sequence – $v1 = (a1,b1)$,
$v2 = (a2,b2)$,
$v3 = (a3,b3)$ ...

Figure 2.16. Example of the Complex Sequence Derived From a Shape Contour Mapped onto the Complex Plane.

2.2.2.1 Walsh Transform Shape Descriptors

Walsh functions are useful for the approximation of functions. When compared with other orthogonal functions, there are two advantages related to Walsh functions: 1) Walsh coefficients can be calculated by a parallel algorithm; and 2) computation of the coefficients can be achieved using only additions and subtractions. The Walsh functions can be derived as follows [2.14]:

For k = 0:

$$f_0(x) = 1 \qquad 0 <= x < 1$$

For k >= 1 :

$$f_k(x) = \begin{cases} f_p(2x) & 0 <= x < 1/2 \\ & p = [k/2] \\ (-1)^{k+p} f_p(2x-1) & 1/2 <= x < 1 \end{cases} \qquad [2.4]$$

Therefore the first eight functions are :

$f_0(x) =$ 1 1 1 1 1 1 1 1

$f_1(x) =$ 1 1 1 1 -1 -1 -1 -1

$f_2(x) =$ 1 1 -1 -1 -1 -1 1 1

$f_3(x) =$ 1 1 -1 -1 1 1 -1 -1

$f_4(x) =$ 1 -1 -1 1 1 -1 -1 1

$f_5(x) =$ 1 -1 -1 1 -1 1 1 -1

$f_6(x) =$ 1 -1 1 -1 -1 1 -1 1

$f_7(x) =$ 1 -1 1 -1 1 -1 1 -1

The Walsh transform can be calculated using the fast Walsh transform algorithm. For an eight element sample set the Walsh transform

is defined as :

$$c_i = \sum_{k=0}^{2n-1} (1/2n)\ g_k * f_i(x) \qquad k/2n <= x < (k+1)/2n \qquad [2.5]$$

Since $f_i(x)$ is 1 or -1 for all i, the calculation of c_i involves only additions and subtractions of the g_k values. Division by the 2n value can be carried out using a shift operation.

The fast Walsh transform recognizes that the transform can be broken down into a series of additions and subtractions of pairs of sample values. For example, calculation of the 4 coefficients for a 4 sample Walsh transform is as follows:

$$c_0 = 1/4(g_0 + g_1 + g_2 + g_3) = 1/4((g_0 + g_1) + (g_2 + g_3))$$
$$c_1 = 1/4(g_0 + g_1 - g_2 - g_3) = 1/4((g_0 + g_1) - (g_2 + g_3))$$
$$c_2 = 1/4(g_0 - g_1 - g_2 + g_3) = 1/4((g_0 - g_1) - (g_2 - g_3))$$
$$c_3 = 1/4(g_0 - g_1 + g_2 - g_3) = 1/4((g_0 - g_1) + (g_2 - g_3))$$

If the bracketed pairs are first calculated then the fast Walsh transform takes the form:

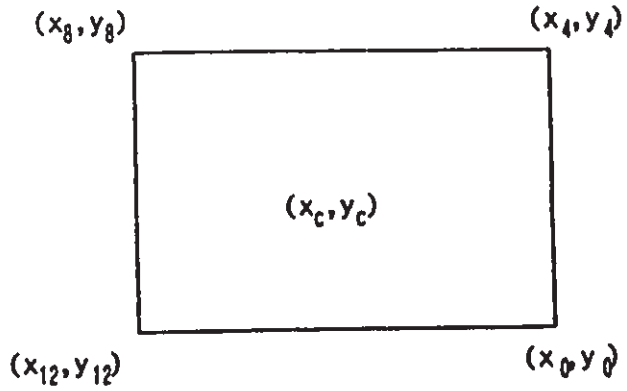
original data	Stage 1	Stage 2
g_0	$a_0 = g_0 + g_1$	$c_0 = a_0 + a_2$
g_1	$a_1 = g_0 - g_1$	$c_1 = a_0 - a_2$
g_2	$a_2 = g_2 + g_3$	$c_2 = a_1 - a_3$
g_3	$a_3 = g_2 - g_3$	$c_3 = a_1 + a_3$

All that remains is the shift to perform the division by 4. The number of computations necessary to perform a Walsh transform for various sample sizes is given in Table 2.2.

Number of Samples	Additions/Subtractions	Shifts
4	8	4
8	24	8
16	64	16
n	n*log(n)	n

Table 2.2. Computations Necessary for Walsh Transform

To determine the overall computational complexity involved in using Walsh descriptors, the overhead required in calculating the samples must be taken into account. For example, if 16 Walsh coefficients are to be calculated for the object of Figure 2.10, 16 samples must first be derived from the shape. This sampling requires that the perimeter and centroid of the object be found first in order to normalize the object with respect to size and translation. Furthermore, the Walsh transform coefficients vary with respect to cyclic shift of the starting point; i.e. different starting points on the object boundary result in a different ordering of samples which in turn results in different Walsh coefficients. In order to eliminate the effects of rotations, the next four moments would have to be calculated as was the case for moment calculations.



$$g_0 = \sqrt{(x_0 - x_c)^2 + (y_0 - y_c)^2}$$

$$x_1 = x_0 + m,\ y_1 = y_0 + 1/m \text{ etc.}$$

Figure 2.17. Shape Samples Derived from Simple Boundary Shape of Figure 2.10.

Having normalized the object with respect to size, rotation and translation, a simple set of samples can then be derived by navigating the boundary and finding the magnitude of radius vectors at equal distances. For the object of Figure 2.10, each sample requires 4 additions, 2 multiplications and 1 square root (Figure 2.17).

The overall number of computations required to derive 16 Walsh descriptors for the object of Figure 2.10 is given in Table 2.3. When compared with the results obtained for the moment calculations, significant computational savings are achieved. As the number of coefficients or moments is increased, the ratio of computational savings becomes greater. This is due to the fact that the first seven moments need only be computed once regardless of the subsequent number of Walsh

	Mult.	Add/Sub/Shift	Squ. Root	Divisions
Normalization	212	216	4	4
Sample Calculation	32	64	16	-
Walsh Trans.	-	80	-	-
	244	360	20	4

Table 2.3. Computations for Walsh Coefficients

coefficients derived. There are two drawbacks related to using Walsh descriptors. First, the necessity of adjusting the object's rotational orientation results in a significant amount of computation. Second, the object samples must be real values; this results in the need to use either the radius magnitude function with its related drawbacks, or the angular direction function which is more expensive to compute since it requires the computation of a number of angles.

2.2.2.2 Rapid Shape Descriptors

An interesting variation of the Walsh descriptors are the rapid shape descriptors described by Ma, Wu and Lu [2.5]. As is the case for Walsh descriptors the necessity for real input results in a similar boundary sampling method. Unlike the Walsh transform the rapid transform is invariant to cyclic shift in the input data. As a result the object need not be normalized for rotation. However, if the rotational

orientation of the object is desired, the computations necessary to find the orientation for the Walsh transform would still be necessary.

Computation of the rapid transform is similar to that of the Walsh transform with the addition of an absolute value operation at each stage. Thus a two stage transform for 4 coefficients takes the form:

original data	Stage 1	Stage 2
g_0	$a_0 = g_0 + g_1$	$c_0 = a_0 + a_2$
g_1	$a_1 = ABS(g_0 - g_1)$	$c_1 = ABS(a_0 - a_2)$
g_2	$a_2 = g_2 + g_3$	$c_2 = ABS(a_1 - a_3)$
g_3	$a_3 = ABS(g_2 - g_3)$	$c_3 = a_1 + a_3$

	Mult.	Add/Sub/Shift/ABS	Squ. Root	Divisions
Orientation*	212	216	4	4
Sample Calculation	32	64	16	-
Walsh Trans.	-	112	-	-
	244	392	20	4

* optional if orientation required

Table 2.4. Computations for Rapid Coefficients

The computational complexity for the rapid transform for Figure 2.17 is similar to that of the Walsh transform with the addition of the absolute value calculations and is summarized in Table 2.4.

2.2.2.3 Fourier Transform Shape Descriptors

The use of Fourier descriptors tends to alleviate the problems encountered by the use of moments or Walsh descriptors. The amount of computation necessary is significantly reduced when compared with moment calculations and the need to adjust the rotational orientation is eliminated. Furthermore the input to the Fourier transform may be a complex sequence. This eliminates the drawbacks of magnitude sampling methods. In order to demonstrate how this is accomplished the Fourier transform is briefly reviewed and the computational complexity of the fast Fourier transform is described.

The discrete Fourier transform is defined as follows:

$$G(k) = \sum_{n=0}^{N-1} g_n * e^{-jk(2\pi/N)} \qquad [2.6]$$

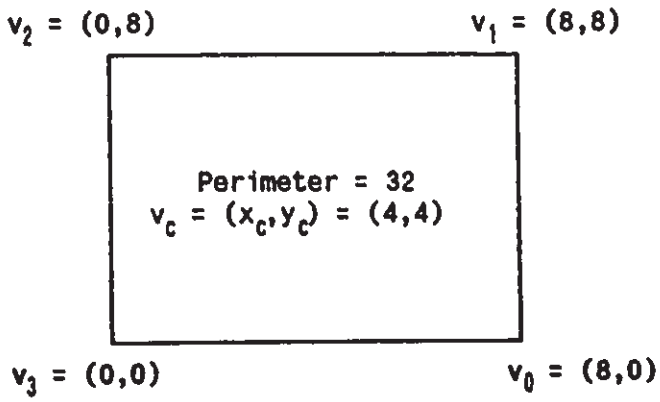
where:

g_n are the elements of the sample set and may be real or complex

The fast Fourier transform (FFT) recognizes the fact that many of the products which are summed to calculate the coefficients are repeated for the different coefficients.

The input for the FFT is the set of complex values derived from the object contour. In order to implement an effective set of shape

descriptors the object samples must first be normalized with respect to object size and translation. This is accomplished by finding the object perimeter and centroid. Normalization is carried out by mapping the centroid onto the origin and scaling the complex values according to area or perimeter. For comparative purposes the simple boundary example of Figure 2.10 is again used. This object and its complex sequence



In order to calculate the complex input values the following computations are made:

$$(x_0, y_0) = v_0 - v_c = (4,-4)$$

$$(x_1, y_1) = (x_0 + 1/m - x_c\ ,\ y_0 + m - y_c)$$

$$(x_2, y_2) = (x_1 + 1/m - x_c\ ,\ y_1 + m - y_c) \text{ etc.}$$

where m and 1/m are calculated as follows:

$m = S*(y_{i+1} - y_i)/(x_{i+1} - x_i)$ if $x_{i+1} <> x_i$, $y_{i+1} <> y_i$

$m = S\ ,\ 1/m = 0$ if $x_{i+1} = x_i$

$m = 0\ ,\ 1/m = S$ if $y_{i+1} = y_i$

S = scale factor

Figure 2.18. Computations Necessary to Compute Complex Input Samples for FFT.

calculation is given in Figure 2.18.

Thus the computations necessary to derive a set of 16 Fourier descriptors for the simple object of Figure 2.10 are summarized in Table 2.5.

	Real Mult.	Real Add/Sub/Shift	Squ. Root	Divis.
Normalization	20	24	4	4
Sample Calculation	-	32	-	-
FFT	256	264	-	-
	276	320	4	4

Table 2.5. Computations Necessary to Derive Fourier Coefficients

The rotational orientation of the object can be computed from the complex Fourier descriptors derived using the FFT. The equations used to calculate the rotational orientation are derived from the FFT process by examining the changes in the input to the FFT when an object is rotated. The method is clarified by the following example in which the same object at two different rotations is used (Figure 2.19).

The complex samples derived from the rotated version of the object differ from the samples derived from the reference object by a shift in the time axis t_0 and a rotation about the origin equivalent to multiplying the sample by e^{-jw} (w is the rotation about the origin). Thus the second sequence of

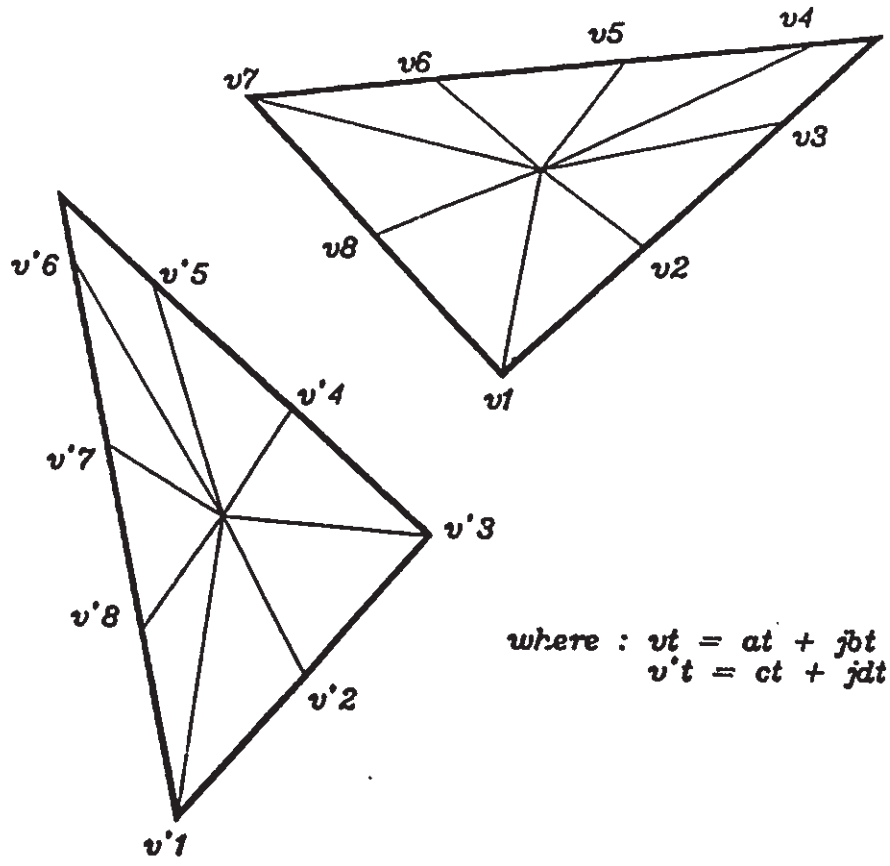


Figure 2.19. Input from the Same shape at Different Rotational Orientations.

samples can be written as a function of the first sequence. The equation relating the two sequences is:

$$x'(t) = e^{-jv} * x(t + t_0)$$

Thus the Fourier transform of the second sequence can be written in terms of the first sequence as:

$$X'(k) = \sum_{i=0}^{N-1} e^{-jv} * x_i(t + t_0) * e^{-jk(2\pi/N)}$$

$$= e^{-jw} * X(k) * e^{-jkto(2\pi/N)}$$

$$= e^{-j(w + kto(2\pi/N))} * X(k)$$

Thus for any two coefficients the difference in Fourier coefficient angles for an object and a rotated version of the same object is:

$$\angle X(k) - \angle X'(k) + 2\pi n = k * t_0 * (2\pi/N) + w$$

Since the angles of the Fourier coefficients of the reference object and the rotated object are known the relative rotation can be determined with 2 sets of Fourier phases :

$$\angle X(k_i) - \angle X'(k_i) + 2\pi n = k_i * t_0 * (2\pi/N) + w$$

$$\angle X(k_j) - \angle X'(k_j) + 2\pi m = k_j * t_0 * (2\pi/N) + w$$

Therefore:

$$w = A*k_j/(k_i-k_j) - B*k_j/(k_i-k_j) + 2\pi(k_j n - k_i m)/(k_i-k_j) \quad [2.7]$$

where :

n,m are unknown integer values
A is the angle difference of the i coefficient
B is the angle difference of the j coefficient

This equation can be solved for certain values of k_i and k_j. For example, if $(k_i - k_j) = 1$ or $k_j = 8$ and $k_i = 4$ then the third term of the equation becomes an integer multiple of 2π which can then be ignored. This will be referred to as the rotation calculation condition.

2.3 Selection and Performance of Fourier Descriptors

A number of shape descriptors were investigated in order to find a set of descriptors which would best meet a set of 6 criteria. All of the descriptors presented could be normalized in order to be independent of scale, rotation and cyclic shift of the starting point. Although the Walsh descriptors and the rapid descriptors are computationally efficient, both are computed using a sampling technique which is not unique for the shape given. As a result it is possible that these two methods yield ambiguous results. Furthermore, if rotational orientation is required a significant amount of computation is added. Both the Fourier descriptors and the moment calculations employ a unique sampling of the object contour. Of these two methods the Fourier descriptors are preferable since they require less computation time than the moments. Furthermore, in a study conducted by Reeves et al. [2.9] using airplane shapes it was shown that in terms of shape classification the Fourier descriptors outperformed moments derived from object boundaries. In this study the optimum correct classification for moments was 75% while the correct classification for Fourier descriptors was 92%. Also the first 7 moments cannot be used as descriptors since they must be employed to normalize the object with respect to rotation and scale. In the case of Fourier descriptors the orientation can be computed from the descriptors themselves. As a result of these observations a closer study of Fourier descriptors was carried out to determine their suitability.

Using the object shapes employed by MA,WU and LU a number of experiments were carried out to determine the operating characteristics

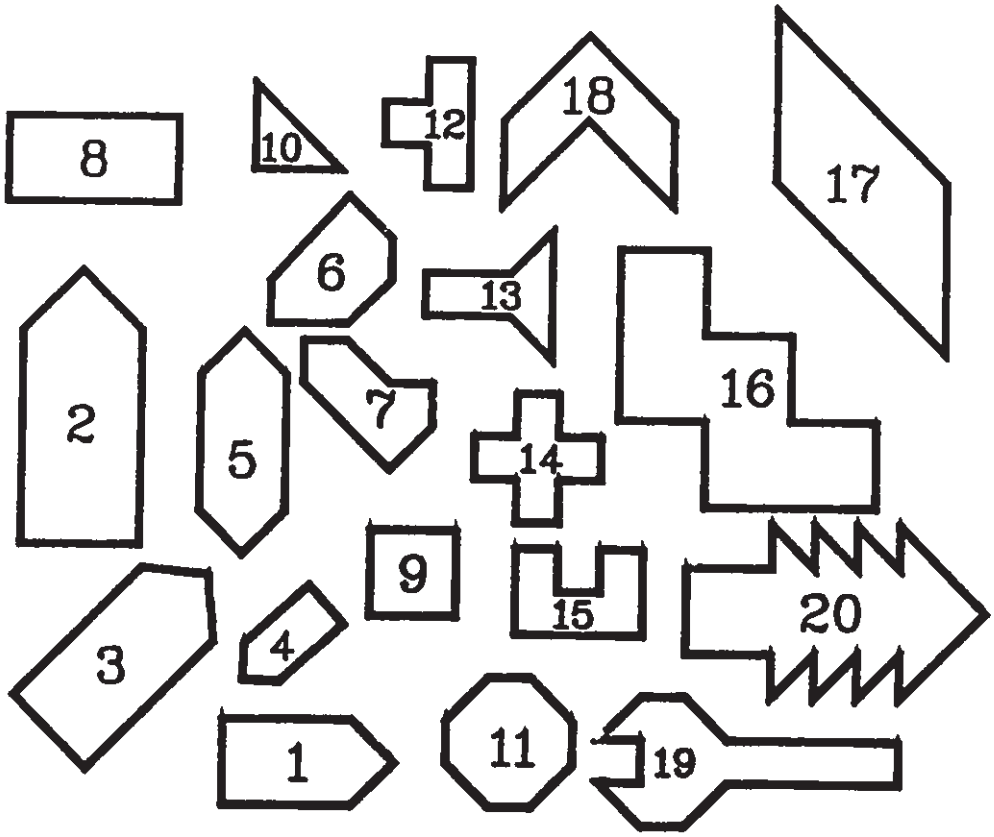


Figure 2.20. 20 Test Shapes Introduced by Ma, Wu, and Lu.

of Fourier descriptors. These shapes are depicted in Figure 2.20. In particular the following set of questions were addressed:

1) What is the relationship between the number of Fourier descriptors employed and the descriptor error for objects at a number of rotational orientations ?

2) How is object identification affected by the resolution of the image ?

3) How accurately can the object orientation (i.e. rotation) be calculated and how is this accuracy related to resolution and number of samples employed ?

In order to answer these questions the Fourier descriptors for the sample object shapes of Figure 2.20 were calculated using 4,8,16 and 32 samples. In each case the tests were carried out as follows. A set of template Fourier descriptors were learned for the shape at an initial orientation. Then the shape was rotated at 5 degree increments and the Fourier descriptors were calculated at each rotation. The coefficients were then compared to the template values and two statistics were computed. First, using the template as the norm, the average coefficient magnitude error was computed. Second, the average error in the rotation calculation was computed. Figure 2.21 depicts the average coefficient magnitude error for the 8 of the 20 shapes when 4,8,16 and 32 samples are used. These results are typical of the results obtained for the 20 shapes. As is evident from

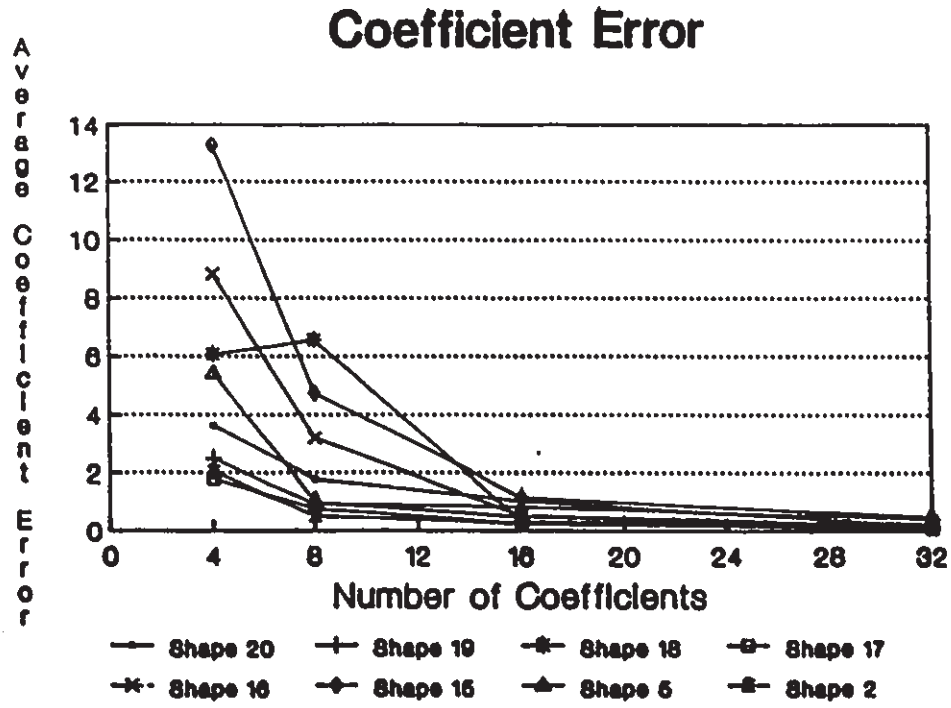


Figure 2.21. Average Coefficient Magnitude Error for 8 of the 20 Shapes.

the graph there is a significant decrease in error for each shape as the number of descriptors increases. Figure 2.22 depicts the average rotation calculation error for each of the 8 shapes. When less than 16 descriptors are used the average rotational error is as much as $\pi/2$ radians while for 32 descriptors it is reduced to about 0.1 radians. Thus if rotational orientation is important then 16 or 32 samples should be used for this set of shapes.

It should be noted that the choice of descriptors used in calculating the rotation must be made carefully. Generally, the phases of descriptors displaying the largest magnitudes should be used since these will be least affected by digitization of the shape contour. For example, Figure 2.23 depicts the Fourier descriptor magnitudes for shape 13. In

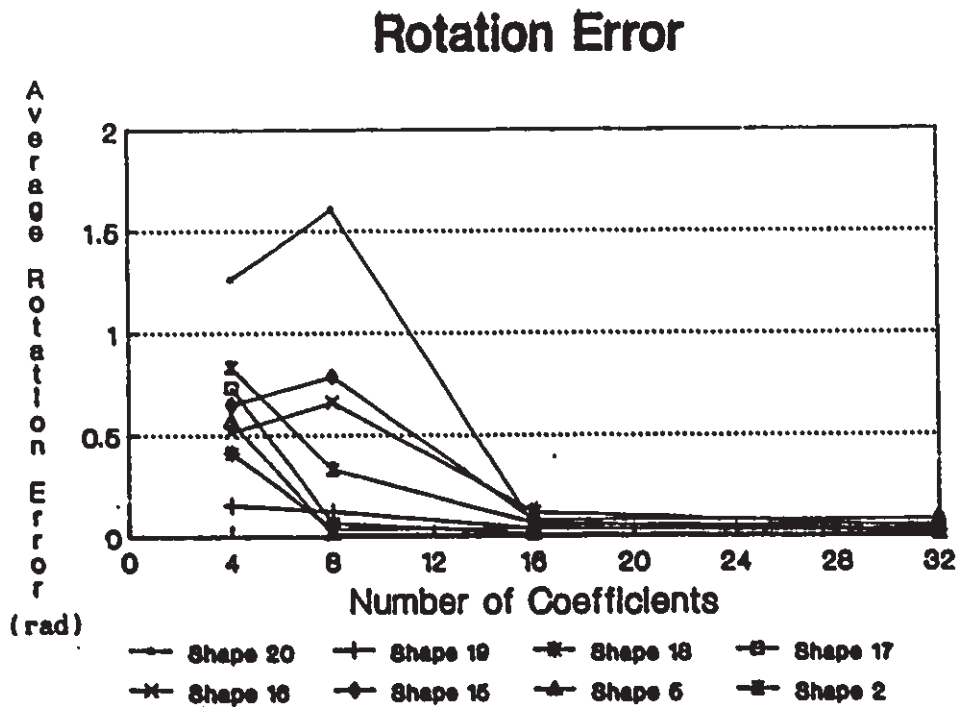


Figure 2.22. Average Rotation Error for 8 of the 20 Shapes.

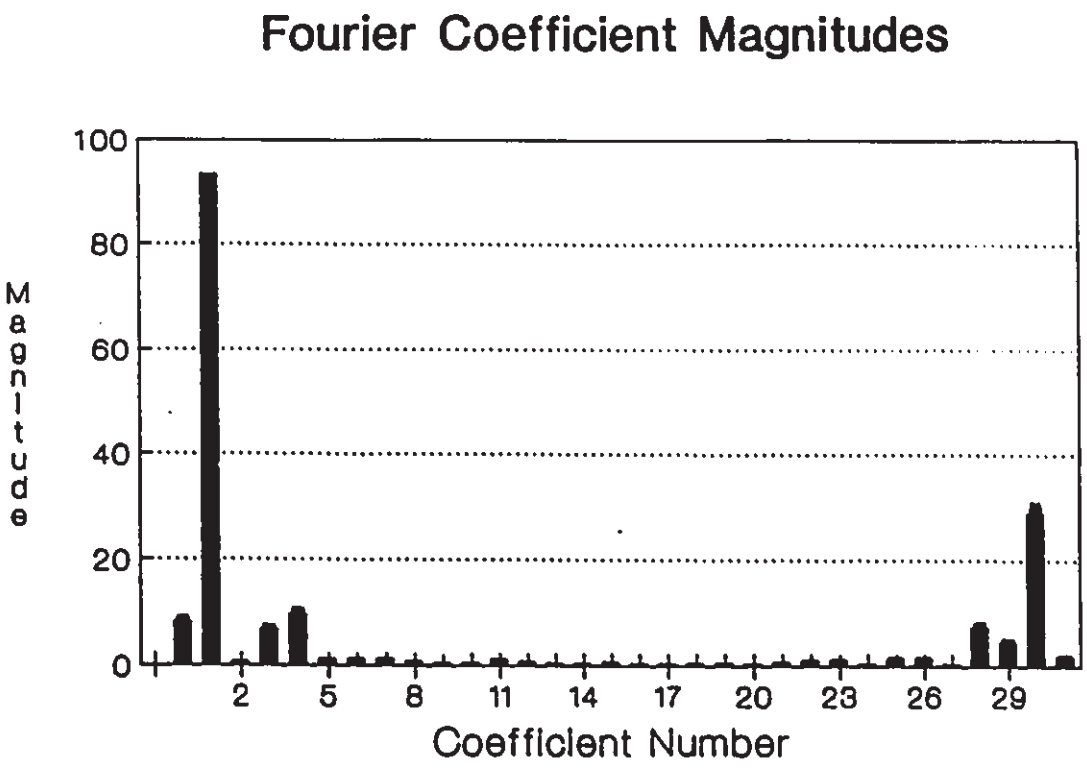


Figure 2.23. Fourier Descriptor Magnitudes for Shape 13.

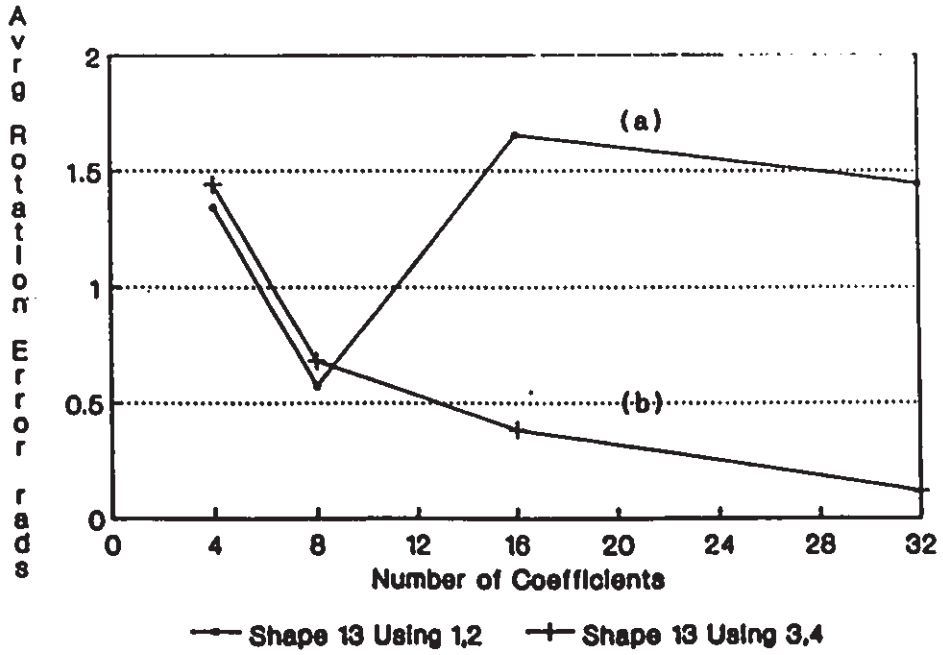


Figure 2.24. Resulting Rotation Error Calculating Rotation Using
a) coefficients 1,2
b) coefficients 3,4

order to fulfill the condition for rotation calculation the coefficients chosen must result in the third term of the equation 2.7 being a multiple of 2π. Two sets of coefficients which fulfill this requirement are 1,2 or 3,4. The resulting rotational error calculated using the first and second pair of coefficients is depicted in Figure 2.24 for 4,8,16 and 32 coefficients at 36 rotations of 5 degrees each. The second set of descriptors displays a superior accuracy. The cause of the poor rotation calculations yielded by the first set of descriptors can be traced to the small magnitude of the second descriptor.

A related problem arises when dealing with shapes displaying rotational symmetry such as shape 17 or shape 14 the Fourier descriptor magnitudes of which are displayed in Figure 2.25 (a) and (b). For the shape having 180° rotational symmetry every even descriptor displays a small magnitude and for the shape having 90° rotational symmetry only every fourth odd descriptor displays a significant magnitude. This presents a problem in that there does not exist, for each of these two cases, a pair of coefficients of significant magnitude which allow the rotational calculation condition to be met. This problem can be solved by taking account of the fact that for shapes displaying 180° rotational symmetry a valid range of rotation is 0 - 180° (0 - π) and for 90° symmetry a valid range is 0 - 90° (0 - $\pi/2$). Thus if the phases of descriptors 1 and 3 were used to calculate the rotation of shape 17 the following would result from equation 2.7:

$$w = A * (3/2) - B * (1/2) + 2\pi (3n - m)/2$$

$$= A * (3/2) - B * (1/2) + \pi (3n - m)$$

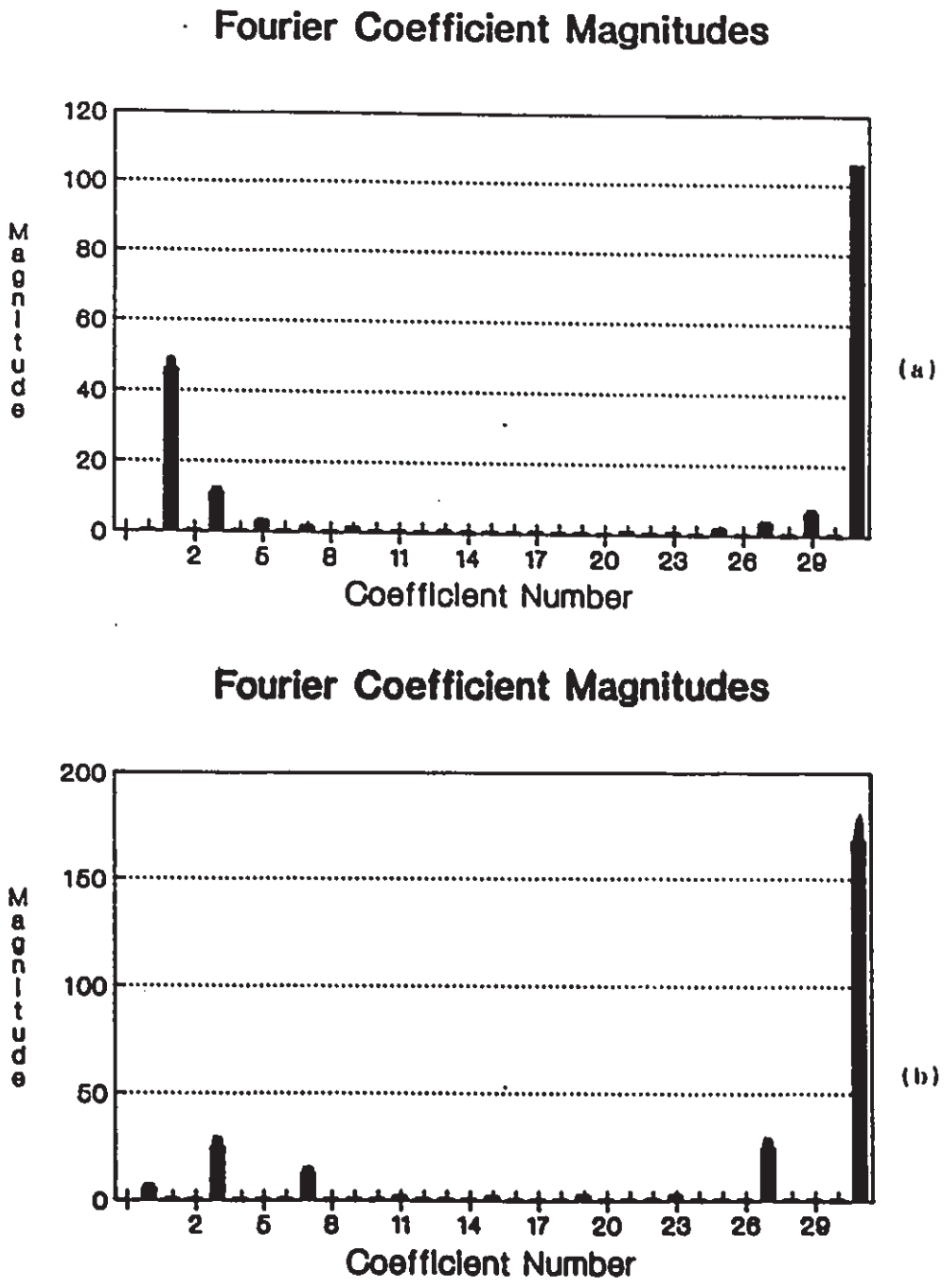


Figure 2.25. Fourier Descriptor Magnitudes for Shapes Displaying Rotational Symmetry.
a) magnitudes for shape 17
b) magnitudes for shape 14

Since the third term is an integer multiple of π and the valid range for rotation is 0 - π, the third term can be ignored. A similar result is yielded for object 14 when descriptors 3 and 7 are used.

In order to determine the effects of resolution a number of descriptor magnitude error and rotation error experiments were carried out using sets of shapes where each set consisted of similar shapes of differing size. A sample of the results obtained for shapes 9 and 13 are depicted in Figures 2.26 and 2.27. These figures show the improvement in both magnitude and rotation calculation as the resolution is increased. It should also be noted that for the set of shapes employed in these experiments the use of 32 coefficients yields relatively accurate magnitude and rotation calculation for a wide range of resolution values.

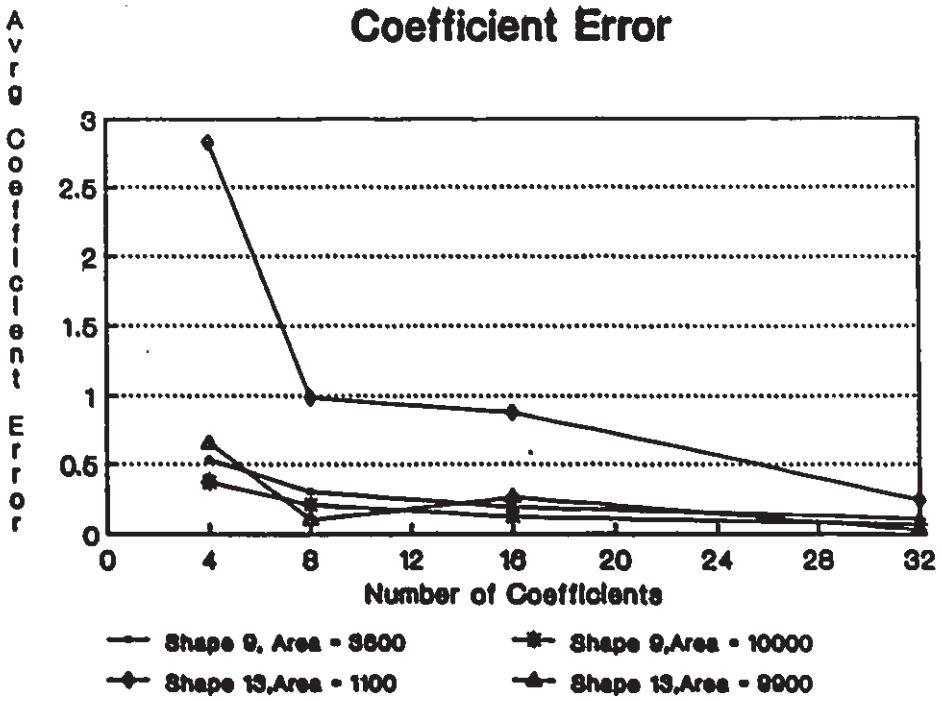


Figure 2.26. Effects of Resolution on Coefficient Error for Shapes 9 and 13.

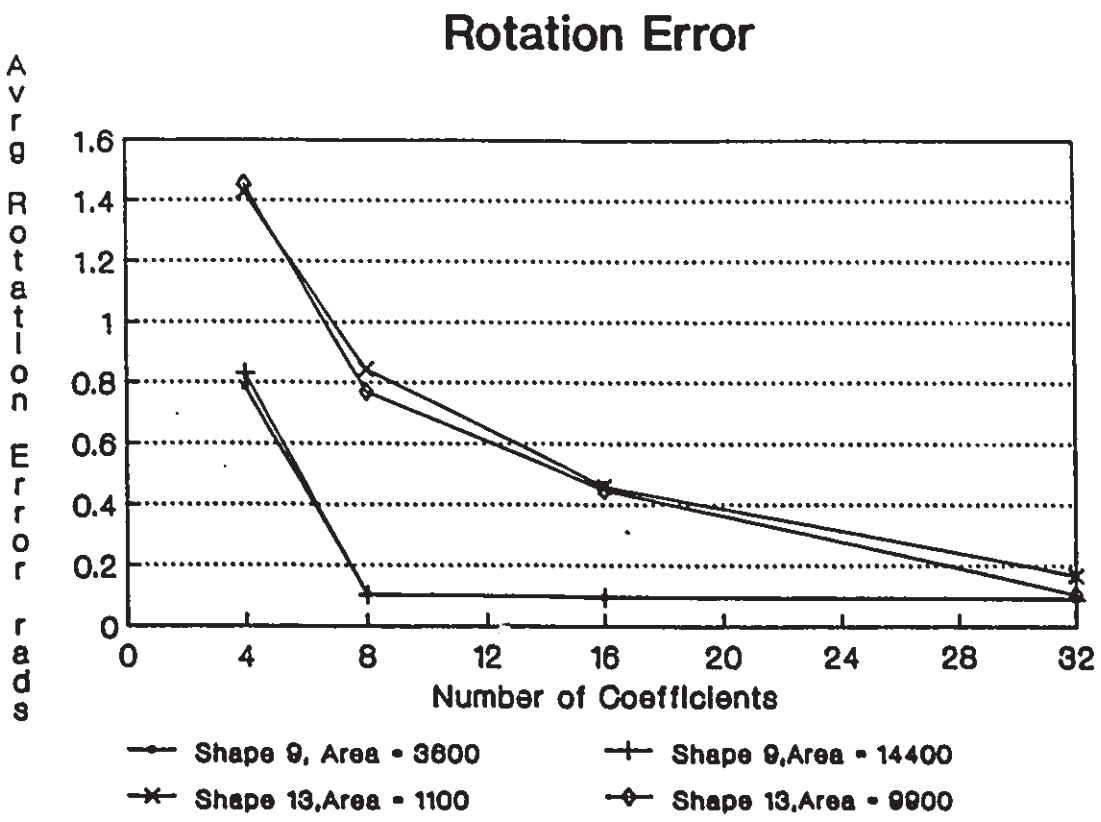


Figure 2.27. Effects of Resolution on Rotation Error for Shapes 9 and 13.

CHAPTER 3

TRACKING SYSTEM

The operation of the tracking system can be divided into four tasks: training, feature lock-on, feature window processing, and photogrammetric solution. The training phase includes learning the various projections of the planar shape under pitch and yaw perturbations and the selection of features of the shape to be used for tracking. Corners are useful features of planar shapes which may be used in order to track them. Davis et al. [3.1] have shown that corners have the property that they can be safely regarded as projections of scene features whose general appearance is invariant to rigid motion. In this respect corner projections are intuitively more appealing than target spot centroids. Spot centroids are calculated from projections and will not be the true centroid projections of the circular spots when viewed under yaw and pitch.

The remaining three tasks are carried out during tracking. The shape to be tracked is presented to the system at an arbitrarily selected orientation. The tracking system first attempts to identify the position of the shape by comparing it with the previously learned projections. Feature windows are placed at the estimated corner locations. To use

corners to estimate motion, a fast method of locating them within windows has been developed. This technique uses information which is similar to that derived for the location of target spots by Pinkney [1.13] and which can be gathered in much the same way. Data from the windows is gathered and analyzed to more accurately locate the corners and locations of the corner projections are then used to determine the position of the shape.

This chapter describes the four tasks carried out by the tracking system. The feature lock-on system based on the use of Fourier shape descriptors is first described. The data gathered during the training phase to carry out this task is also described.

Next, a brief description of the window placement strategy is included and the window placement calculations are described.

The data to be gathered and the calculations necessary to quickly locate the corners within the feature windows are detailed.

The following section consists of a description of the traditional photogrammetric approach for determining the three dimensional position of an object from a single perspective view.

This may be contrasted with the new technique developed first for a square four point target and then for an arbitrary four point planar pattern. Use of the new technique requires some further information about the pattern to be computed during the training phase. The computations necessary are also described in this section.

To implement the tracking system a number of practical aspects must be considered. Practical aspects were confronted during the development of hardware capable of collecting data for a target dot

tracking system. This hardware and its approach to dealing with the practical problems of lighting and camera selection as well as the processing speed requirements are discussed in the final section of this chapter.

3.1 Shape Projection Identification and Lock-On

During the training phase, the projections of the shape are learned by generating the Fourier descriptors for selected combinations of pitch and yaw. The present system performs a 32 point FFT using 32 complex samples taken at equal distances along the object contour. In generating the projections to be learned, two practical considerations should be taken into account; first, the shape projection at arbitrary pitch and yaw angles varies according to the shape's position in the frame and; second, the shape projection varies with distance from the camera. These two phenomena can be demonstrated by simple examples using a square shape. Figure 3.1 shows the projection of a square having a pitch of 45 degrees at two locations in the frame; one centred at the perspective centre and the other with the bottom edge of the shape through the perspective centre. Figure 3.2 shows the projection of the square at distances of 100mm, 200mm and 300mm from the camera. The 200mm and 300mm distant squares have been scaled to produce a projections of similar size. In each case the projection points were generated using the equations relating object and image points for perspective projections:

$$x' = \frac{f_c * X}{Z}$$

$$y' = \frac{f_c * Y}{Z} \qquad [3.1]$$

where:

(x',y') - image projection point

(X,Y,Z) - object point relative to the focal point

These two practical problems can be solved if the tracking system is trained using the following two rules.

1) The distance for object lock-on should be predefined. This is accomplished by moving the camera until the object is within an acceptable range.
2) The shape projection should have its centroid at the camera's perspective centre.

Thus when the shape projections are generated, they are generated at a fixed distance from the camera centred at the perspective centre. The training takes place in the following way:

1) A shape is presented to the system such that its plane is parallel to the image plane.
2) The user interactively places feature windows over the corners to be used for tracking.
3) The software then locates the corners and stores the relevant information for tracking (described in section 3.2.2).
4) Projections of the shape are generated for pitch and yaw angle combinations in 5 degree increments from -48 degrees to 47 degrees. A total of 400 projections are generated. For each projection, the 32 Fourier descriptors are generated and the first 8 are phase normalized and stored in a list.

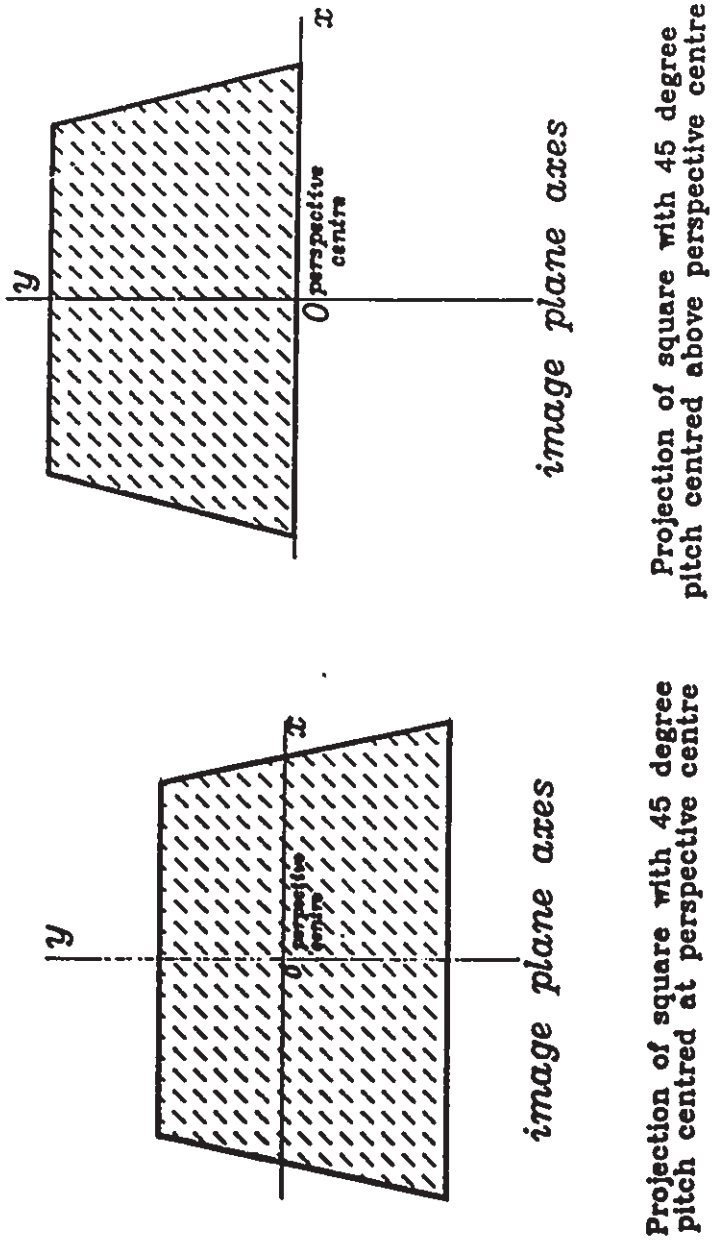


Figure 3.1. Projection of a Square Shape Pitched at 45 Degrees at Two Locations in the Frame.

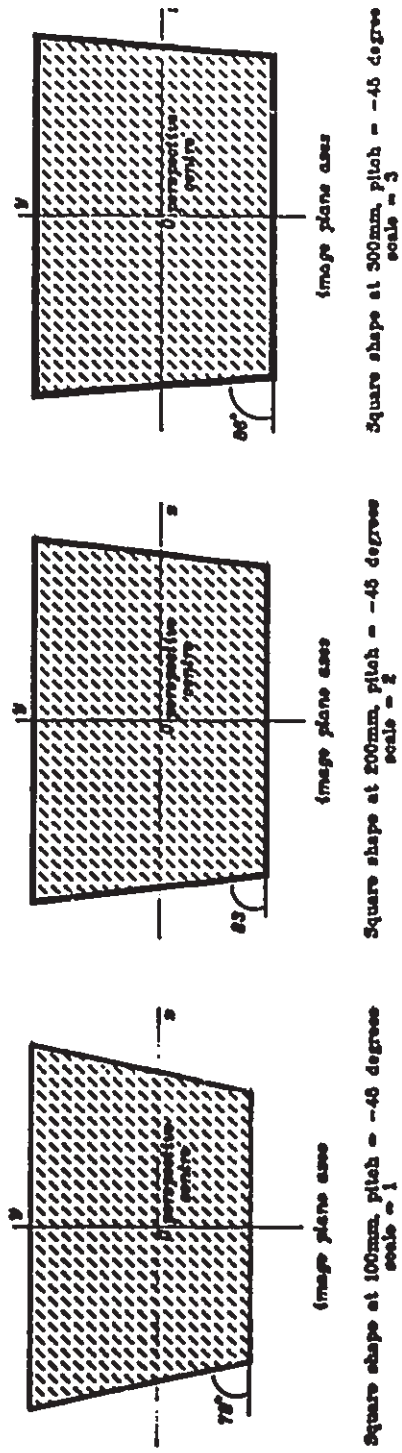


Figure 3.2. Projection of a Square Shape Pitched at 45 Degrees Located 100mm, 200mm and 300mm from the Camera.

When the descriptors of the shape's projections have been stored the training phase is complete. Locking on to an arbitrary projection is then accomplished by first generating the Fourier descriptors of the projection, phase normalization, and then finding the closest match amongst the set of 400. Phase normalization is carried out using the technique described by Wallace [1.17]. Each descriptor is multiplied by:

$$e^{j[(i-1)u + (1-i)v]/(k-1)}$$

where:

$u = \underline{/A(1)}$	phase of descriptor 1
$v = \underline{/A(k)}$	phase of descriptor of highest magnitude (other than 1)

This has the effect of setting the phases of descriptors 1 and k to zero and normalizing the remaining phases.

The closest match is defined as that having the smallest distance from the unknown projection. The distance measure is defined by equation 3.2.

$$d = \sum_{i=0}^{7} \sqrt{(x_{pi} - x_{ti})^2 + (y_{pi} - y_{ti})^2} \qquad [3.2]$$

where:

(x_{pi}, y_{pi}) are the complex coefficients of the first eight descriptors of the unknown projection derived using the FFT algorithm.

(x_{ti}, y_{ti}) are the first eight complex coefficients of each projection stored in the table.

Once the closest match is found the roll is estimated using the technique outlined in Chapter 2. Given the estimated pitch (Ω), yaw (θ) and roll (Φ)

the initial window positions are calculated by finding the projections of the corners to be used for tracking. This is accomplished by transforming the corner positions learned when the planar shape was initially presented parallel with the image plane. The rotation matrix is defined as:

$$M = \begin{bmatrix} \cos(\Phi)\cos(\Omega) & \sin(\theta)\sin(\Omega)\cos(\Phi) + \cos(\theta)\sin(\Phi) & \sin(\theta)\sin(\Phi) - \cos(\theta)\sin(\Omega)\cos(\Phi) \\ -\cos(\Omega)\sin(\Phi) & \cos(\theta)\cos(\Phi) - \sin(\theta)\sin(\Omega)\sin(\Phi) & \sin(\theta)\cos(\Phi) + \cos(\theta)\sin(\Omega)\sin(\Phi) \\ \sin(\Omega) & -\sin(\theta)\cos(\Omega) & \cos(\theta)\cos(\Omega) \end{bmatrix}$$

Thus the corner projection (x',y') of a corner initially located at position (x,y) during the training phase is:

$$x' = f_e \frac{a_{11} x + a_{12} y + a_{13} z}{Z}$$

$$y' = f_e \frac{a_{21} x + a_{22} y + a_{23} z}{Z} \qquad [3.3]$$

where:

$z = 0$ since the trained planar shape is parallel to the image plane

$Z = f_e a_{31} x + a_{32} y + a_{33} z$

f_e - is the camera focal length.

a_{ij} - are the elements of the rotation matrix.

The windows are then placed centred at the estimated corner projections. For example, for the projection of a square shape in Figure 3.3 (a) the actual orientation of the square is pitch 30^0, yaw 30^0, roll 30^0. The estimated position based on the closest match in the data base

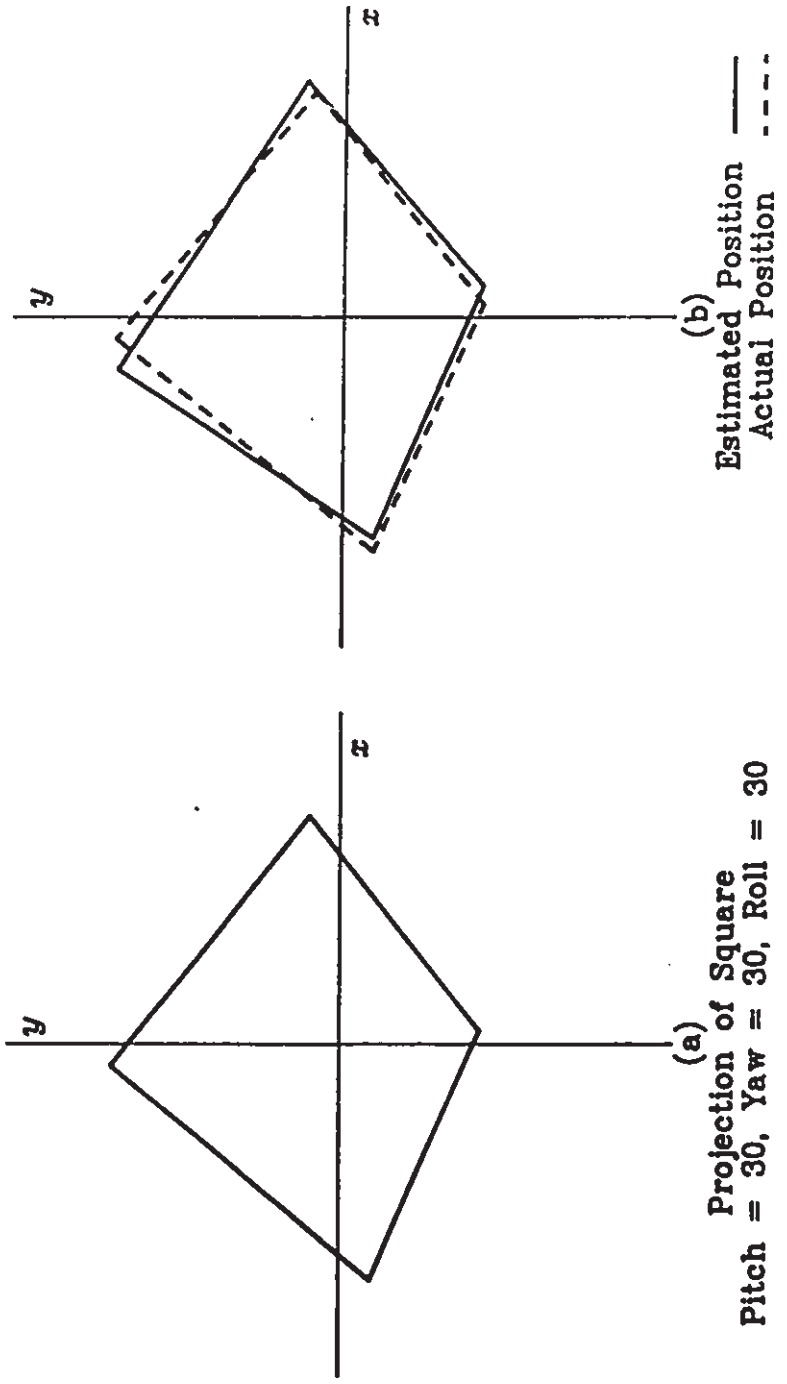


Figure 3.3. Comparison of Actual Projection and Projection Estimated using the Closest Matching Fourier Descriptor Set.
a) Actual square shape projection.
b) Estimated square shape projection position.

of projections is pitch 32^0, yaw 27^0, estimated roll 23^0. Thus the windows are placed at the estimated corner projection positions computed using 3.3 as shown in Figure 3.3(b).

3.2 Corner Location

For corners intersecting a square or rectangular window the four cases of Figure 3.4 would result. To locate the corner within each window the following data is collected for the portion of the planar shape contained within it:

1) area
2) x-moment
3) y-moment
4) area of intersection with each window edge
5) y-moments of intersections with x window edges
6) x-moments of intersections with y window edges

Using this information the corner within the window is located using the geometric properties of each intersection case. To determine the intersection case the area of intersection with each window edge is examined. If three intersecting areas are zero then case 1 is present, if two areas are zero then case 2 is present etc.

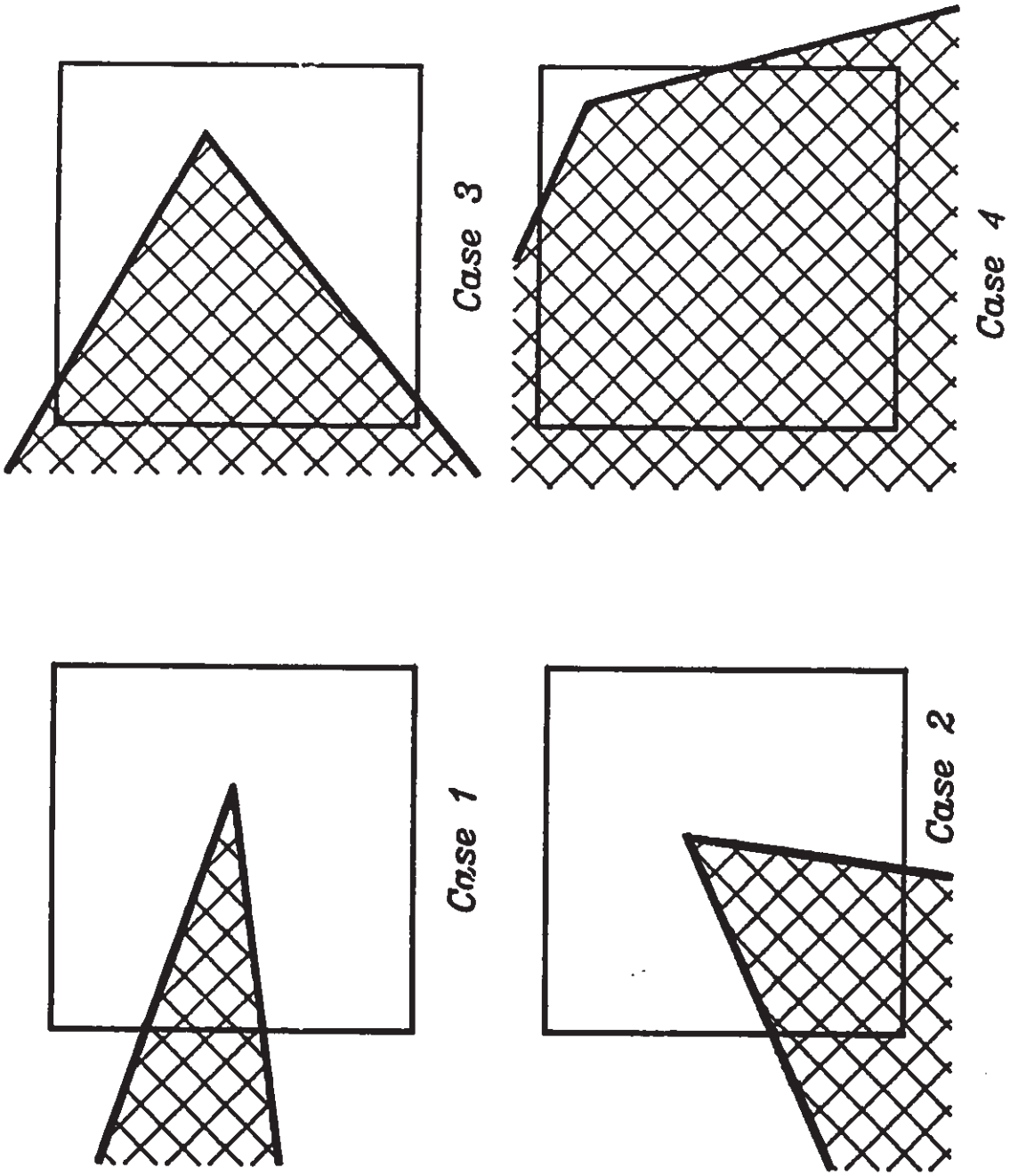
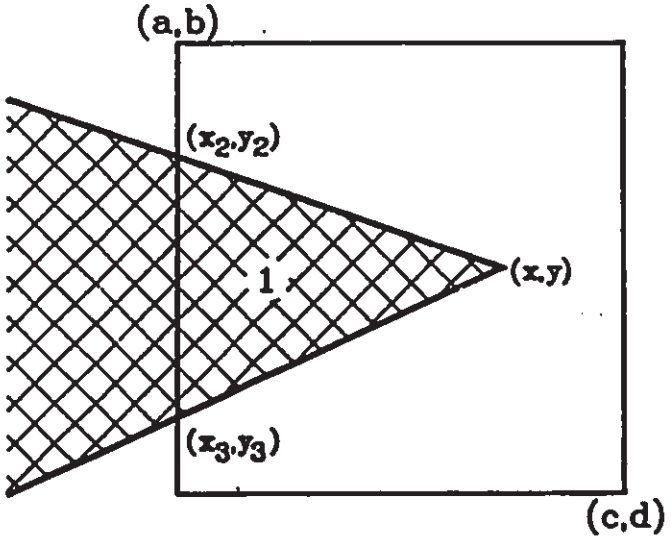


Figure 3.4. Four Cases of Corner Window Intersection.

For case 1 intersections the relevant information is depicted in Figure 3.5. Location of the shape corner (x,y) is carried out by the following computations.



where : (a,b),(c,d) – coordinates of window corners

$(x,y),(x_2,y_2),(x_3,y_3)$ – unknown

Data gathered for shape area in window

X_m – *x moment*
Y_m – *y moment*
A – *area*
Y_a – *y moment of shape intersection with left edge*
A_a – *area of shape intersection with left edge*

Figure 3.5. Data Collection for Case 1 Intersections.

First, the centroid of the triangular shape (1) is determined.

$$X_{c1} = X_s / A$$
$$Y_{c1} = Y_s / A \qquad [3.4]$$

Since the x centroid can also be calculated from:

$$X_{c1} = (x + x_2 + x_3) / 3$$

Therefore:

$$x = 3*X_{c1} - x_2 - x_3 \qquad [3.5]$$

where:

$x_2 = x_3 = a$ = the left window edge .

Similarly the y centroid can be calculated from:

$$Y_{c1} = (y + y_2 + y_3) / 3$$

Therefore:

$$y = 3*Y_{c1} - y_2 - y_3$$
$$= 3*Y_{c1} - (y_2 + y_3)$$

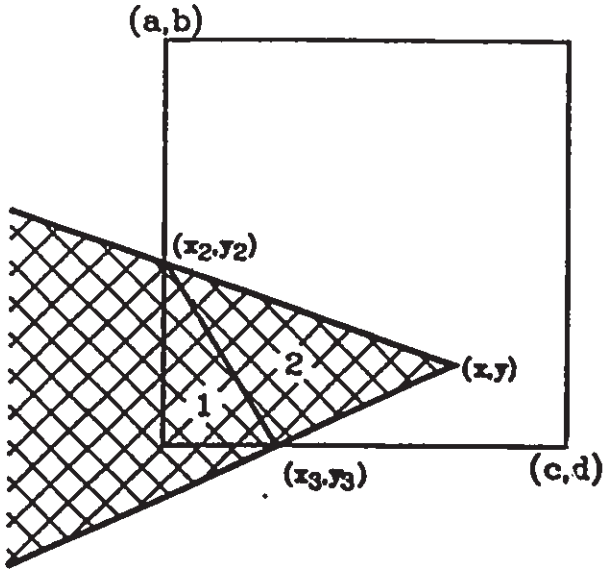
but, $(y_2 + y_3)$ is related to the y-moment of the intersection of the shape with the left window edge in the following way:

$$(y_2 + y_3) /2 = Y_s / A_s$$

Thus:

$$y = 3*Y_{c1} - 2*(Y_s / A_s) \qquad [3.6]$$

For case 2 intersections the relevant information is illustrated in Figure 3.6. Location of the shape corner (x,y) is carried out as follows. The moments and area of the triangle (2) of which (x,y) is a vertex are found by subtracting the moments and area of the triangle (1) from the moments and area gathered for the shape area within the window. The area of (1) is found using the lengths of the two intersecting sides.



where : (a,b),(c,d) – coordinates of window corners
$(x,y),(x_2,y_2),(x_3,y_3)$ – unknown

Data gathered for shape area in window

X_m – *x moment*
Y_m – *y moment*
A – *area*
A_a – *area of shape intersection with left edge*
A_d – *area of shape intersection with bottom edge*

Figure 3.6. Data Collection for Case 2 Intersections.

The moments of the triangle (1) are found by multiplying its area with its centroid.

$$A_1 = (A_a * A_d) / 2$$

$$X_{c1} = (2*a + (A_d + a)) / 3$$

$$Y_{c1} = (2*d + (d - A_a) / 3$$

$$X_{s1} = X_{c1} * A_1$$

$$Y_{s1} = Y_{c1} * A_1$$

Therefore:

$$A_2 = A - A_1$$

$$X_{c2} = (X_s - X_{s1}) / A_2$$

$$Y_{c2} = (Y_s - Y_{s1}) / A_2$$

Once the centroid of (2) is found the corner (x,y) is easily computed.

$$x = 3 * X_{c2} - a - (a + A_d) \qquad [3.7]$$

$$y = 3 * Y_{c2} - d - (d - A_a) \qquad [3.8]$$

For case 3 intersections the relevant information is depicted in Figure 3.7. Location of the shape corner (x,y) is carried out by first finding the centroid and area of the triangle (3) in Figure 3.7 and then using a method similar to that of case 2. The moments and area of the rectangle (1) are first computed.

$$A_1 = A_a * A_b$$

$$X_{s1} = A_1 * (A_b / 2 + a)$$

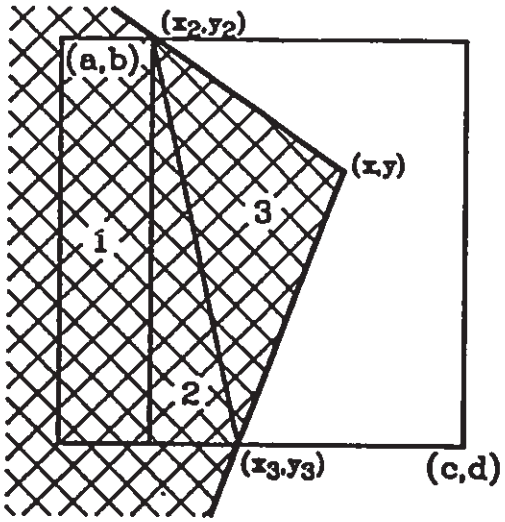
$$Y_{s1} = A_1 * (A_a / 2 + b)$$

The moments and area of the intermediate triangle (2) are also computed.

$$A_2 = 1/2 * (A_d - A_b) * A_b$$

$$X_{s2} = A_2 * 1/3 * (A_d + a + 2 * (A_b + a))$$

$$Y_{s2} = A_2 * 1/3 * (2 * d + b)$$



where : (a,b),(c,d) – coordinates of window corners
(x,y),(x_2,y_2),(x_3,y_3) – unknown

Data gathered for shape area in window

X_m – *x moment*
Y_m – *y moment*
A – *area*
A_a – *area of shape intersection with left edge*
A_b – *area of shape intersection with top edge*
A_d – *area of shape intersection with bottom edge*

Figure 3.7. Data Collection for Case 3 Intersections.

Thus the moments and area of the triangle containing the corner (3) can be computed.

$$A_3 = A - A_1 - A_2$$

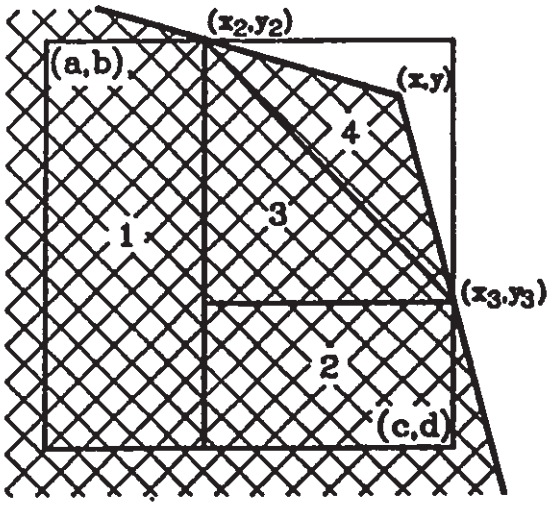
$$X_{a3} = X_a - X_{a1} - X_{a2}$$

$$Y_{a3} = Y_a - Y_{a1} - Y_{a2}$$

The estimated coordinates of the corner (x,y) can then be found by the equations.

$$x = 3 * (X_{sj}/A_j) - (a + A_b) - (a + A_d) \quad [3.9]$$

$$y = 3 * (Y_{sj}/A_j) - b - d \quad [3.10]$$



where : (a,b),(c,d) – coordinates of window corners
(x,y),(x_2,y_2),(x_3,y_3) – unknown

Data gathered for shape area in window

X_m – *x moment*
Y_m – *y moment*
A – *area*
A_a – *area of shape intersection with left edge*
A_b – *area of shape intersection with top edge*
A_c – *area of shape intersection with right edge*
A_d – *area of shape intersection with bottom edge*

Figure 3.8. Data Collection for Case 4 Intersections.

For case 4 intersections the relevant data is listed in Figure 3.8. In this case the estimated location of the shape corner is found by separating the area within the window into 4 parts, 2 rectangles and 2 triangles. First, the areas and moments of the 2 rectangles (1 and 2) and one triangle (3) are found.

$$A_1 = A_a * A_b$$

$$X_{a1} = A_1 * (A_b / 2 + a)$$

$$Y_{a1} = A_1 * (A_a / 2 + b)$$

$$A_2 = A_c * (A_d - A_b)$$

$$X_{a2} = A_2 * ((A_d - A_b / 2) + a + A_b)$$

$$Y_{a2} = A_2 * (A_a / 2 + b)$$

$$A_3 = 1/2 * (A_a - A_c) * (A_d - A_b)$$

$$X_{a3} = A_3 * 1/3 \ (2*(A_b + a) + A_d + a)$$

$$Y_{a3} = A_3 * (2*(d - A_c) + b)$$

Thus the moments and area of the second triangle (4) can be calculated by subtracting the moments and area of the rest of the window from the totals gathered.

$$A_4 = A - A_1 - A_2 - A_3$$

$$X_{a4} = X_a - X_{a1} - X_{a2} - X_{a3}$$

$$Y_{a4} = Y_a - Y_{a1} - Y_{a2} - Y_{a3}$$

In this case the estimated coordinates of the shape corner (x,y) can then be found using the equations.

$$x = 3* (X_{a4}/A_4) - (a + A_b) - (a + A_d) \qquad [3.11]$$

$$y = 3* (Y_{gl}/A_l) - b - (d - A_c) \qquad [3.12]$$

The algorithm to estimate the shape corner location can be divided into 2 stages. First, the data is gathered for each of the windows and the intersection case (1 to 4) for each window is determined. Next, the computations for the appropriate case are carried out to determine the corner location in each window.

The data necessary to find the corners in any of the four cases can be summarized as follows:

X_g - x moment of the shape area in the window

Y_g - y moment of the shape area in the window

A - area of the shape within the window

A_a - shape area intersecting the left edge

A_b - shape area intersecting the top edge

A_c - shape area intersecting the right edge

A_d - shape area intersecting the bottom edge

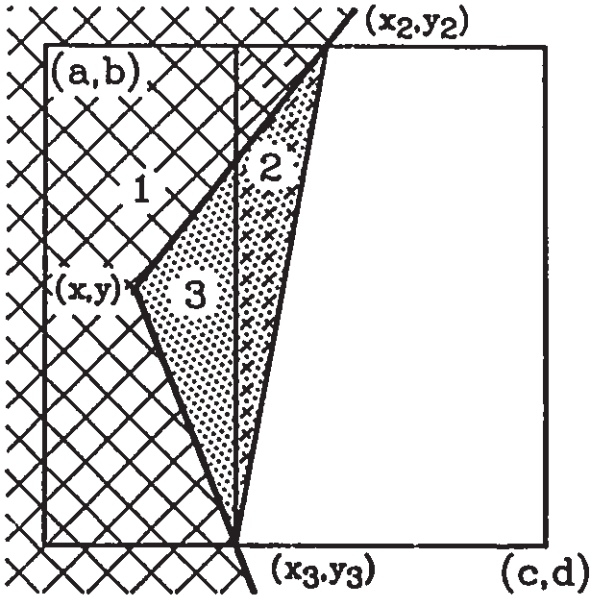
Y_a - y moment of shape area intersecting the left edge

X_b - x moment of shape intersecting the top edge

Y_c - y moment of shape intersecting the right edge

X_d - x moment of shape intersecting the bottom edge

The intersection case is determined by the intersection area values A_a - A_d. The number of non-zero areas determines the case and subsequently the processing employed to determine the corner position. It should be noted that for interior corners as in Figure 3.9 similar computation takes place. For the example in Figure 3.9 where three intersections are present the position calculations are identical until the last step at which point



where : (a,b),(c,d) – coordinates of window corners
(x,y),(x_2,y_2),(x_3,y_3) – unknown

Data gathered for shape area in window

X_m – *x moment*
Y_m – *y moment*
A – *area*
A_a – *area of shape intersection with left edge*
A_b – *area of shape intersection with top edge*
A_d – *area of shape intersection with bottom edge*

Figure 3.9. Data Collection for an Interior Corner for Case 3 Intersections.

the area and moments of the triangle (3) are found by reversing the order of subtraction.

$$A_3 = A_1 + A_2 - A$$

$$X_{a3} = X_{a1} + X_{a2} - X_a$$

$$Y_{a3} = Y_{a1} + Y_{a2} - Y_a$$

This situation can be detected by carrying out the computation in the same way as for exterior corners until the area of the triangle containing the shape corner is found. If the area is negative then an interior corner is indicated. Similar conditions exist for interior corners for cases 1, 2, or 4.

A number of experiments were carried out to characterize the performance of the corner detection algorithm. The effects of three parameters were investigated: the angle of the corner, the angle at which the corner intersects the window, and the size of the window. The experiments were carried out as follows. A number of corners of angles from 10 degrees to 170 degrees as depicted in Figure 3.10 were generated. Square windows of widths 10 to 80 pixels were placed over the corners with the corner point in the centre of the window in each case. For each corner size combined with each window size the corners were rotated into 24 different positions in 15 degree increments as depicted in Figure 3.11 for a 10 degree corner. In each case (about 3000 tests) the corner position calculated by the corner detection algorithm was compared to the actual corner projection point and the absolute error was calculated from:

$$\text{absolute\_error} = \sqrt{(x' - x)^2 + (y' - y)^2} \qquad [3.13]$$

where:

(x',y') - estimated corner position
(x,y) - actual corner projection location

The results were then accumulated according to corner angle, window size, and intersection angle in order to observe the effects of

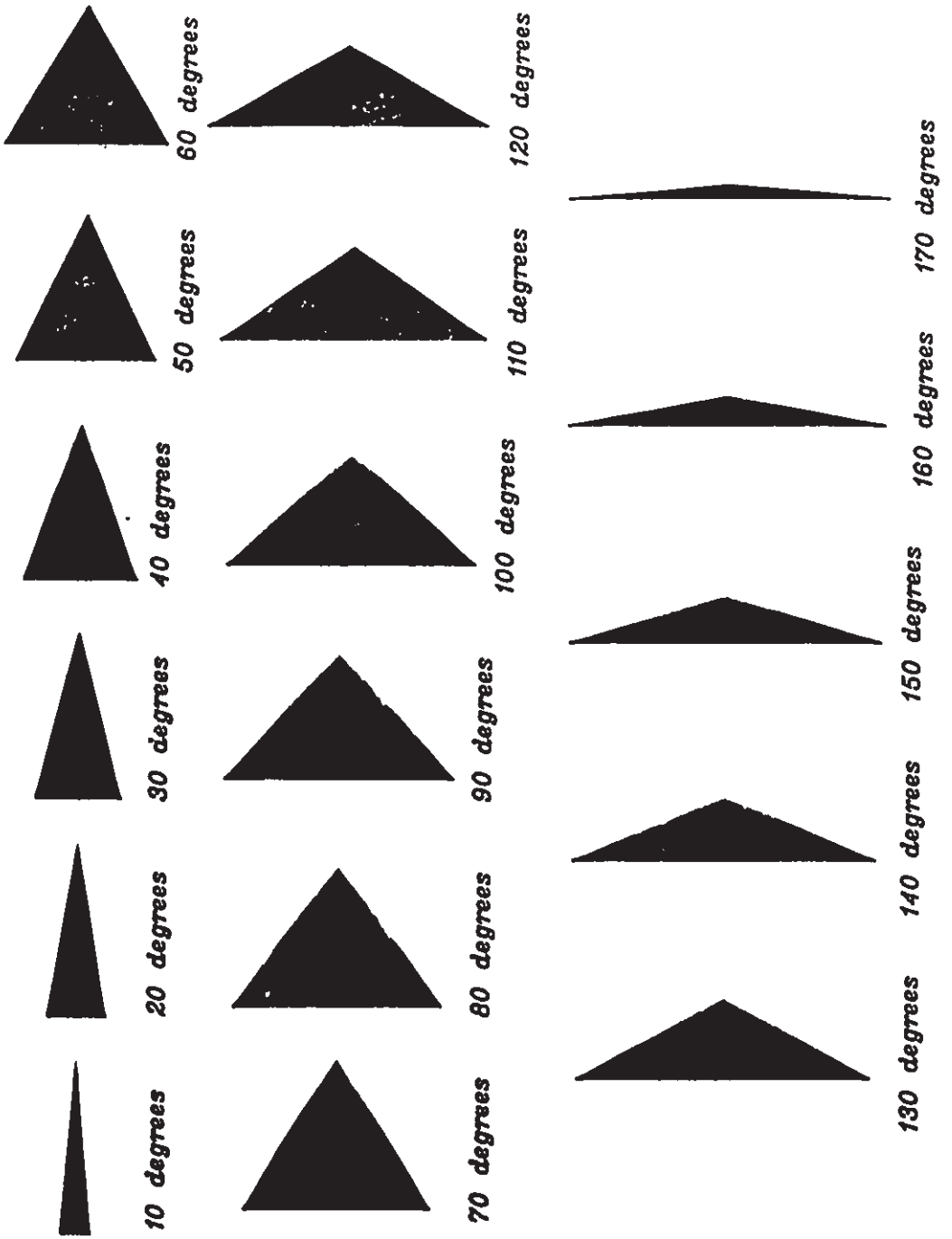


Figure 3.10. Shape Corner Angles of 10 to 170 Degrees.

these parameters. The average error vs the corner angle results are illustrated in Figure 3.12. As the graph indicates the average error for corner angles of between 30 and 110 degrees is less than 0.5 pixels whereas smaller or larger angles result in larger errors. The standard deviations of the errors are of similar magnitude.

The results obtained for average error vs the window size are plotted in Figure 3.13. Window size does not effect the average error significantly. The error in corner calculation is due to the errors

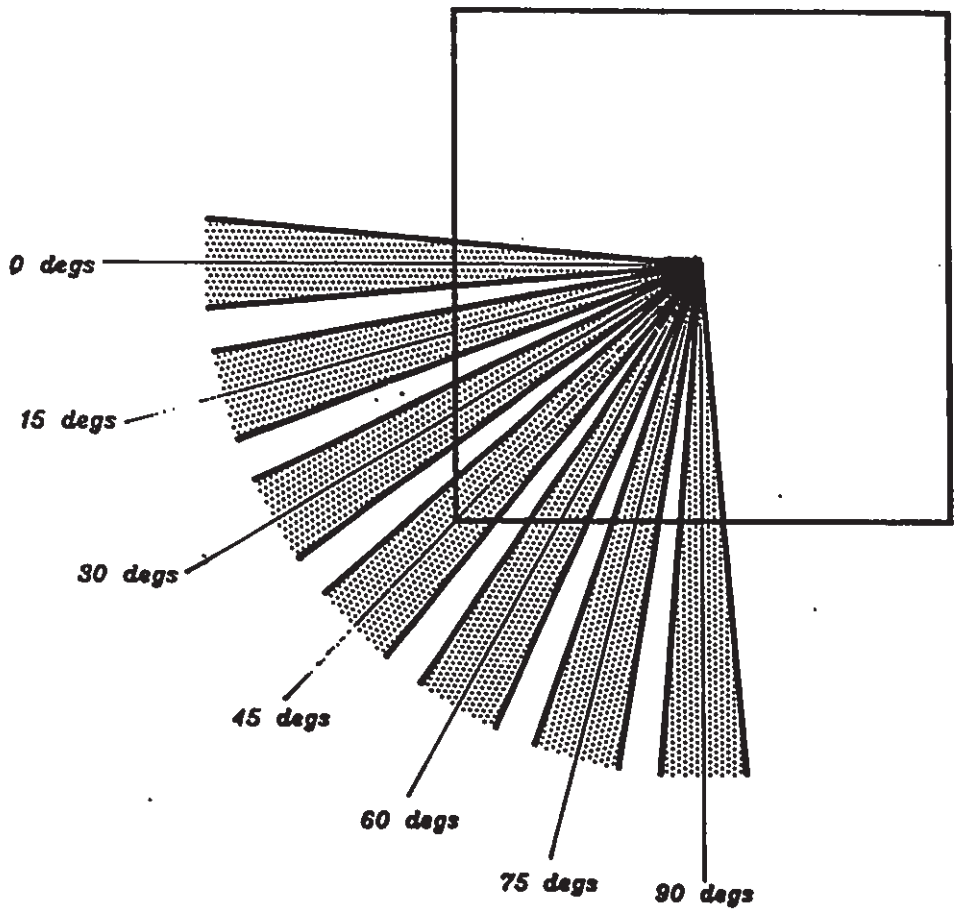


Figure 3.11. Corner Window Intersection Angles for a 10 Degree Corner from 0 to 90 Degrees.

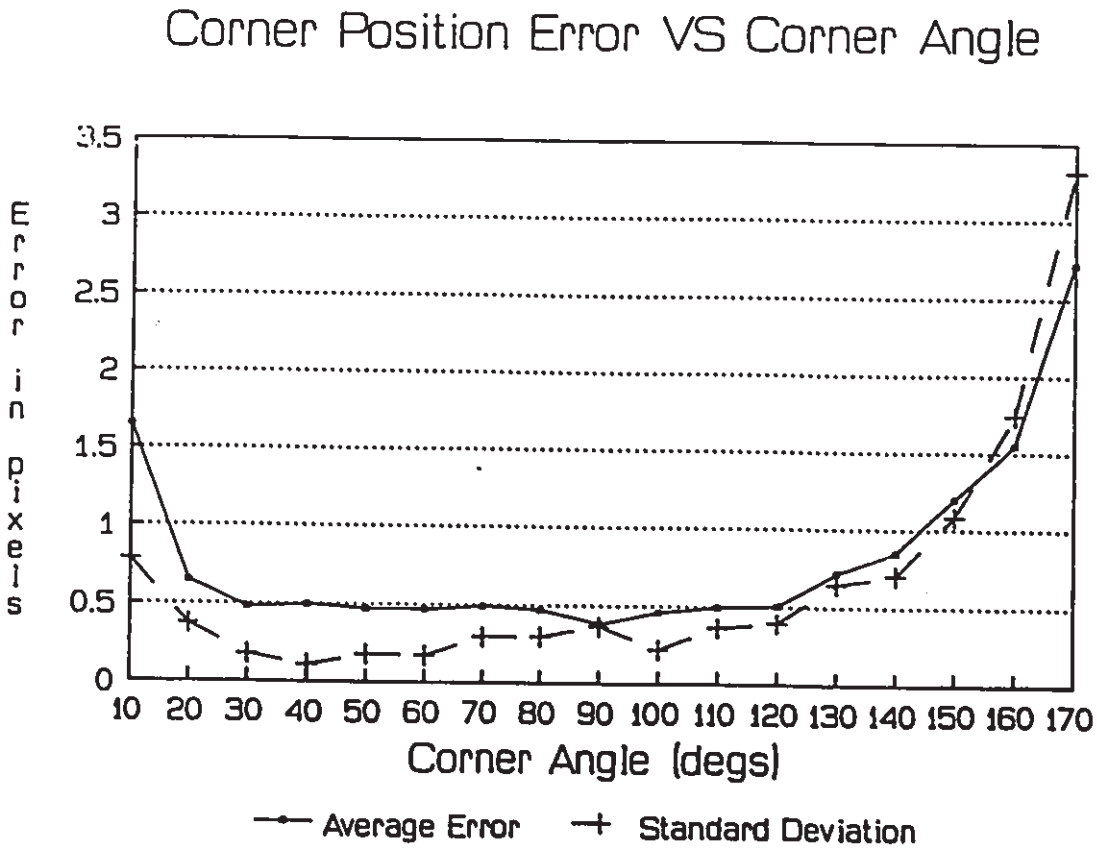


Figure 3.12. Average Corner Location Error vs Corner Angle

introduced through the digitization of the corner edges. Larger windows result in more object edge within the window and consequently proportionally more error is introduced into the calculations to determine the corner location. Thus the average error remains stable for different window sizes.

Figure 3.14 illustrates the effects of the corner intersection angle with the window. Whereas intersection angles of 0, 90, 180, and 270 degrees result in minimum average errors of about 0.3 pixels intersection angles of 45, 135, 225, and 315 degrees result in the maximum average errors of about 0.8 pixels. This phenomenon may be explained intuitively if the resulting digitized images in Figure 3.15 for a corner angle of 60

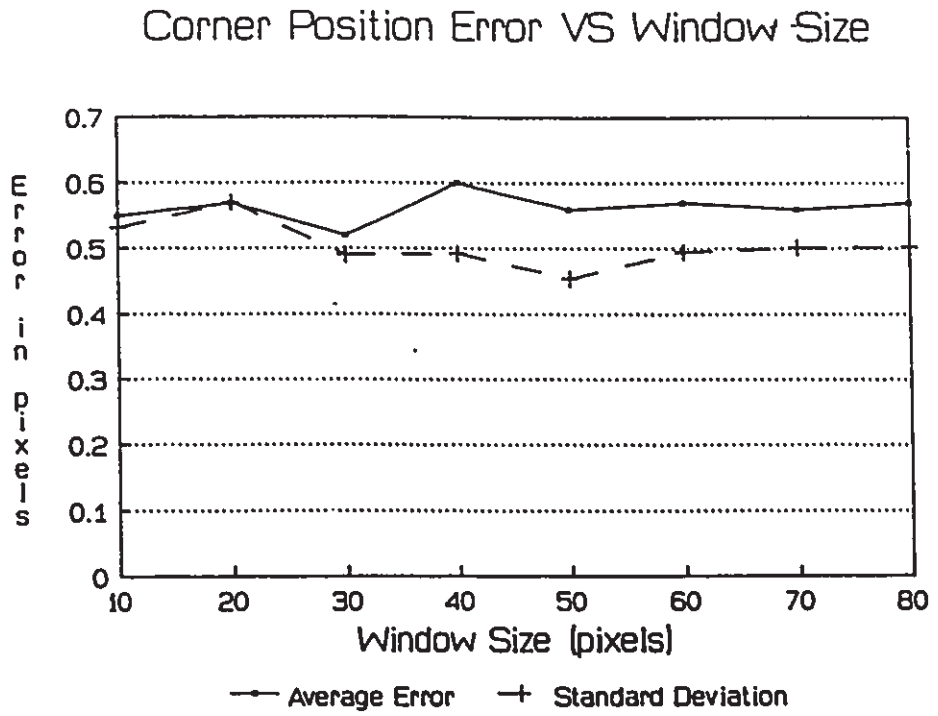


Figure 3.13. Average Corner Location Error vs Window Size.

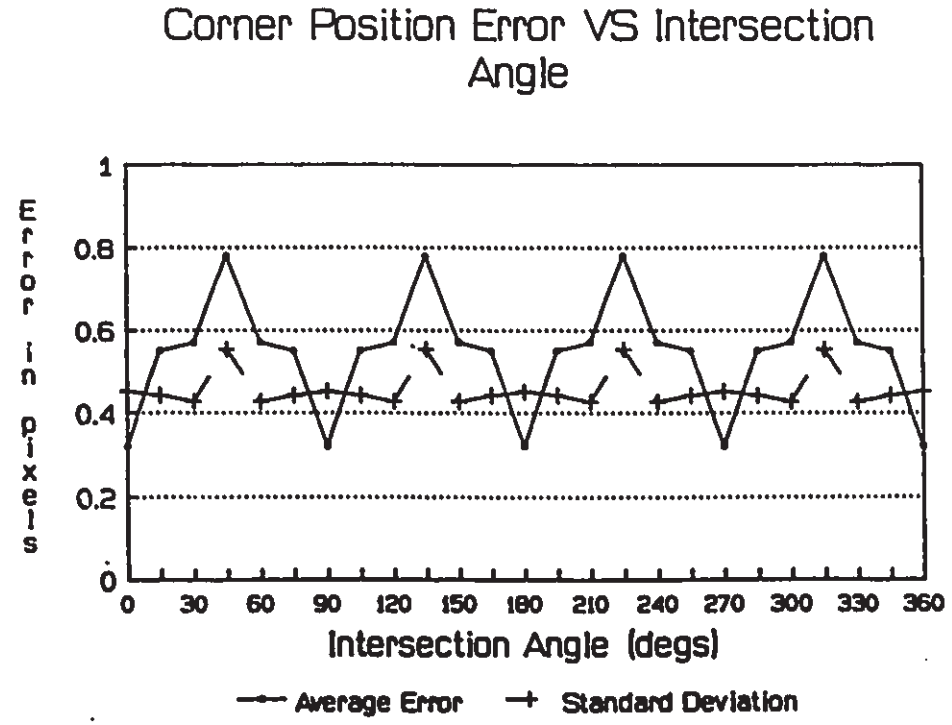


Figure 3.14. Average Corner Location Error vs Intersection Angle.

degrees at intersection angles of 0 and 45 degrees are studied. The actual and gathered data for the two cases are reported in Table 3.1. The data reveals that when the corner intersects at 0 degrees the Y moment error is smaller. This may be attributed to the symmetry of the two corner edges about the horizontal line passing through the Y centroid. This symmetry causes the errors introduced by the edge digitization of the top and bottom edge to cancel each other out for the Y moment data. By contrast the 45 degree intersection leads to errors in both the X and Y moments.

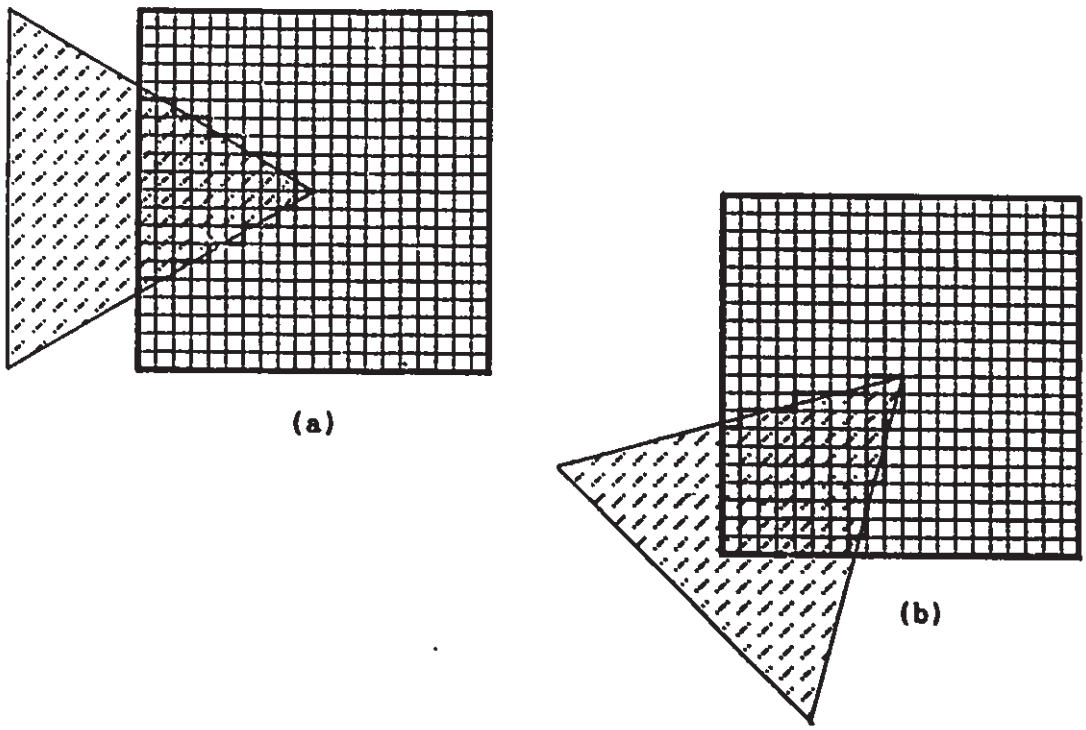


Figure 3.15. Digitized 60 Degree Shape Corner Intersecting Window at a) 0 degrees and b) 45 Degrees.

Intersection Angle (degs)	Area		X-moment		Y-moment	
	Actual	Gathered	Actual	Gathered	Actual	Gathered
0	57.74	58	202.09	230	606.27	609
45	72.98	74	339.6	379	339.6	379

Table 3.1.Table of Gathered and Actual Data for the Corners of Figure 3.15.

3.3 Photogrammetric Solution

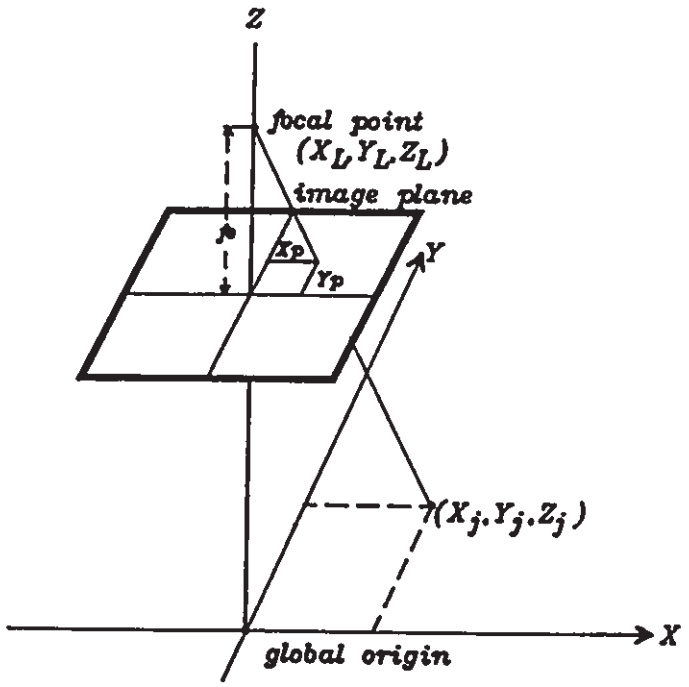
The purpose of the photogrammetric solution is to determine the three dimensional position and orientation of an object from the perspective projection points of a known pattern of object features on the image plane. The traditional photogrammetric approach treats the object plane and the image plane within a global coordinate systems and uses a least squares approach to derive estimates of the x,y,z location and pitch, yaw and roll rotations of the object plane that would result in the observed projection. A detailed description of the traditional approach is given in section 3.3.1.

A new approach which uses the geometric relationship between object points and image points in introduced in this thesis. Instead of estimating the position elements, an analytic method of determining the object position and rotation is described. This method is first derived for a square planar point pattern and then for a general four point planar

pattern.

3.3.1 Traditional Least Squares Solution

The traditional least squares approach begins with the fundamental photogrammetric relations. These fundamental relations (known as the



(X_j, Y_j, Z_j) – coordinates of object point at nominal position

(X_p, Y_p) – image projection of object point

(X_L, Y_L, Z_L) – coordinates of camera's principal point

Figure 3.16. Relationship Between Parallel Shape and Image Planes.

collinearity equations) are derived by relating points on the object and image planes under X,Y,Z translations and pitch, yaw, roll rotations. The derivation presented here is similar to that presented by Moffitt and Mikhail [3.2]. The two planes and the associated variables are illustrated in Figure 3.16 and the symbols defined for the derivation are as follows:

X_j, Y_j, Z_j	-	coordinates of a control point j on the object
X_L, Y_L, Z_L	-	coordinates of the perspective centre or focal point w.r.t. object plane coordinate system
x_j, y_j	-	coordinates of the projection of an object control point j on the image plane
Ω, θ, Φ		pitch yaw and roll of camera axes w.r.t. the object axes
f_e	-	camera effective focal length

Formulation of the fundamental relations is derived from the similar triangles present in single image perspective projections illustrated in Figure 3.16. If the object is at a nominal position and not rotated then the coordinates of a control point in object space coordinates (X_j, Y_j, Z_j) are related to the projection point in camera coordinates (x_j, y_j) by the equations :

$$(X_j - X_L) / (Z_j - Z_L) = x_j / f_e \qquad [3.14]$$

$$(Y_j - Y_L) / (Z_j - Z_L) = y_j / f_e \qquad [3.15]$$

If rotations are present then the image axes must first be rotated before equations 3.14 and 3.15 are employed. The effects of rotations are included by placing a second set of axes (x_t, y_t, z_t in Figure 3.17) at the principal point and rotating the object point relative to these axes. The

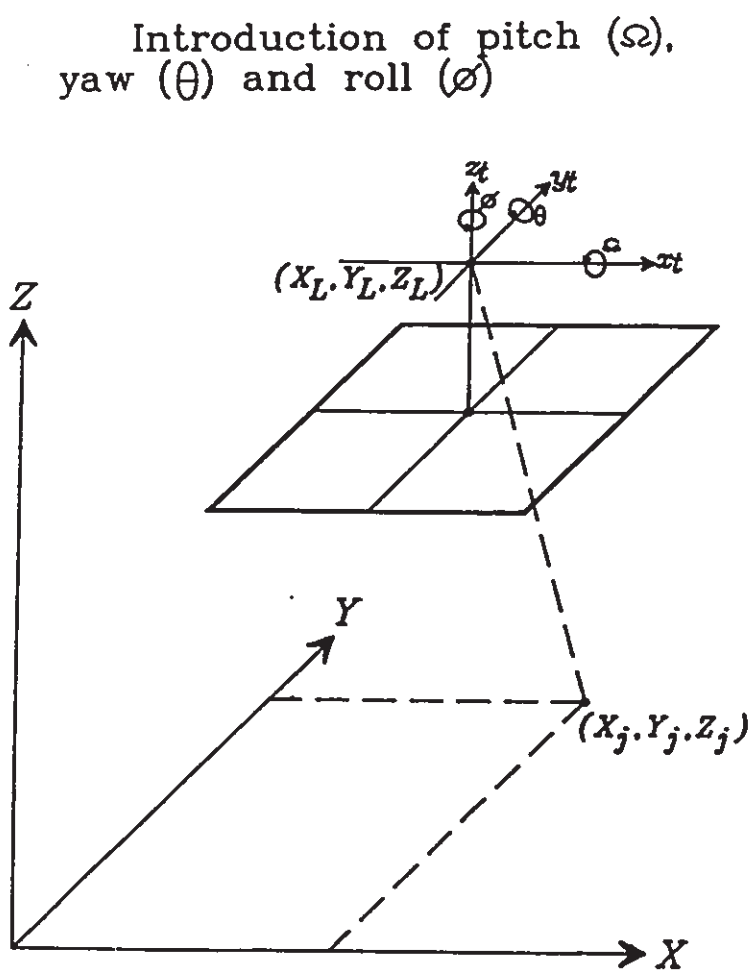


Figure 3.17. Relationship Between Image and Shape Coordinates when Rotational Axes are Added.

relative coordinates of the control point (X_j, Y_j, Z_j) w.r.t. this new set of axes are $(X_j - X_l, Y_j - Y_l, Z_j - Z_l)$. The coordinates relative to the axes if they are rotated first by yaw (θ), then by pitch (Ω), and finally by roll (Φ) can be found by multiplying the relative coordinates of the control point by the transformation matrix M:

$$\begin{bmatrix} X^* \\ Y^* \\ Z^* \end{bmatrix} = M \begin{bmatrix} X_j - X_L \\ Y_j - Y_L \\ Z_j - Z_L \end{bmatrix}$$

The equations for (X^*,Y^*,Z^*) are substituted for $(X_j - X_L, Y_j - Y_L, Z_j - Z_L)$ in equations 3.14 and 3.15 yielding the collinearity equations which relate object and image points in a global coordinate system whose origin is at the principal point:

$$x_j = f_e \left[\frac{m_{11}(X_j - X_L) + m_{12}(Y_j - Y_L) + m_{13}(Z_j - Z_L)}{m_{31}(X_j - X_L) + m_{32}(Y_j - Y_L) + m_{33}(Z_j - Z_L)} \right]$$

$$y_j = f_e \left[\frac{m_{21}(X_j - X_L) + m_{22}(Y_j - Y_L) + m_{23}(Z_j - Z_L)}{m_{31}(X_j - X_L) + m_{32}(Y_j - Y_L) + m_{33}(Z_j - Z_L)} \right] \qquad [3.16]$$

where:

$$m_{11} = \cos(\Phi)\cos(\Omega)$$

$$m_{12} = \sin(\theta)\sin(\Omega)\cos(\Phi) + \cos(\theta)\sin(\Phi)$$

$$m_{13} = \sin(\theta)\sin(\Phi) - \cos(\theta)\sin(\Omega)\cos(\Phi)$$

$$m_{21} = -\cos(\Omega)\sin(\Phi)$$

$$m_{22} = \cos(\theta)\cos(\Phi) - \sin(\theta)\sin(\Omega)\sin(\Phi)$$

$$m_{23} = \sin(\theta)\cos(\Phi) + \cos(\theta)\sin(\Omega)\sin(\Phi)$$

$$m_{31} = \sin(\Omega)$$

$$m_{32} = -\sin(\theta)\cos(\Omega)$$

$$m_{33} = \cos(\theta)\cos(\Omega)$$

When four control points are utilized 8 collinearity equations result, 2 for each point. Solution of these equations yields the relative camera-object position (X_L, Y_L, Z_L) and orientation (θ, Ω, Φ).

The equations are non-linear due to the transcendental functions and thus are not easily solved. The traditional approach is to develop a least squares solution. The collinearity equations can be written as:

$$x_j = F_{xj}\ (X_L, Y_L, Z_L, \theta, \Omega, \Phi)$$

$$y_j = F_{yj}\ (X_L, Y_L, Z_L, \theta, \Omega, \Phi)$$

If the six estimates of the position variables are introduced then:

$$x_j = x_{j0} + dF_{xj}$$

$$y_j = y_{j0} + dF_{yj} \qquad [3.17]$$

where:

dF_{xj} and dF_{yj} are the linear Taylor series expansions of F_{xj} and F_{yj}

x_{j0} and y_{j0} are computed using the initial estimates of the six position variables and the collinearity equations

Thus for each pair of collinearity equations:

$$x = x_0 + B_{11}\,\delta X_L + B_{12}\,\delta Y_L + B_{13}\,\delta Z_L + B_{14}\,\delta\theta + B_{15}\,\delta\Omega + B_{16}\,\delta\Phi$$

$$y = y_0 + B_{21}\,\delta X_L + B_{22}\,\delta Y_L + B_{23}\,\delta Z_L + B_{24}\,\delta\theta + B_{25}\,\delta\Omega + B_{26}\,\delta\Phi$$

where:

B_{ij} - are the partial derivatives calculated at the initial estimate point.

When four or more object control points are used, the least squares estimate seeks to select position corrections $(\delta X_L, \delta Y_L, \delta Z_L, \delta\theta, \delta\Omega, \delta\Phi)$ to minimize the sum of the squares of the error terms in equation 3.18.

$$e = v_{x1}^2 + v_{y1}^2 + v_{x2}^2 + v_{y2}^2 + v_{x3}^2 + v_{y3}^2 + \ldots \qquad [3.18]$$

where:

v_{xi}, v_{yi} (i = 1..n) are the residuals of the equations.

If the observations are equally weighted, the least squares normal equation takes the form:

$$B^t B u = B^t f$$

where:

B - is the n x 6 matrix of partial differentials for each of the linearized collinearity equations at the estimated position.

u - is the n x 1 vector of corrections

f - is the n x 1 vector of observations

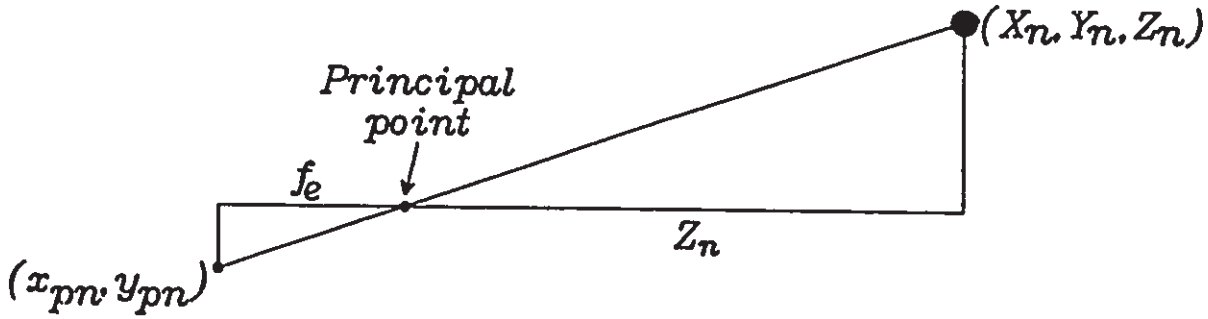
Solving for u yields:

$$u = (B^t B)^{-1} B^t f$$

The least squares solution is found by finding the inverse of the matrix $(B^t B)$ and multiplying through the left side of the equation. The corrections can then be added to the initial estimate to form a new estimate which can be checked for accuracy by using the collinearity equations. If the difference between the computed observations and the actual observations is above a desired threshold the least squares estimate can be improved by repeating it with the new estimates. Thus the solution can be iterated until the desired accuracy is achieved.

3.3.2 Analytic Solution for a Square Target

An analytic solution to the photogrammetry problem can be found if the square planar target is considered in its geometric context. The photogrammetric solution is found in two stages: first the 3-D coordinates of each of the target dots is found, then the centroid of the target and



(X_n, Y_n, Z_n) – *world coordinates of target dot w.r.t. camera*
(x_{pn}, y_{pn}) – *observed camera coordinates of target dot*
f_e – *effective camera constant*

$$x_{pn} = f_e/Z_n * X_n$$
$$y_{pn} = f_e/Z_n * Y_n$$

Figure 3.18. Image Target Dot Coordinates and Global Coordinates Equations.

its rotational orientation are derived. Each of the target dots' coordinates is related to the coordinates of the projected camera dot by the equations given in Figure 3.18. Thus for a target of four dots, eight equations result:

$$x_{p1} = f_e/Z_1 * X_1 \qquad [3.19]$$

$$y_{p1} = f_e/Z_1 * Y_1 \qquad [3.20]$$

$$x_{p2} = f_e/Z_2 * X_2 \quad [3.21]$$

$$y_{p2} = f_e/Z_2 * Y_2 \quad [3.22]$$

$$x_{p3} = f_e/Z_3 * X_3 \quad [3.23]$$

$$y_{p3} = f_e/Z_3 * Y_3 \quad [3.24]$$

$$x_{p4} = f_e/Z_4 * X_4 \quad [3.25]$$

$$y_{p4} = f_e/Z_4 * Y_4 \quad [3.26]$$

Since the target is made up of four planar points arranged as a square, the centroid of the square can be found by using either pair of opposite corners ((1 and 4) OR (2 and 3)). From this information three more equations can be derived.

For the X coordinate of the centroid of the square X_c:

$$X_c = (X_1 + X_4)/2 \quad \text{OR}$$

$$X_c = (X_2 + X_3)/2$$

Therefore:

$$X_1 + X_4 = X_2 + X_3 \quad [3.27]$$

Similarly:

$$Y_1 + Y_4 = Y_2 + Y_3 \quad [3.28]$$

$$Z_1 + Z_4 = Z_2 + Z_3 \quad [3.29]$$

These simple relationships result in a linear system of 11 equations and 12 unknowns. A twelfth equation which incorporates the size of the square is the magnitude of one side of the square:

$$2a = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2 + (Z_2 - Z_1)^2}$$

where:

2a - length of a side

Substitutions for X_1, X_2, Y_1 and Y_2 in the above equation in terms of Z_1 and Z_2 are obtained by rewriting equations 3.19 - 3.26 (i.e. $X_n = Z_n/f_c * X_{pn}$ etc.). This results in an equation of the form:

$$2a = \sqrt{A*Z_1^2 - 2B*Z_1*Z_2 + C*Z_2^2} \qquad [3.30]$$

where:

$$A = 1 + \frac{x_{p1}^2 + y_{p1}^2}{f_c^2}$$

$$B = 1 + \frac{x_{p1}x_{p2} + y_{p1}y_{p2}}{f_c^2}$$

$$C = 1 + \frac{x_{p2}^2 + y_{p2}^2}{f_c^2}$$

Equation 3.30 is then further simplified by finding a linear equation relating Z_1 and Z_2. This is accomplished by substituting for the X_n and Y_n values in the linear equations 3.27 - 3.29 using rearranged forms of equations 3.19 - 3.26 resulting in three linear equations with four unknowns $Z_1 - Z_4$. Gaussian elimination results in an equation relating Z_1 and Z_2 as follows:

$$Z_1 = D * Z_2 \qquad [3.31]$$

where:

$$D = \frac{y_{p3}(x_{p2} - x_{p4}) + y_{p4}(x_{p3} - x_{p2}) + y_{p2}(x_{p4} - x_{p3})}{y_{p3}(x_{p1} - x_{p4}) + y_{p4}(x_{p3} - x_{p1}) + y_{p1}(x_{p4} - x_{p3})}$$

Substituting into 3.30 yields:

$$(A*D^2 - 2*B*D + C) * Z_2^2 = (2a)^2$$

Therefore:

$$Z_2 = \sqrt{(2a)^2 / (A*D^2 - 2*B*D + C)} \qquad [3.32]$$

The value of Z_1 can then be found using equation 3.31. The values of Z_3 and Z_4 can be found by manipulating equations 3.27 - 3.29 and the X and Y values can be found by substituting the Z values into equations 3.19 - 3.26. The target's translation and rotation can then be found using the three dimensional points previously derived. The target centroid is found from:

$$X_c = (X_1 + X_4)/2 \qquad [3.33]$$

$$Y_c = (Y_1 + Y_4)/2 \qquad [3.34]$$

$$Z_c = (Z_1 + Z_4)/2 \qquad [3.35]$$

The target rotations are found by finding the rotations of a pair of orthogonal vectors from the centroid to the middle of the right side and from the centroid to the middle of the top of the target pattern as shown

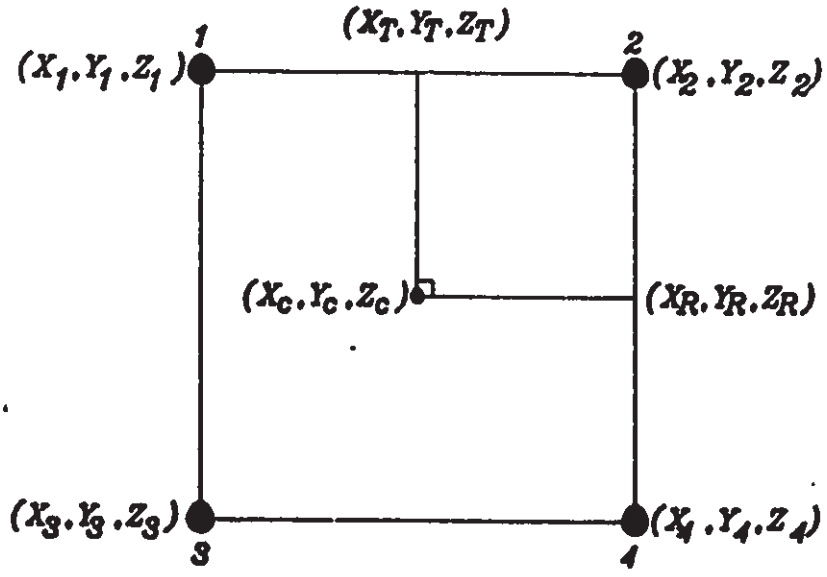


Figure 3.19. Derivation of Vectors for Rotation Calculations.

in Figure 3.19. First the rotated orthogonal vectors on the translated and rotated target are found using equations 3.36 and 3.37.

$$\begin{aligned} X_R &= (X_2 + X_4)/2 - X_c \\ Y_R &= (Y_2 + Y_4)/2 - Y_c \\ Z_R &= (Z_2 + Z_4)/2 - Z_c \end{aligned} \qquad [3.36]$$

$$\begin{aligned} X_T &= (X_2 + X_1)/2 - X_c \\ Y_T &= (Y_2 + Y_1)/2 - Y_c \\ Z_T &= (Z_2 + Z_1)/2 - Z_c \end{aligned} \qquad [3.37]$$

Then the roll (Φ) is calculated:

$$\Phi = \text{Tan}^{-1} (Y_R/X_R) \qquad [3.38]$$

The two vectors are then rotated by the roll angle yielding $(X_{R\Phi}, Y_{R\Phi}, Z_{R\Phi})$ and $(X_{T\Phi}, Y_{T\Phi}, Z_{T\Phi})$. The pitch angle ($\Omega$) is then calculated:

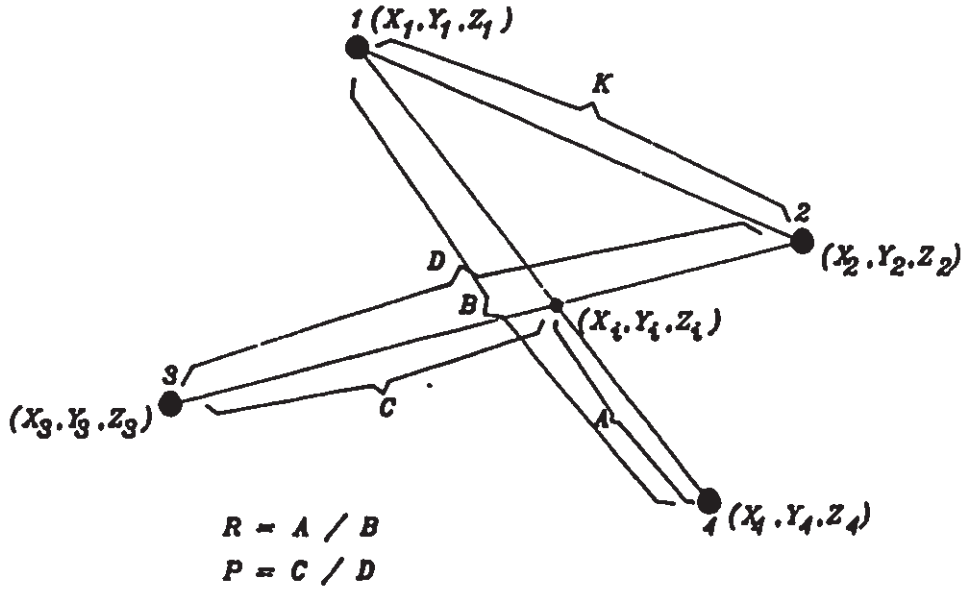
$$\Omega = \text{Tan}^{-1} (Z_{R\Phi}/X_{R\Phi}) \qquad [3.39]$$

The two vectors are then rotated by the pitch angle yielding $(X_{R\Phi\Omega}, Y_{R\Phi\Omega}, Z_{R\Phi\Omega})$ and $(X_{T\Phi\Omega}, Y_{T\Phi\Omega}, Z_{T\Phi\Omega})$. The yaw angle ($\theta$) can then be calculated:

$$\theta = \text{Tan}^{-1} (Z_{T\Phi\Omega}/Y_{T\Phi\Omega}) \qquad [3.40]$$

3.3.3 Analytic Solution for an Arbitrary Planar Four Point Pattern

A general solution for four point planar patterns can be derived with the constraint that the four planar points be non-collinear. This solution is derived by modifying equations 3.27 - 3.29. For the square target these equations assume that the intersection of the two lines joining opposite corners is midway between the opposite corners. For an arbitrary pattern of four points as depicted in Figure 3.20 the ratio of



$$R = A / B$$

$$P = C / D$$

$$X_i = R*(X_1 - X_4) + X_4 = P*(X_2 - X_3) + X_3$$
$$Y_i = R*(Y_1 - Y_4) + Y_4 = P*(Y_2 - Y_3) + Y_3$$
$$Z_i = R*(Z_1 - Z_4) + Z_4 = P*(Z_2 - Z_3) + Z_3$$

Figure 3.20. Derivation of Ratios for General Analytic Solution for a Four Point Planar Pattern.

the distance from a corner to the intersection point to the distance between the corners is used. The modified versions of equations 3.27 - 3.29 are:

$$R*X_1 + (1 - R)*X_4 = (1 - P)*X_3 + P*X_2 \qquad [3.41]$$

$$R*Y_1 + (1 - R)*Y_4 = (1 - P)*Y_3 + P*Y_2 \qquad [3.42]$$

$$R*Z_1 + (1 - R)*Z_4 = (1 - P)*Z_3 + P*Z_2 \qquad [3.43]$$

where:

R and P are the intersection point ratios for the lines joining corner 1 and 4 and corner 2 and 3 as shown in Figure 3.20.

As was the case for the square target, the X and Y values are substituted using equations 3.19 - 3.26 and equations 3.41 - 3.43 are manipulated to get an equation relating Z_1 and Z_2 as follows:

$$Z_1 = D_1 * Z_2 \qquad [3.44]$$

where:

$$D_1 = \frac{P}{R} * \frac{y_{p3}(x_{p2} - x_{p4}) + y_{p4}(x_{p3} - x_{p2}) + y_{p2}(x_{p4} - x_{p3})}{y_{p3}(x_{p1} - x_{p4}) + y_{p4}(x_{p3} - x_{p1}) + y_{p1}(x_{p4} - x_{p3})}$$

The value of Z_2 can then be found using an equation similar to 3.32 where the absolute distance between corners 1 and 2 (K) is substituted for 2a.

$$Z_2 = \sqrt{K^2 / (A*D_1^2 - 2*B*D_1 + C)} \qquad [3.45]$$

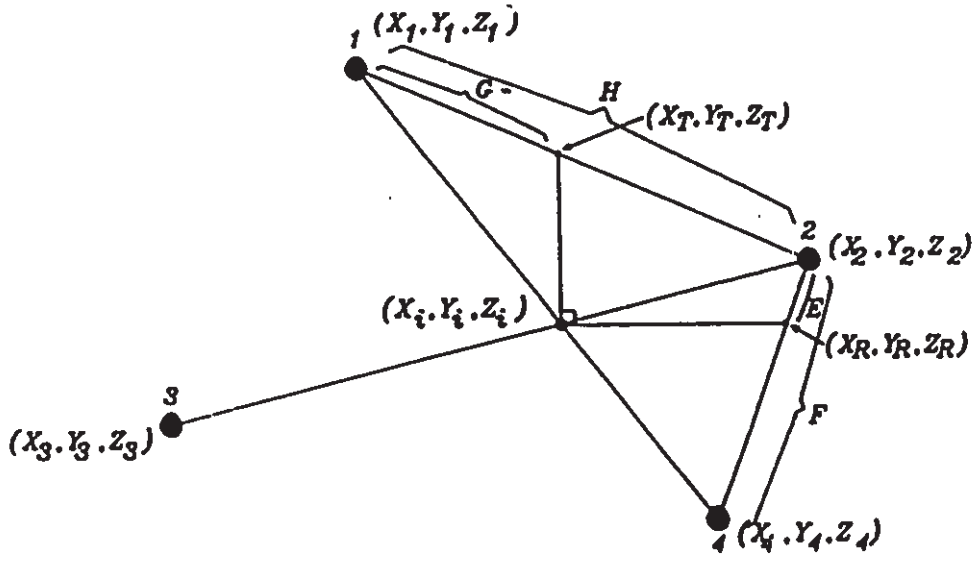
The three dimensional corner points can then be found by back substitution. Derivation of the translation and rotation are then carried out using the derived corner points. A straight forward manner is to derive the six values relative to the intersection point of the lines joining the corners. First the intersection point (X_i, Y_i, Z_i) is found:

$$X_i = R * (X_1 - X_4) + X_4 \quad [3.46]$$

$$Y_i = R * (Y_1 - Y_4) + Y_4 \quad [3.47]$$

$$Z_i = R * (Z_1 - Z_4) + Z_4 \quad [3.48]$$

Finally, a set of orthogonal vectors on the planar shape are found in order to determine the pitch, yaw and roll. Figure 3.21 shows the two vectors on an arbitrary four point pattern and the two ratios which must



$$T = G / H$$

$$S = E / F$$

Figure 3.21. Derivation of Ratios to Obtain Vectors for Rotation Calculation for the General Four Point Solution.

be used in order to find these vectors on a translated and rotated pattern. The equations for the two vectors are:

$$\begin{aligned} X_R &= S*(X_3 - X_2) + X_2 - X_i \\ Y_R &= S*(Y_4 - Y_2) + Y_2 - Y_i \\ Z_R &= S*(Z_4 - Z_2) + Z_2 - Z_i \end{aligned} \quad [3.49]$$

$$\begin{aligned} X_T &= T*(X_2 - X_1) + X_1 - X_i \\ Y_T &= T*(Y_2 - Y_1) + Y_1 - Y_i \\ Z_T &= T*(Z_2 - Z_1) + Z_1 - Z_i \end{aligned} \qquad [3.50]$$

The rotations are then calculated in the same manner as for the square target.

3.4 Practical Considerations

Implementation of the tracking system requires the consideration of a number of practical issues. Amongst these are lighting, processing speed and camera selection to achieve the desired accuracy. During the course of this research, hardware capable of collecting the necessary data to perform real time tracking of multi-dot targets was developed in conjunction with Diffracto Canada Ltd., Windsor, Ontario. This single multibus board based on the TMS320C25 processor is capable of collecting data for the target dot solution at 60Hz (i.e. video field rates). This hardware implementation represents significant improvements over the original dot tracking hardware developed for the NRC tracking system [1.13]. The one multibus board solution is a fivefold decrease in hardware. In addition, this implementation does not require feature windows in order to track the target dots; the target dots can be found at any location within the image field. The power of the processor also makes it possible to refine the dot edge locations to below the pixel level to enhance the accuracy of the dot centroid calculation. Furthermore, the image processing also provides a measure of dot circularity and height to assist in distinguishing between the target dots and background data. The practical considerations relevant to tracking

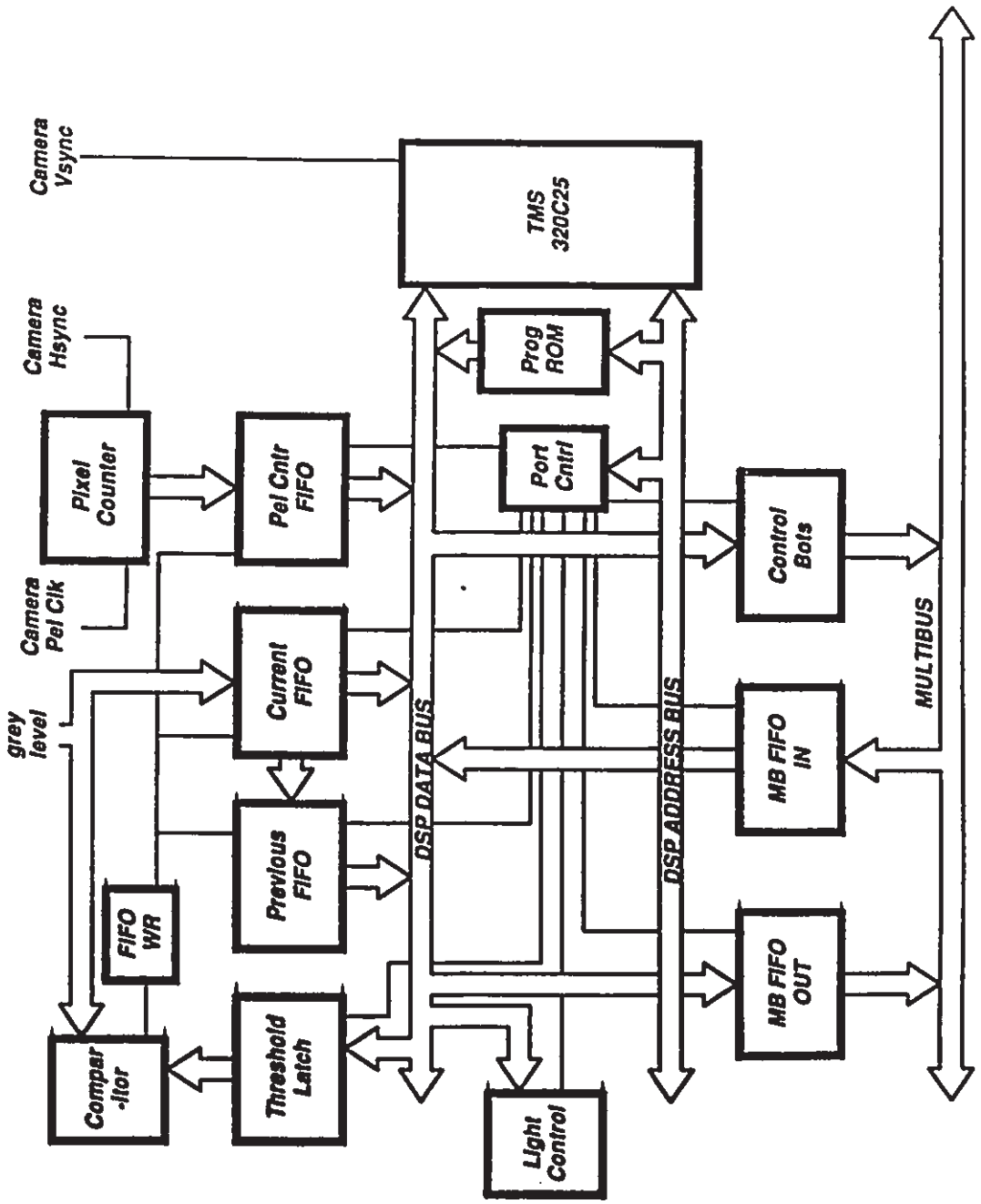


Figure 3.22. Block Diagram of Field Rate Data Collection Hardware for Target Dot Tracking System.

systems are discussed in the context of this target dot implementation.

A block diagram of the board is illustrated in Figure 3.22. The board functionality can be divided into three major tasks: 1) camera interface ; 2) image processing; and 3) multibus interface. The camera interface employs the vertical sync pulse to synchronise the processing operation with the camera fields. The horizontal sync and pixel clock drive a counter which indicates the current pixel being processed. Target dots are separated from the background by a combination of lighting control and a thresholding operation. This lighting control mechanism is essential to the successful operation of the system. Generally, on a factory floor, vision systems have little control over ambient lighting conditions. The lighting control in this implementation employs infra-red emitters and an infra-red sensitive camera combined with an narrow band infra-red filter in order to minimize the effects of ambient lighting. The duty cycle of the infra-red emitters is controlled by the processor in order to adjust the light level for varying ambient conditions and for the range of movement of the target.

The camera chosen for the implementation was required to be suitable both in terms of its resolution and its sensitivity to infra-red light. Further gains in resolution were obtained by estimating the position of the edge of the target dots between pixels based on the difference in grey levels of the pixels preceding and following the threshold position. This difference is affected by the lighting and is a factor used in adjusting the lighting. When higher resolution cameras are employed higher data rates result. To buffer the data from the camera a

number of FIFOs were employed. This allows changes in adjacent pixels to be recorded without the necessity of having the processor read the pixel positions and associated grey levels at video rates.

The computation necessary and the potentially large amount of data required a high speed processor capable of performing fast multiplications in order to calculate the moments. Target dot position calculation is done at field rates in order to provide a reasonable feedback loop for a robot. The TMS320C25 was chosen for these reasons. The algorithm used to compute the moments of the target dots is a modified version of the Capson algorithm [2.4]. In this implementation, the vertex lists are not maintained, but the same connectivity rules are followed. For each dot, the area, x moment, y moment, dot height, and a measure of circularity are found during the field scan time. These can then be reported to the controlling processor at the end of each field.

As was the case for the camera interface, communication with the controlling processor over the multibus is accomplished using FIFOs. In this way the TMS320C25 processor does not have to synchronise with the controlling processor in order to communicate. The results of field analysis are put in the FIFO and the controlling processor is notified by control bits. The controlling processor can then retrieve the data as it is needed.

To implement the vision system using corners similar practical considerations must be addressed. Object areas must be separated from the background by a combination of lighting and thresholding or edge detection. The camera chosen must meet the necessary resolution and

sensitivity requirements to provide the accuracy required for robot guidance. Finally, the system processing must be fast enough to be an provide effective feedback.

CHAPTER 4

TRACKING PERFORMANCE RESULTS

This chapter is devoted to testing the various aspects of the tracking system. Simulations were carried out to determine the merits of the tracking phases as well as the accuracies attainable by the system. The simulations were carried out using a PC equipped with a 3-plane graphics card and a math co-processor. The perspective projections of planar shapes moved in space in X, Y and Z and rotated by pitch, yaw and roll were simulated graphically and identification and tracking computations were carried out employing these simulated projections in much the same way and would have resulted if an image frame store were employed.

In section 4.1, projection identification and lock-on experimentation is described. Section 4.2 compares the results obtained using the new photogrammetric with those obtained using the single iteration least squares solution developed by the NRC for a square target. The next section contains the results of simulations with the new photogrammetric solution using arbitrary target patterns. This has been carried out to compare the effects of different patterns on the accuracy of the solution computed using the general analytic solution. Next, the corner detectors were tested in tracking simulations to determine their

accuracy: the results are compared with target tracking. Finally, simulations were carried out to determine the effects of camera resolution and distance from the camera.

4.1 Projection Identification and Lock-On Performance

Projection identification and lock-on tests were performed using the 20 shapes introduced by Ma, Wu and Lu and shown in Figure 2.20. The system was trained for each shape as described in section 3.1. In each case the effective focal length was set at 100mm and the nominal distance was set at 100mm (for the simulation, 1 pixel represents 1mm at a distance of 100mm from the camera). After the training phase for each shape the identification system was tested by generating 1000 projections of random pitch, yaw and roll. The range for pitch and yaw was -50 to 50 degrees and the range for roll was 0 to 90 degrees. The match was considered successful if the selected matching projection was within 5 degrees for pitch and yaw of the actual projection. For each shape the average and standard deviation of the absolute pitch and yaw errors was calculated for the successful matches. The technique described in section 2.3 was employed to estimate the roll of the projections. The average and standard deviation for the estimated roll was also computed. The results of these simulations are given in Table 4.1.

The results of the projection matching vary from 25.5% successful matching for shape 11 to 100% for shapes 6,7,16 and 18. The variation can be attributed to the size of the projections and the presence of ambiguous

Results For Fourier Descriptor Matching Tests							
Shape No.	% correct matches	Pitch Error		Yaw Error		Roll Error	
		Avg.	sigma	Avg.	sigma	Avg.	sigma
1	95.2	3.16	0.19	2.14	0.13	16.34	4.40
2	97.2	2.33	0.13	1.87	0.10	20.54	4.59
3	96.5	2.72	0.16	2.12	0.12	22.08	5.09
4	79.3	3.97	0.23	2.52	0.15	21.77	5.64
5	68.3	4.56	0.29	3.30	0.26	36.25	2.51
6	100	1.61	0.08	1.56	0.08	5.99	1.78
7	100	1.79	0.12	1.93	0.12	8.62	2.98
8	92.5	2.64	0.10	2.69	0.18	10.04	1.33
9	75.9	2.52	0.10	1.99	0.08	6.77	1.01
10	60.2	1.81	0.15	2.59	0.21	16.70	5.78
11	25.5	3.08	0.20	3.84	0.29	33.55	2.89
12	92.8	1.41	0.09	1.38	0.09	22.95	5.73
13	96.4	1.80	0.12	2.18	0.13	19.47	5.08
14	84.0	3.13	0.09	2.75	0.09	5.65	0.41
15	99.5	2.14	0.13	1.70	0.12	14.64	4.28
16	100	1.89	0.10	1.80	0.11	6.07	2.46
17	97.7	1.42	0.08	1.83	0.11	4.58	1.01
18	100	1.35	0.08	1.75	0.10	11.24	3.94
19	99.5	2.01	0.13	1.44	0.08	10.34	3.55
20	92.8	2.89	0.17	3.39	0.18	26.95	5.11

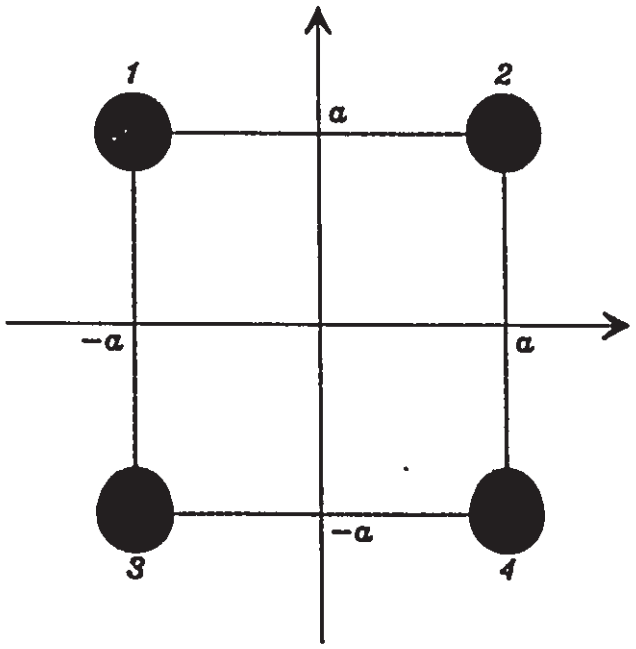
Table 4.1. Results Obtained from Projection Identification Experiments (errors in degrees)

projections for some of the shapes. The first 4 shapes which are identical except for size reveal the effects of this factor on success. The effects of ambiguous projections are most evident in the case of the octagon shape (11) for which a large number of ambiguous projections exist. Shapes displaying rotational symmetry such as 5 or 14 also lead to ambiguous projections. Shapes which do not display any rotational symmetry and which consist of 4 or more corners such as 6,7,16 or 18 yielded 100% correct matching for the simulations conducted.

The pitch and yaw errors are below 3 degrees in most cases and have standard deviations of less than 0.2 degrees. This allows the corner position estimates to be close to the actual positions for window placement provided the roll can be accurately estimated. However, average roll error varies from 4.58 degrees for shape 17 to 36.25 degrees for shape 5. Shapes displaying a high roll error could lead to misplacement of the windows. The high roll error could be caused by the use of coefficients having a small magnitude in the roll calculation. The use of only 8 coefficients limits the choice of coefficients for roll calculation. This problem might be solved if instead of using the first eight coefficients for shape identification, the table of descriptor sets is analyzed and the 8 coefficients displaying the highest average magnitudes are employed.

4.2 Target Tracking Using Geometric Solution vs Least Squares

The geometric solution was compared to the least squares solution using simulated perspective views of the target depicted in Figure 4.1. The effective focal length of the system was set at 100mm. The size of the target side (2a) was set at 80mm with each dot of diameter 20mm. For each



a = half the side of the square

Figure 4.1. Target used to Compare Analytic vs Least Squares Solution

test the target's initial position was 100mm from the camera with pitch, yaw and roll angles of zero.

The equations developed by the NRC using the least squares approach are given in Appendix A. These equations were developed using the following assumptions:

1) The pitch, yaw and roll angles are small and can thus their cosines can be estimated as 1 and their sines can be estimated as

the angles.

2) The first derivatives of the collinearity equations (to be used in the Taylor series expansion) with respect to x and y were estimated using the slope of the ray passing through the focal point and the object point.

Tests were carried out to determine the effects of rotations and translations on the relative accuracy of the solutions computed using the least squares and the geometric solution. Independent tests for target movement along the Z axis, pitch rotation, yaw rotation and roll rotation were performed. The results of these tests show both the accuracy of the solution and the effects of pitch and yaw rotations on the estimated X, Y and Z position calculations. Combined pitch, yaw and roll rotation and X,Y and Z translation tests were also performed to compare the accuracies of the two solutions. In each case the errors reported are the average absolute error between the computed values and the actual position of the shape.

Figure 4.2 shows the errors in Z position when the target is moved along the Z axis from the nominal 100mm position to 145mm with no rotations. The results obtained by the least squares solution and the geometric solution seem to be similar. The digitization error is evident in the cyclic nature of the error. There is also evidence of a trend toward increasing error as the target is moved further away. Since the target becomes smaller as it is translated, the effects of digitization are enhanced.

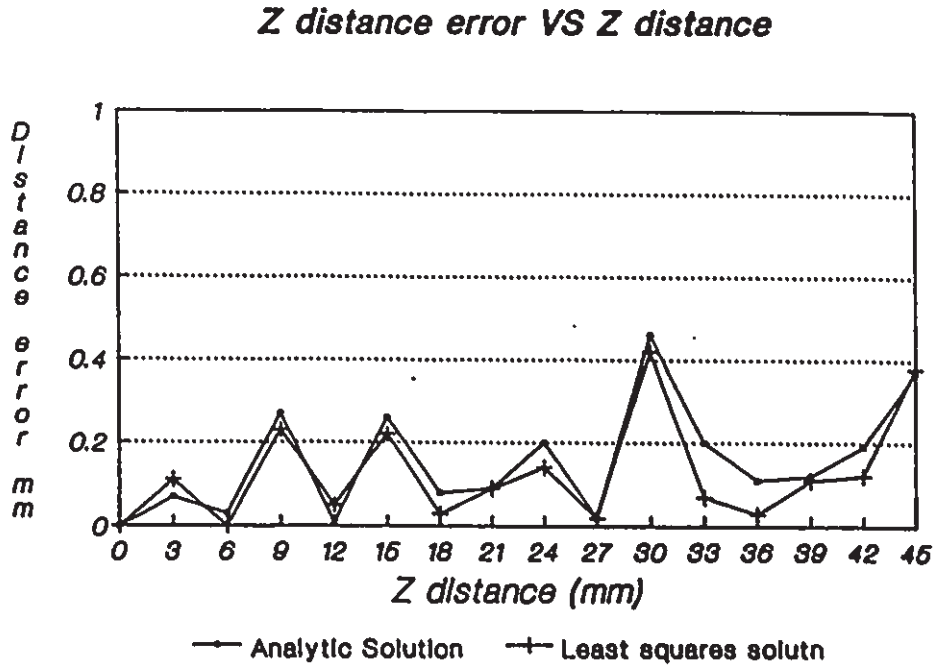
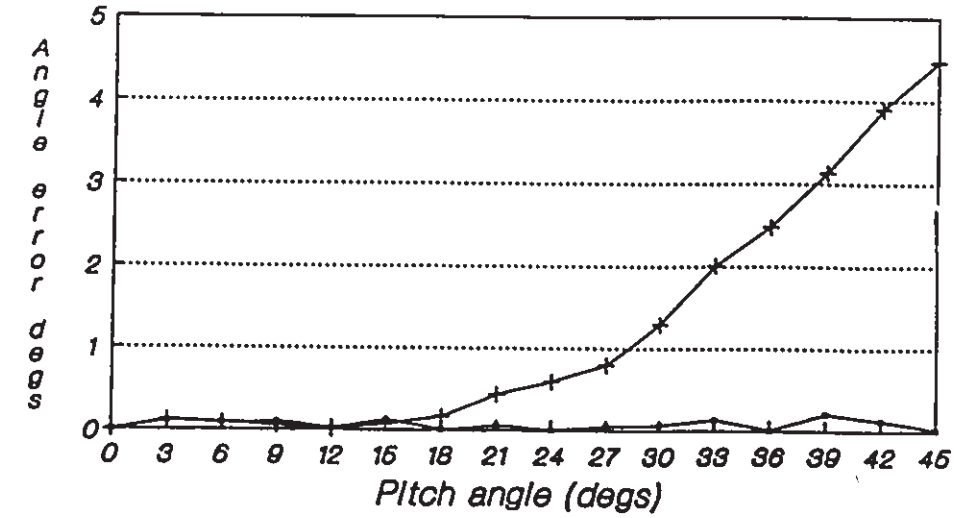
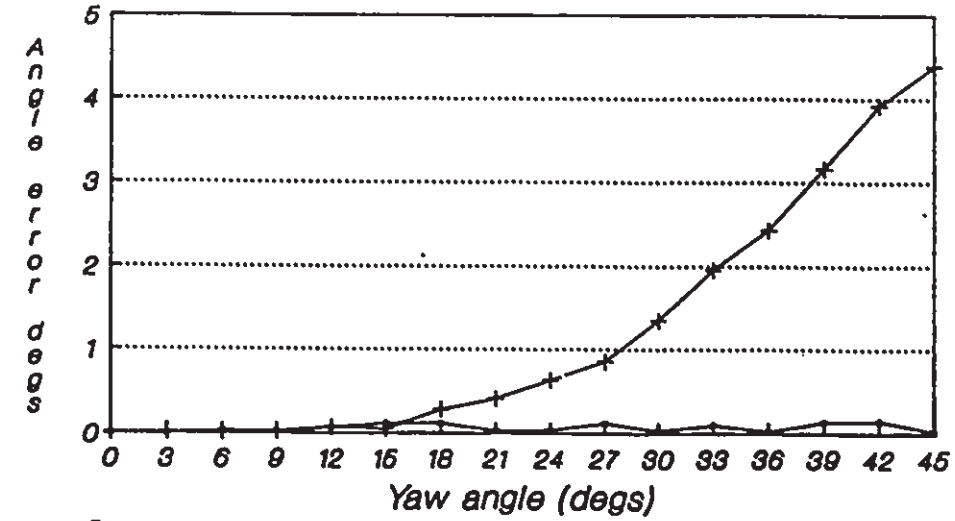


Figure 4.2. Effects of Z Translations on Z Position Calculation.

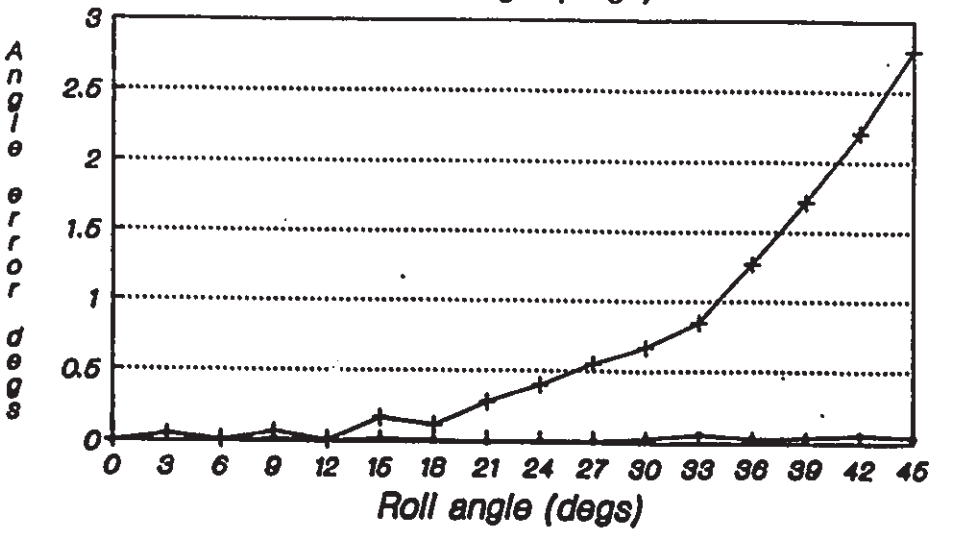
Figure 4.3 (a), (b) and (c) show the results obtained when the target is pitched from 0 to 45 degrees,yawed from 0 to 45 degrees and rolled from 0 to 45 degrees respectively. For the pitch and yaw tests the errors obtained from the geometric solution are similar to the results obtained by the least squares solution up to approximately 15 degrees of rotation. After this point the error results obtained by the geometric solution remain small while the error obtained from the least squares solution rises. This is expected since the derivation of least squares solution assumes that small rotations are present. The results for roll angle also show a similarity in error for both techniques at small angles whereas the least squares error begins to rise at approximately 12 degrees.



(a)



(b)



(c)

Figure 4.3. Target Testing for Pitch, Yaw and Roll
a) Pitch Error (pitch = 0 to 45 degrees)
b) Yaw Error (yaw = 0 to 45 degrees)
c) Roll Error (roll = 0 to 45 degrees)

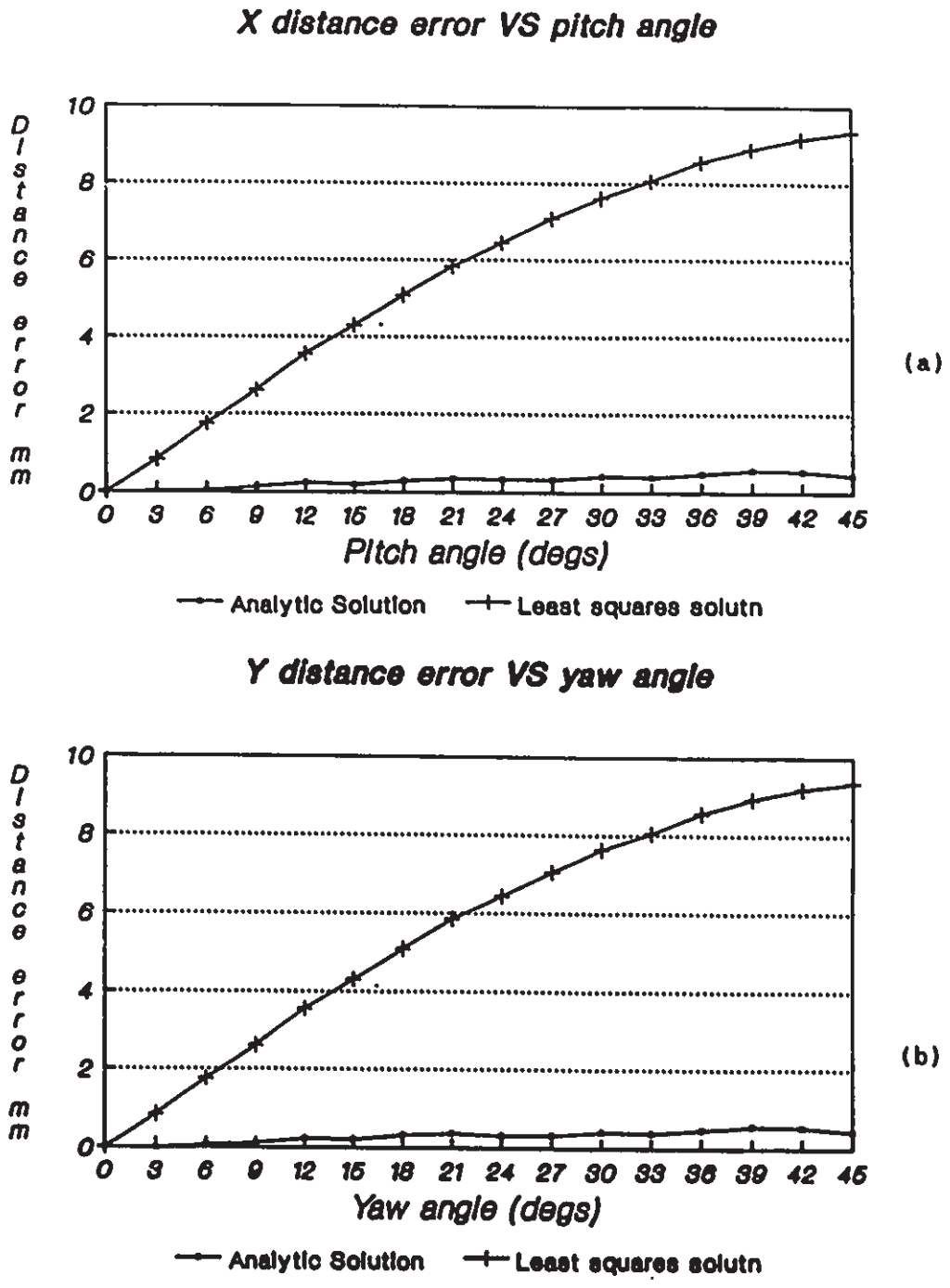


Figure 4.4. Effects of Pitch and Yaw on X and Y Position Calculation
a) X position error vs pitch angle
b) Y position error vs yaw angle

Further testing was performed to determine the effects of pitch and yaw on the measurement of X,Y and Z position. Figure 4.4 (a) and (b) show the errors in calculation of X position VS pitch angle and Y position VS yaw angle. It is evident that pitch rotations directly affect the accuracy of the X position calculation and yaw rotations directly affect the accuracy of Y position calculation, but the errors reported for the geometric solution are consistently smaller. Figure 4.5 (a) and (b) show the resulting Z position error when the target is pitched or yawed. Both tests show small Z position errors for both methods up to approximately 9°, where the least squares estimate error begins to rise significantly more quickly than the geometric solution.

Finally, the target was subjected to pitch, yaw and roll simultaneously in order to compare the resulting position and rotation errors. Each rotation step in Figure 4.6 (a) and (b) represents a 2° increase in each of the pitch, yaw and roll angles. The distance errors in Figure 4.6 (a) show that the errors in X and Y estimation begin to rise almost immediately for the least squares estimate while the Z position error begins to rise when pitch, yaw and roll are each approximately 12°. The results obtained by the geometric solution remain below 1mm for the entire test. Figure 4.6 (b) shows the pitch,yaw and roll error begin to rise when target pitch, yaw and roll are approximately 8° for the least squares solution. The geometric solution errors remain small throughout the test.

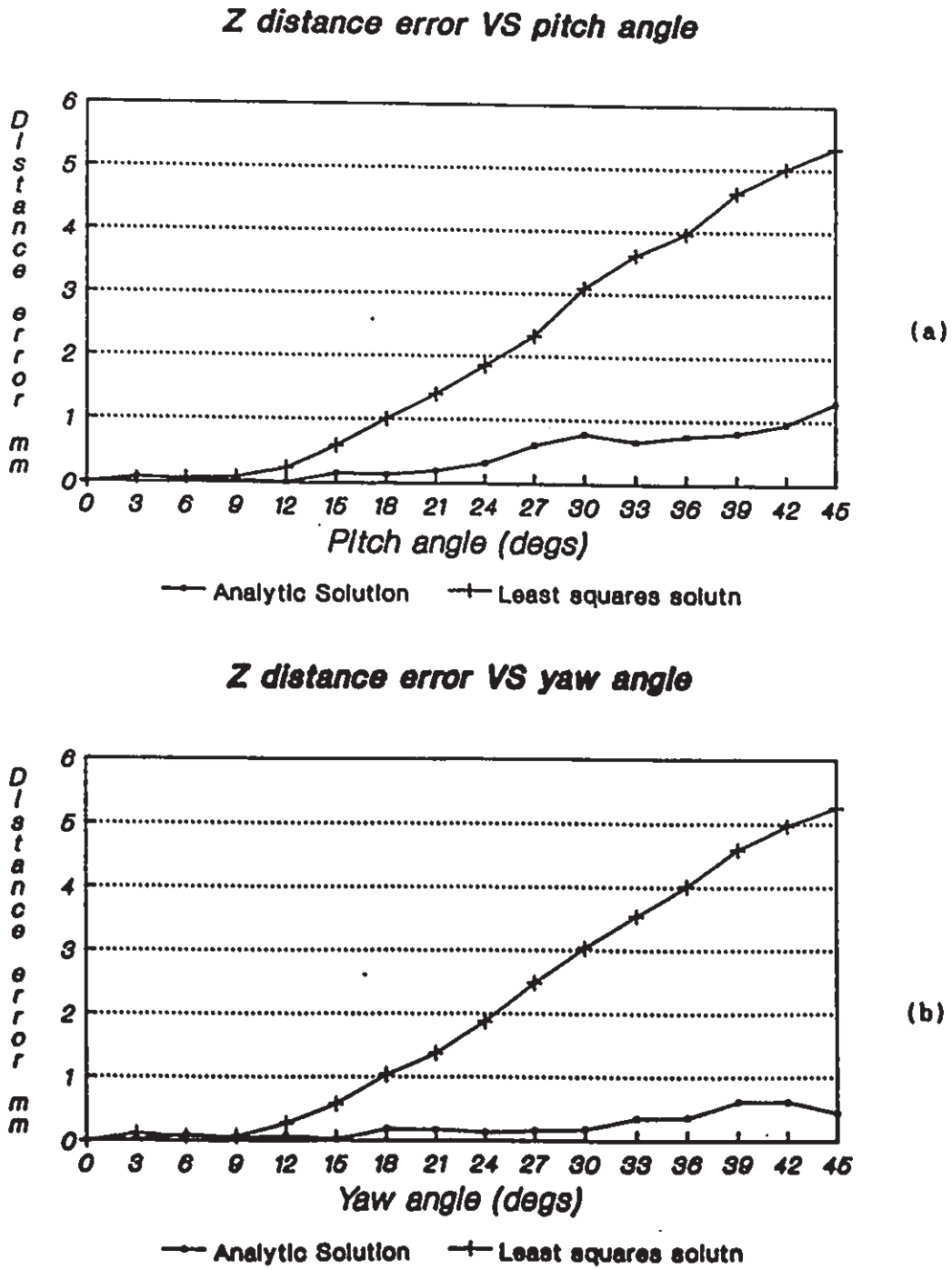


Figure 4.5. Z Position Calculation Error vs Pitch and Yaw Rotation
a) Z position error vs pitch
b) Z position error vs yaw

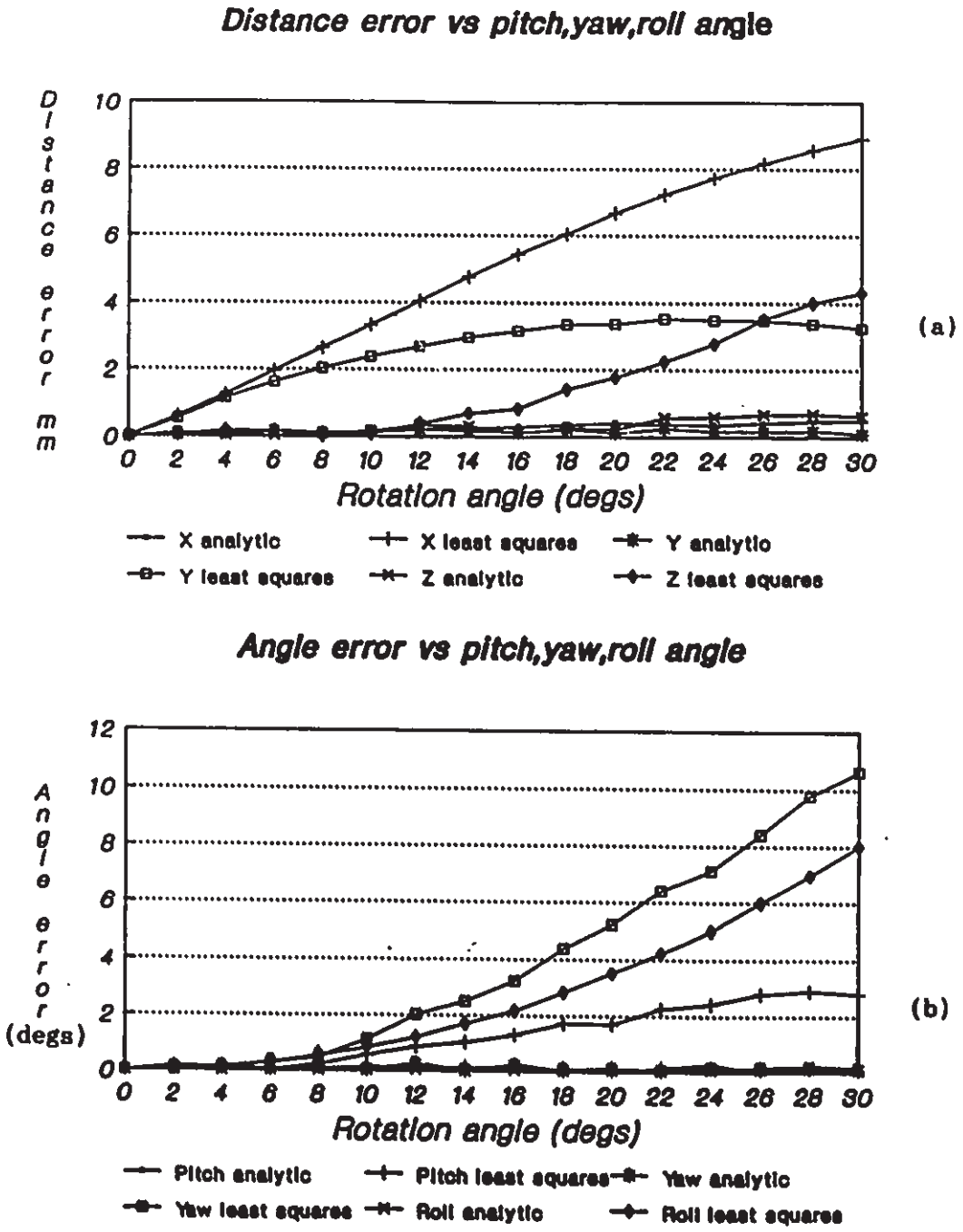


Figure 4.6. Effects of Simultaneous Pitch, Yaw and Roll on Position and Rotation Calculations.
a) Position errors introduced by pitch, yaw and roll.
b) Rotation errors introduced by pitch, yaw and roll.

4.3 Experimental Results for Random Planar Patterns

The generalized geometric solution was tested using a number of planar four point patterns illustrated in Figure 4.7. The patterns were chosen to demonstrate the effects of non-square patterns on the errors in position and orientation calculation. In each case 20mm target dots were used with an effective focal length of 100mm. For each pattern the results of multiple rotation and multiple translation tests are reported.

The results for the multiple rotation tests are listed in Table 4.2. For each pattern 30 perspective projections were generated for rotation steps of 2° in each of pitch, yaw and roll and the rotations calculated using the generalized solution were compared to the actual rotations (error averages and standard deviations were calculated as in section 4.2). For each convex shape the errors tend to decrease as the shape becomes more like a square. For the concave shape the error is more severe than most of the convex shapes. The largest errors are reported for diamond 1 pitch estimation. This result is expected since pitch is rotation about the y axis and the two target dots on the x axis are close together. The average rotation errors for most cases are below 0.2 degrees for pitch and yaw and below 0.1 degrees for roll. These results compare favourably with those obtained for the square target.

Translation tests were carried out in much the same way. For each pattern 30 perspective projections were generated for translations of 2mm in each of the X,Y and Z directions. The average errors and standard deviations are reported in Table 4.3 for each pattern. Although less

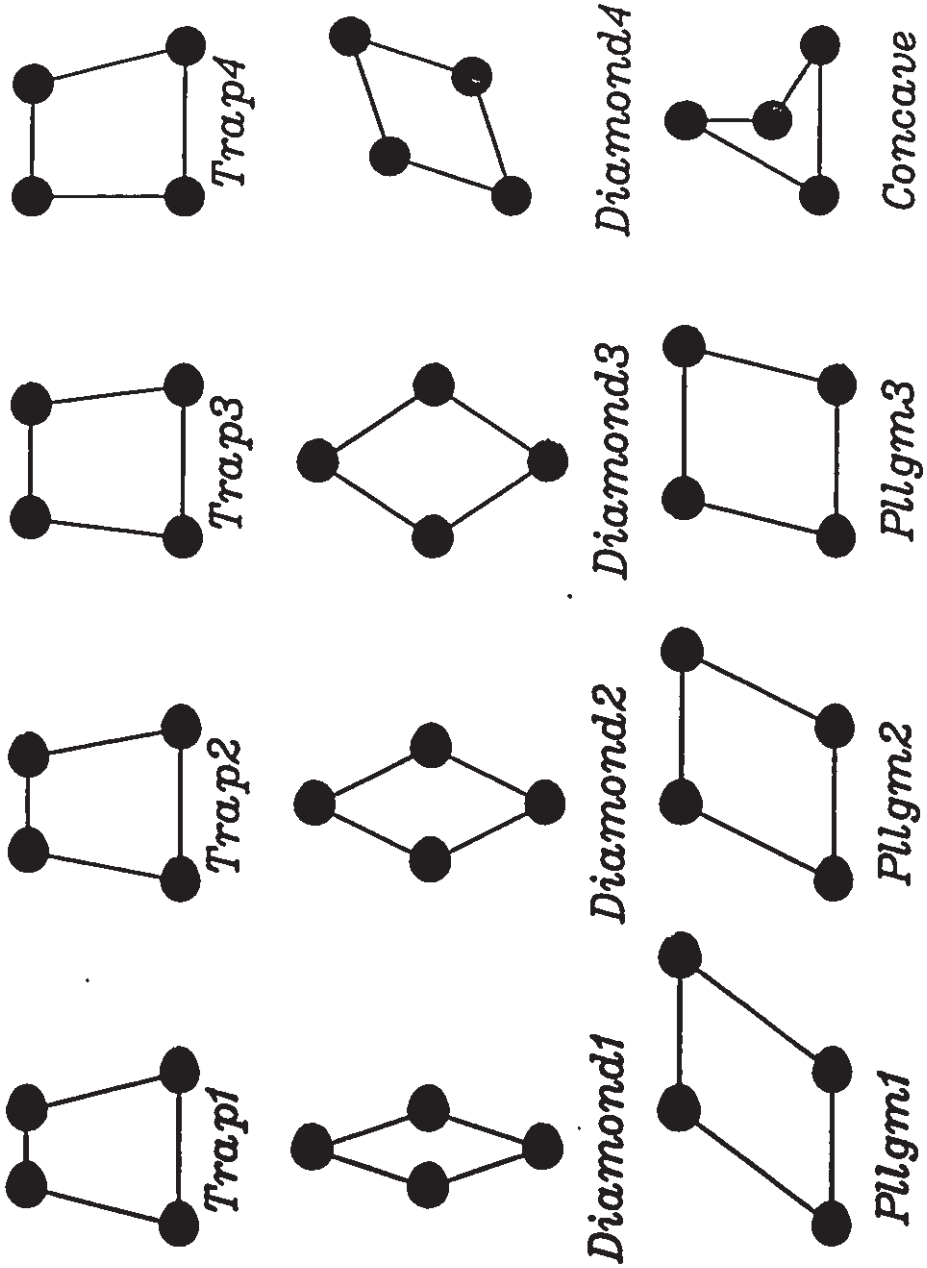


Figure 4.7. Arbitrary Planar Patterns used to Test General Solution

Rotation Tests						
Pattern	Pitch Error		Yaw Error		Roll Error	
	avg.	std.dev.	avg.	std.dev.	avg.	std.dev.
Trap 1	0.113	0.095	0.139	0.071	0.065	0.071
Trap 2	0.099	0.063	0.091	0.084	0.069	0.054
Trap 3	0.075	0.063	0.092	0.077	0.060	0.063
Trap 4	0.085	0.071	0.093	0.084	0.052	0.045
Diamond 1	0.540	0.458	0.07	0.077	0.094	0.063
Diamond 2	0.280	0.161	0.060	0.045	0.075	0.055
Diamond 3	0.150	0.114	0.082	0.054	0.075	0.063
Diamond 4	0.134	0.100	0.135	0.110	0.081	0.063
Pllgm 1	0.071	0.055	0.102	0.084	0.063	0.055
Pllgm 2	0.065	0.063	0.098	0.089	0.055	0.055
Pllgm 3	0.068	0.055	0.072	0.055	0.065	0.045
Concave	0.143	0.127	0.161	0.134	0.083	0.077

Table 4.2. Results Obtained from Rotation Tests
(note: errors measured in degrees)

Translation Tests						
Pattern	X Error		Y Error		Z Error	
	avg.	std.dev.	avg.	std.dev.	avg.	std.dev.
Trap 1	0.025	0.023	0.031	0.022	0.082	0.063
Trap 2	0.024	0.022	0.025	0.020	0.059	0.063
Trap 3	0.021	0.015	0.026	0.020	0.059	0.063
Trap 4	0.025	0.024	0.024	0.024	0.066	0.055
Diamond 1	0.040	0.045	0.028	0.023	0.075	0.077
Diamond 2	0.040	0.045	0.028	0.029	0.074	0.063
Diamond 3	0.039	0.045	0.032	0.028	0.070	0.063
Diamond 4	0.023	0.018	0.023	0.017	0.061	0.063
Pllgm 1	0.036	0.028	0.039	0.029	0.047	0.045
Pllgm 2	0.025	0.023	0.030	0.024	0.060	0.071
Pllgm 3	0.028	0.023	0.027	0.031	0.064	0.077
Concave	0.054	0.045	0.060	0.055	0.186	0.164

Table 4.3. Results Obtained from Translation Tests.
(note: errors measured in mm)

pronounced, the trend toward better results as each convex pattern approaches a square is evident. In this case the errors obtained for the concave pattern are larger than those for convex patterns. There is less variation in the range of errors for position estimation than for rotation estimation. The average x and y errors are below 0.06mm in all cases and the average z error is below 0.1mm for all cases except the concave shape. As was the case for rotation errors these results are comparable to the results obtained for the square target.

4.4 Tracking Using Corner Features

Simulations similar to those for arbitrary planar patterns were carried out using corner features of the shapes depicted in Figure 4.8. The corners chosen for the tracking are windowed on each shape. Note that the triangle of Figure 2.20 is not included amongst the shapes since it does not have the minimum 4 viable corners. As was the case for the planar patterns of section 4.3 combined pitch, yaw, roll rotation and combined X, Y, Z translation tests were carried out. Average and standard deviations of the errors were calculated as in previous sections. The results of the translation tests and the rotation tests are given in Tables 4.4 and 4.5 respectively.

As was the case for projection matching, the size of the shape affected the accuracy of the results for the first 4 shapes. The largest shape (2) resulted in pitch and yaw errors of approximately 2.5 degrees and roll errors of 1 degree. For the smallest shape (4) pitch and yaw errors were approximately 6 times higher and the roll error was twice as

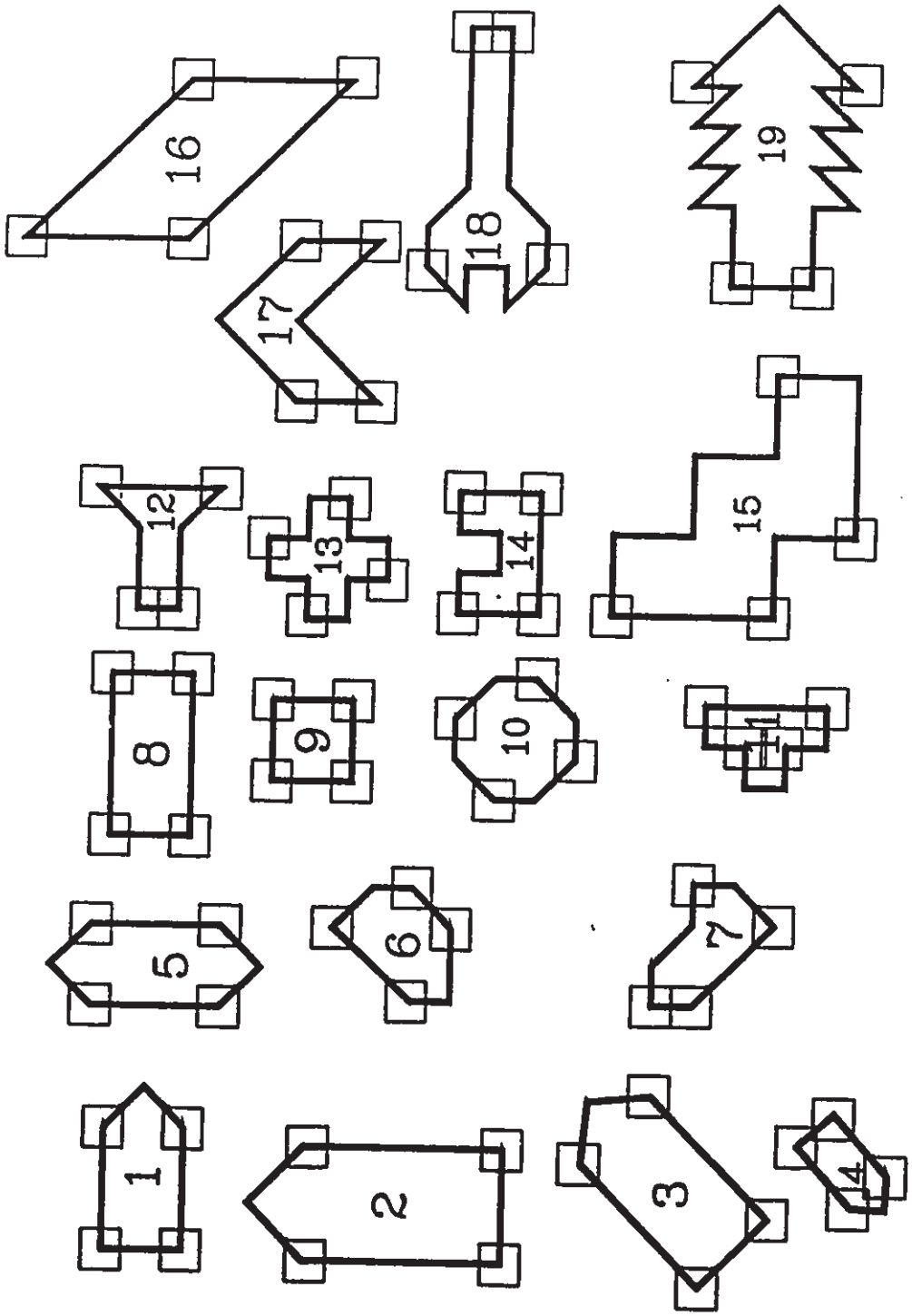


Figure 4.8. Shapes Used to Test Tracking with Corner Features.

large. The variation in error from shape 2 to shape 4 for the translation tests was about 4 times for x and y and 7 times for z. The corners chosen for the tracking also affected the results. Shapes such as 6 or 10 in which corners of large angles were used tended to lead to larger errors. This is consistent with the accuracy of corner location results shown in section 3.2.

When compared to the results obtained using target dot patterns, it is evident that in general, more accurate results were obtained using target dots. The smallest rotation errors reported were for the square (shape 9) which had pitch, yaw and roll errors of approximately 0.4 degrees. In contrast the smallest rotation errors for target dots were approximately 0.05 degrees or almost 10 times more accurate. Comparing the best of the translation tests yields the same ratio. This can be attributed to the fact that the centroids of the target dots are calculated by averaging the moments for the entire dot whereas the corners are located differently. This averaging leads to smaller projection location errors than the corner estimation technique.

Rotation Tests						
Shape No.	Pitch Error		Yaw Error		Roll Error	
	avg.	std.dev.	avg.	std.dev.	avg.	std.dev.
1	3.25	1.03	3.05	0.48	1.46	0.25
2	2.44	0.80	2.55	0.36	1.08	0.21
3	9.32	1.19	4.41	0.55	1.95	0.29
4	17.32	1.85	14.42	1.88	3.24	0.43
5	3.15	0.34	3.15	0.38	0.93	0.11
6	5.65	0.73	9.16	0.95	3.86	0.30
7	2.54	0.35	2.25	0.38	0.88	0.14
8	0.29	0.04	0.55	0.06	0.28	0.02
9	0.40	0.05	0.41	0.05	0.42	0.04
10	2.92	0.43	3.05	0.42	0.80	0.11
11	2.42	0.33	1.54	0.30	0.92	0.10
12	0.90	0.17	1.17	0.18	0.33	0.05
13	0.28	0.05	0.31	0.05	0.42	0.06
14	0.49	0.06	0.84	0.09	0.20	0.02
15	1.99	0.27	1.98	0.26	0.69	0.09
16	0.81	0.33	0.64	0.07	0.89	0.12
17	4.49	1.14	1.90	0.25	1.08	0.26
18	2.35	0.24	2.75	0.44	0.51	0.11
19	0.98	0.24	0.93	0.20	0.32	0.06

Table 4.4. Rotation Error Results Obtained for Corner Feature Tracking
(note: errors measured in degrees)

Translation Tests						
Shape No.	X Error		Y Error		Z Error	
	avg.	std.dev.	avg.	std.dev.	avg.	std.dev.
1	0.87	0.17	0.97	0.15	3.67	0.64
2	0.49	0.09	0.70	0.12	2.11	0.43
3	2.39	0.66	2.99	0.67	9.41	1.91
4	1.72	0.41	2.59	0.45	14.64	2.11
5	0.91	0.23	1.07	0.31	3.78	0.82
6	1.54	0.30	1.39	0.27	4.90	0.71
7	0.72	0.08	0.71	0.07	2.82	0.45
8	0.17	0.02	0.21	0.03	0.43	0.05
9	0.22	0.03	0.22	0.03	0.39	0.05
10	1.12	0.13	1.36	0.24	4.13	0.76
11	0.25	0.03	0.25	0.02	0.86	0.12
12	0.34	0.03	0.20	0.02	0.75	0.11
13	0.23	0.03	0.19	0.03	0.42	0.06
14	0.24	0.03	0.28	0.04	0.38	0.07
15	0.43	0.09	0.57	0.09	1.83	0.31
16	0.20	0.03	0.45	0.05	0.57	0.10
17	1.54	0.10	0.60	0.08	2.47	0.40
18	0.77	0.12	0.60	0.09	1.89	0.24
19	0.26	0.03	0.28	0.04	1.01	0.16

Table 4.5. Position Errors Obtained for Tracking Using Corner Features
(note: errors measured in mm)

4.5 Effects of Resolution

The effect of camera resolution was tested by using various pixel densities to represent similar shapes. Combined rotation and combined position simulations were carried out for each shape size similar to the tests carried out in section 4.4. Typical results of the tests are displayed for shapes 8, 11 and 12 of Figure 4.8 in Figures 4.9, 4.10 and 4.11 respectively. In each case there is a notable improvement in both the distance and angle estimates as the area of the shape is increased. The most notable improvement results in the case of shape 11 where an increase in shape area from 1700 pixels^2 to 10000 pixels^2 decreases the average Z error by a factor of 4 and the average yaw error by a factor of more than 10. This improvement may be attributed to the fact that the errors for shape 11 were initially higher than those of other shapes.

These results emphasize the need to carefully select the camera and lens combination to be used in the tracking system based on the required accuracy of the application. As was the case for the target dot implementation described in chapter 3 it may be necessary to use sub-pixel resolution in order to decrease the measurement error sufficiently.

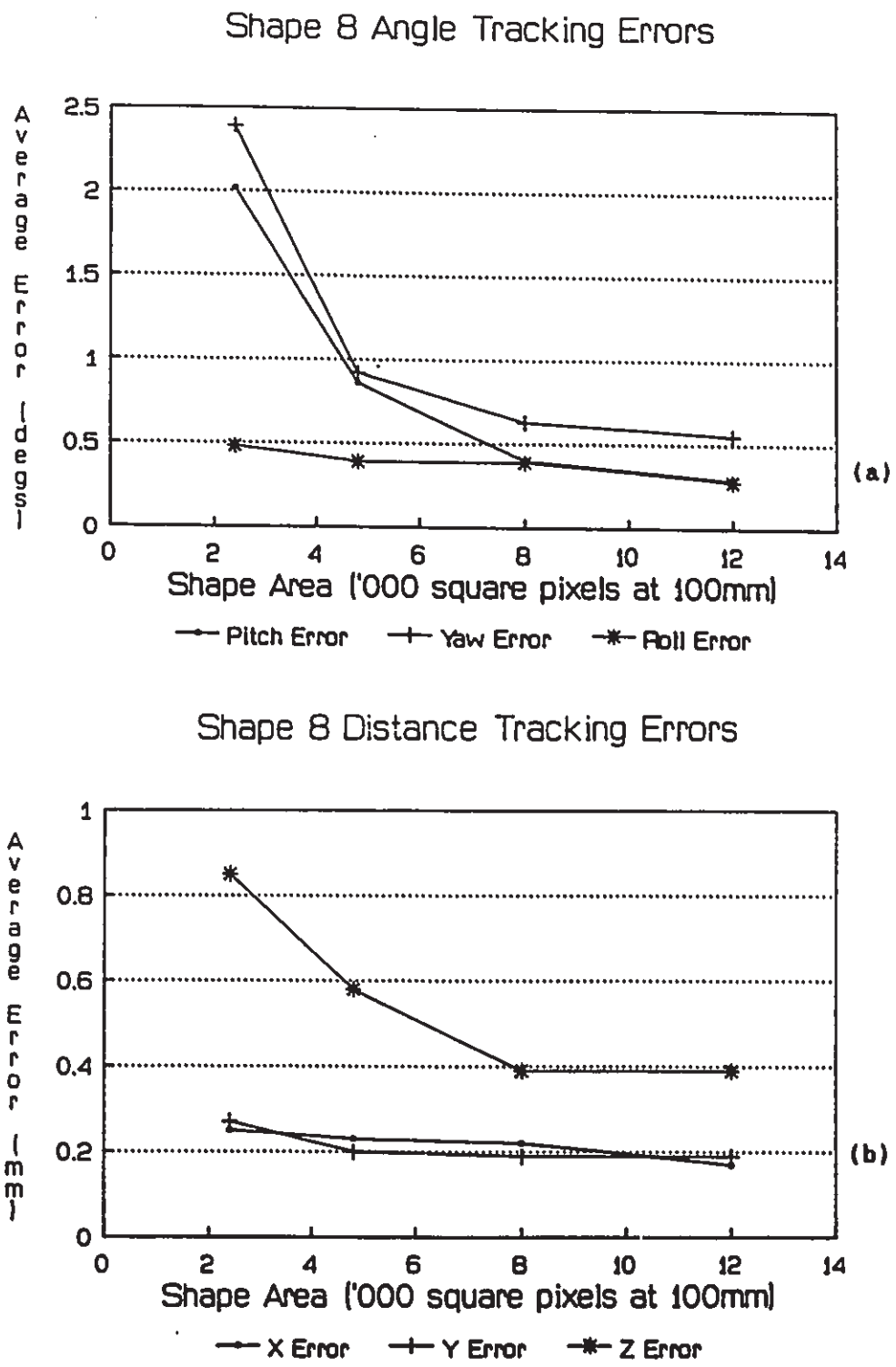


Figure 4.9. Relationship Between Resolution and Position and Rotation Errors for Shape 8.
a) Rotation error vs area.
b) Position error vs area.

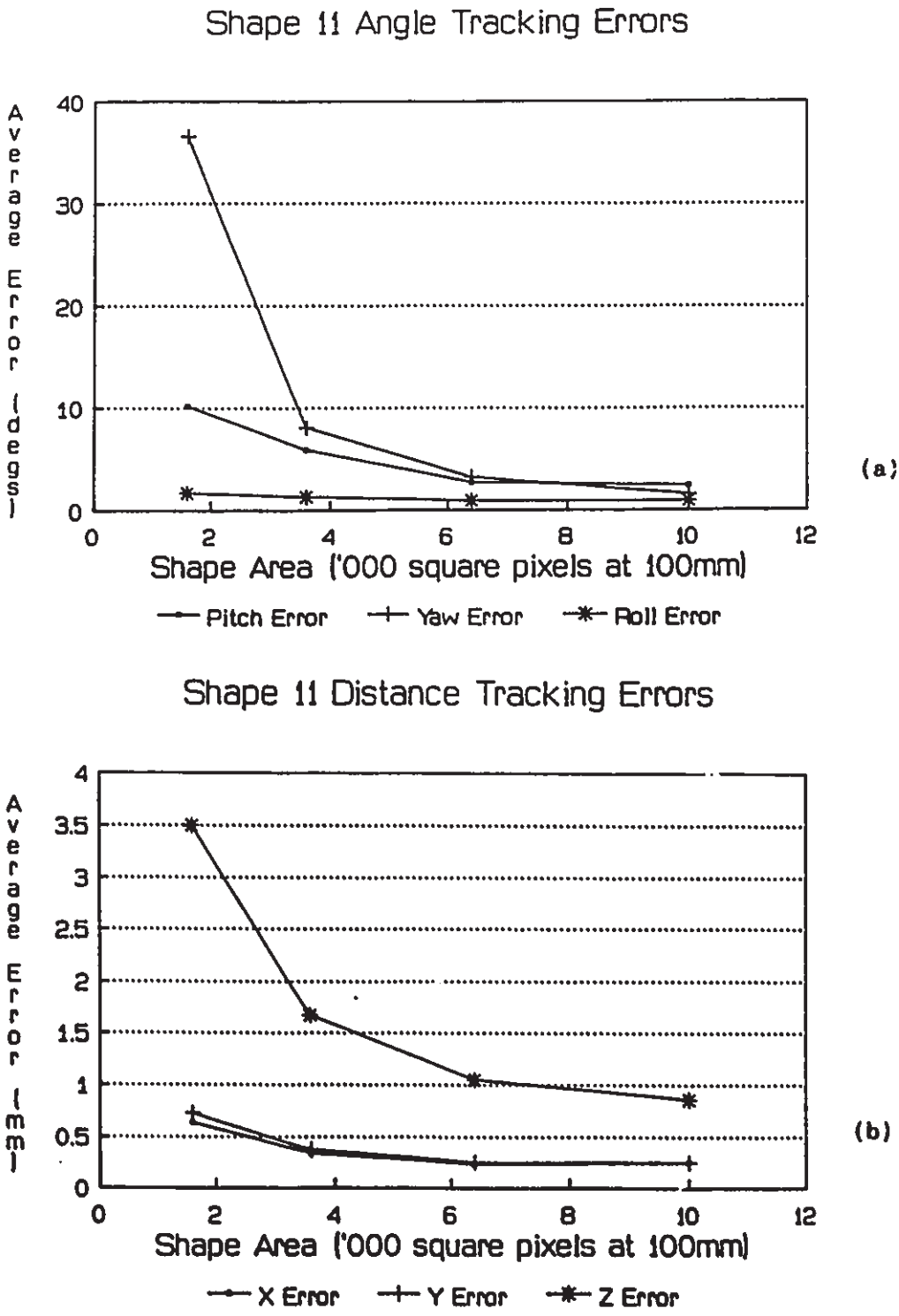
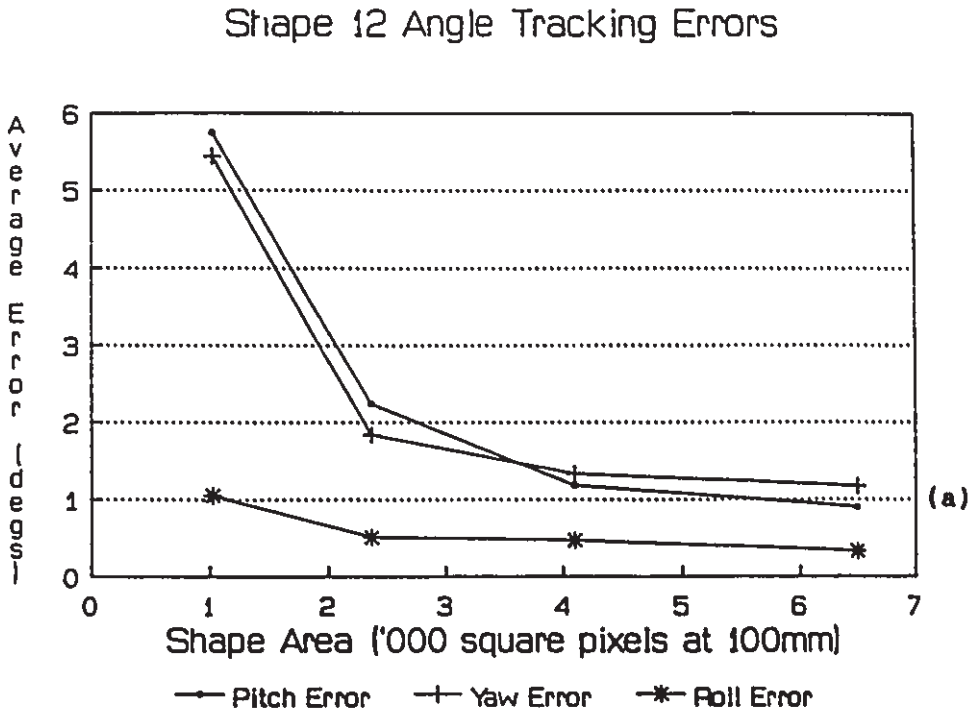


Figure 4.10. Relationship Between Resolution and Rotation and Position Errors for Shape 11.
a) Rotation error vs shape area.
b) Position error vs shape area.



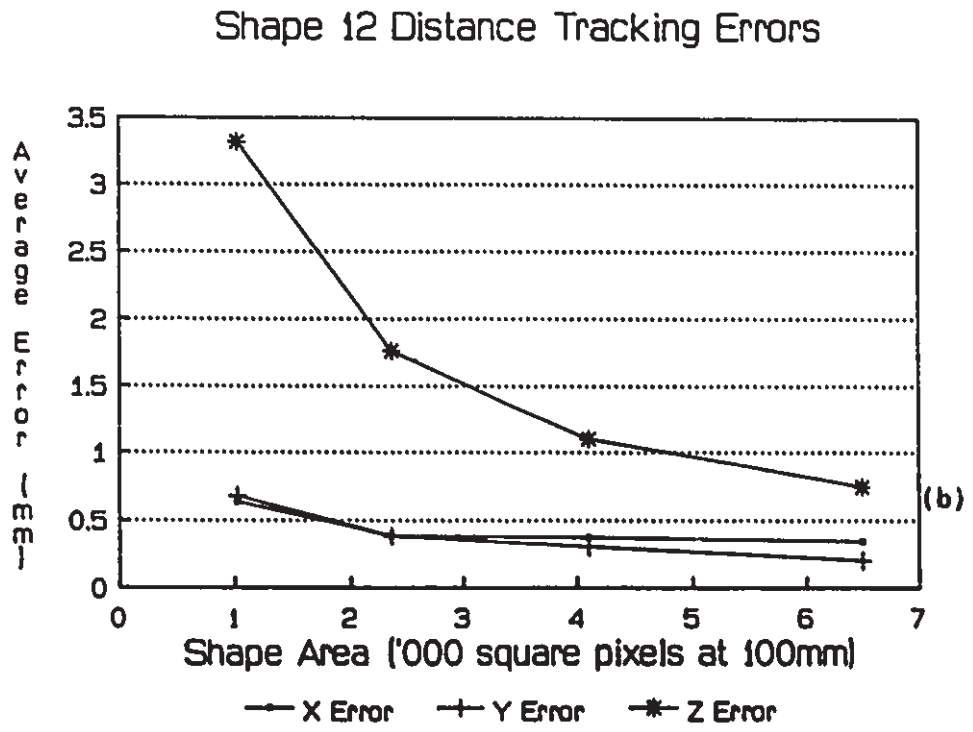
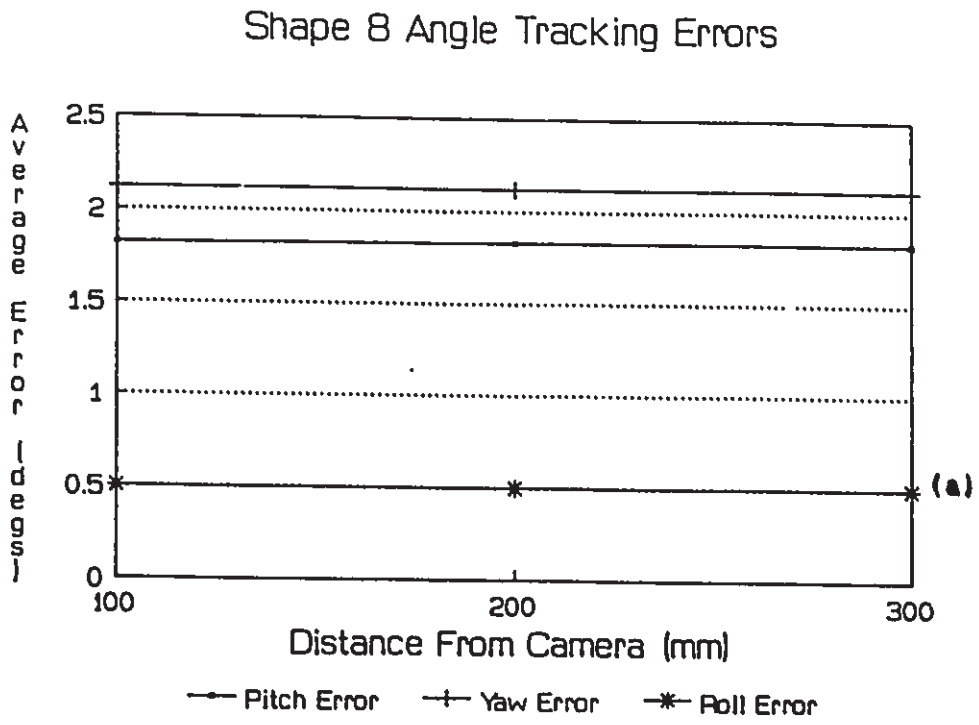


Figure 4.11. Relationship Between Resolution and Rotation and Position Errors for Shape 12.
a) Rotation error vs shape area.
b) Position error vs shape area.

4.6 Distance Effects

The effect of the distance of the shape from the camera was tested by using similar shapes of differing areas such that at the starting position the areas of the projections were the same. Combined position and combined rotation tests were again carried out. Thus for shape 12 of Figure 4.8 starting at a distance of 100mm from the camera the shape area is 1000 mm^2 and the area of the projection is 1000 $pixels^2$ whereas when starting at a distance 200mm from the camera the shape area is 4000mm^2, but the area of the projection is still 1000 $pixels^2$. The results for shapes 8, and 9 at distances 100mm, 200mm and 300mm are illustrated in Figures 4.12 and 4.13.

Clearly, increasing the distance from the camera decreases the accuracy of the system for a fixed focal length. This is particularly evident from the position calculations. This effect is expected since the ratio of camera resolution to shape area decreases as the shape distance is increased. The effects on rotation accuracy are less evident, but if the results of Figure 3.2 are taken into account then as the objects are moved further away the effects of pitch and yaw are less apparent in the resulting projection shape. The angle between two sides of the square pitched at 45 degrees approaches 90 degrees as the shape is moved away. As this 90 degree angle is approached the projection of a shape may become ambiguous and as a result a pitch of +45 degrees or -45 degrees will lead to nearly identical projections.



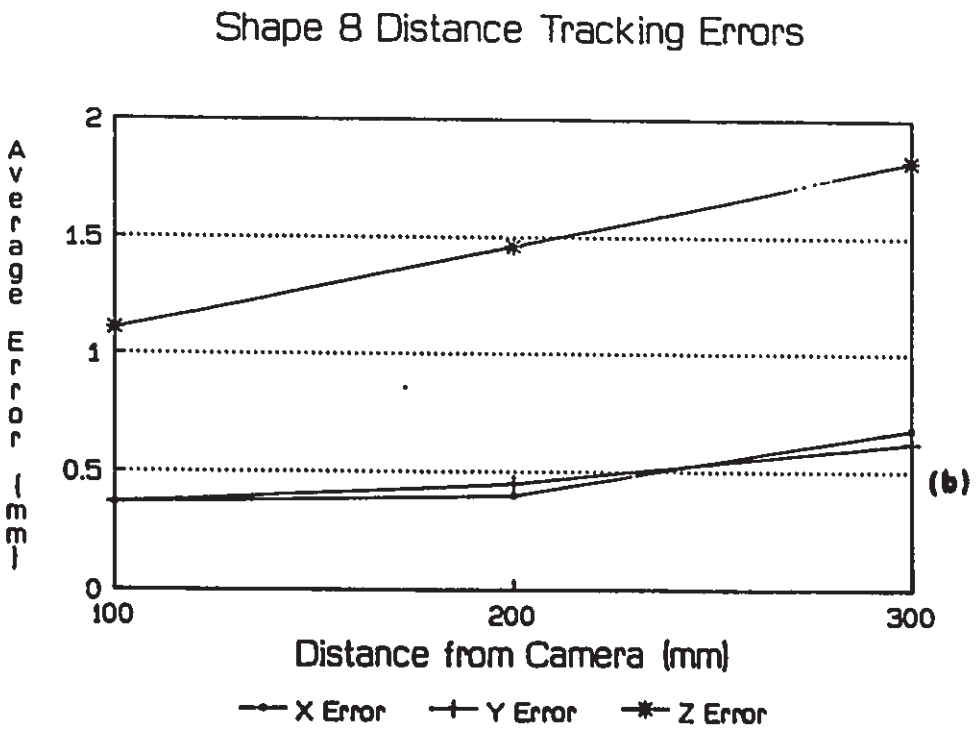
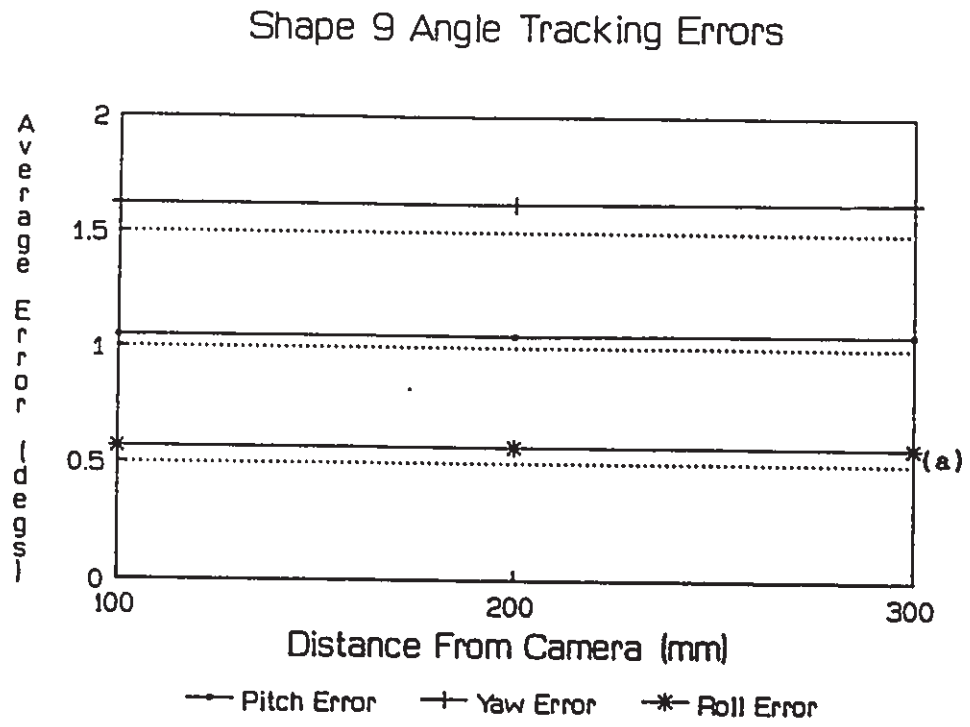


Figure 4.12. Effects of Shape Distance on Rotation and Position Errors for Shape 8.
a) Rotation error vs distance.
b) Position error vs distance.



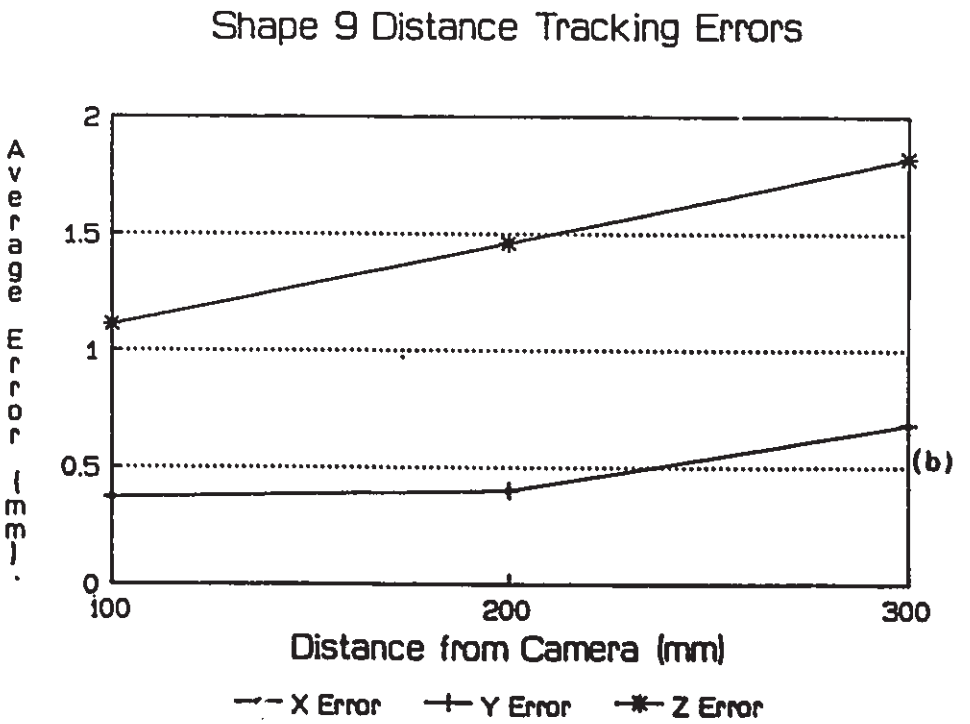


Figure 4.13. Effects of Shape Distance on Rotation and Position Errors for Shape 9.
a) Rotation error vs shape distance.
b) Position error vs shape distance.

4.7 Conclusions Resulting From Simulations

The tests carried out using the Fourier descriptors to estimate the initial projection position reveal many of the potential problems in using shape to determine position. While 100% correct matches are obtained for some shapes, most shapes are not matched with complete success. This can be attributed to the possibility of obtaining ambiguous projections depending on the shape's pitch and yaw. The determination of roll is also not accurate for many shapes. Further investigation of this technique is warranted. It should be noted that in practical applications, the position of the shape may be somewhat more constrained than is the case in these simulations.

The comparison of the least squares (NRC) photogrammetric solution to the analytic solution developed in this thesis reveals the shortcomings of the NRC solution. While the NRC solution performs comparably at near nominal position, the analytic solution provides better results as pitch and yaw angles were increased. This is not surprising since the least squares solution is optimised for the near nominal case, but it points out the usefulness of the analytic solution if shapes are approached at large angles.

The simulations carried out using random planar patterns serve to demonstrate the capability of the analytic solution to track non-square patterns. These results also demonstrate the desirability of near square patterns for which the best results are obtained.

Tracking shapes using corner features serves to demonstrate the abilities of the feature window corner detection technique as well as to

contrast th· use of corners with the use of target dots. In general, the errors obtained using the corner features are an order of magnitude larger than those obtained using target dots. This may be attributed to the calculation of target centroids which is an averaging process and is thus less prone to the effects of digitization error.

The use of various pixel densities to represent similar shapes demonstrates the positive effects of using higher resolution to track shapes.

Finally, the effects of object distance from the camera are shown to affect the results marginally in the simulations conducted.

CHAPTER 5

DISCUSSION

5.1 Conclusions

Vision systems promise to play an increasingly important role in automated manufacturing. Systems capable of determining the three dimensional position and orientation find applications in robot guidance and AGV guidance. A single camera planar tracking system has been presented in this thesis. The system consists of shape projection identification, corner location and fast three dimensional position determination.

The projection identification system uses Fourier descriptors of the shape contour to determine the initial orientation of the shape. Specialized hardware was designed and implemented and an efficient and compact algorithm (less than 2K bytes) was tested in order to extract contours quickly. Feature windows are then placed on the estimated corner locations and the corner positions are more accurately determined.

A geometric approach to the determination of the 3 dimensional position and orientation of a 4 point planar pattern is presented. The results obtained using this approach, when compared to the results obtained using a least squares solution for a square pattern, are seen to be comparable at near nominal position and to generate significantly lower

errors at larger orientation angles. In addition, this geometric approach was extended to arbitrary four point patterns.

The general solution for arbitrary 4 point planar patterns was then tested using a number of patterns. While better results were obtained for nearly square patterns, the accuracy of the results for non-square patterns was of the same order.

The 30Hz four target dot solution developed by the National Research Council demonstrated the capabilities of a fast orientation and translation determination system. The new geometric solution allows accurate results to be obtained over a wider range of orientations. Its ability to employ random planar patterns allows the system to be employed without the need to affix targets to the object to be tracked. Instead, this technique would be suited to the tracking of planar shapes having 4 or more discernable features such as holes or corners. The translation and rotation tests using the various patterns reveal the tendency toward better results when the patterns are approximately square. The worst orientation results were obtained for diamond 1 where the two horizontally opposite pattern points are closest together. This is due to the loss of perspective information for closely neighbouring points. This indicates that smaller errors will be obtained if the features used in tracking are carefully chosen.

The closed form of the geometric solution allows orientation to be determined in a finite time. This result will be applied in future work to the tracking of planar shapes using shape derived features. The development of fast processors such as the TMS320 family of Texas

Instruments makes a 30Hz or faster system feasible.

The processing time required is an important consideration when implementing a tracking system for practical application. Accordingly,

Computation	Processor	Type of Math	Time Required
32 Point Complex FFT	TMS320C25 [5.1]	fixed point	140.7 μs
Projection Matching (400 learned Projections)	TMS320C25	fixed point	45 ms
Calculation of General 4 point Solution	Intel 80287 (5 M.Hz) [5.2]	64 bit Floating Point	3849 μs

Table 5.1. Execution Times for Various Tracking Tasks.

some of the processing times necessary to implement this system are included in Table 5.1. The TMS320C25 was chosen to provide benchmark figures for the FFT calculations and the projection matching because of its availability, and its fast instruction execution time. It is felt that fixed point arithmetic is suitable for these tasks. Floating point arithmetic is desirable to compute the photogrammetric solution, and thus the execution time for the Intel 80287 math coprocessor is given for this task. Clearly the most time consuming task is the shape projection matching. A practical solution would be to split the matching task amongst 2 or more processors, since the distance computations for each projection

are independent.

To implement the corner location algorithm it is necessary to have hardware capable of collecting the necessary data. Hardware that is suited to this end for target dots was developed in conjunction with Diffracto Canada Ltd, Windsor, Ontario. A single multibus board based on the TMS320C25 processor that was developed is capable of collecting the data for the target dot solution at 60Hz. In addition the processing of the RS170 video is performed on this board. Furthermore, the TMS320C25 processor software of this board could be modified to provide the data for corner location.

In summary, this thesis reports contributions in the following areas:

1) Use of the Fourier descriptors to determine the initial position of a planar shape along with simulations to determine the success rate of this method.
2) The development of a feature window analysis technique to find corners of a shape along with simulations to determine the accuracy of corner location.
3) Development of hardware to perform fast connectivity analysis of an image.
4) Development of hardware to perform 60Hz target dot detection for a real-time tracking system.
5) Development of an analytic photogrammetric solution for four point planar patterns and simulations to determine 1) the effects of various patterns on tracking accuracy, 2) the accuracy of the

analytic solution versus the least squares single iteration approximation and 3) the accuracy of tracking using the corner detection technique.

5.2 Future Work

There are two useful avenues of investigation for future work. First, the projection matching algorithm could be improved. Possibly, a tree structure could be developed to find the nearest match, thereby avoiding an exhaustive search of the projection data. The use of the newest generation of signal processor, the TMS320C30, should also be considered. This processor is not only capable of executing instructions (i.e. TMS320C25 programs) 40% faster than the TMS320C25, but is also capable of performing floating point math operations in single instruction cycles. Other notable improvements of the TMS320C30 are increased on board memory, an on board program cache and an expanded memory addressing capability.

A second extension of this work would be to modify the general solution for four point planar patterns in order to determine the three dimensional position of non-planar patterns. This would allow any four discernable object features to be employed to determine its position. A solution for non-planar four point patterns can be found if two observations are made. Consider Figure 5.1 in which a four point non-planar pattern is illustrated. First, two triangular planar areas sharing one side can be found (triangles P1,P2,P3 and P2,P4,P3). This is possible for any configuration of four non-planar points provided no three points

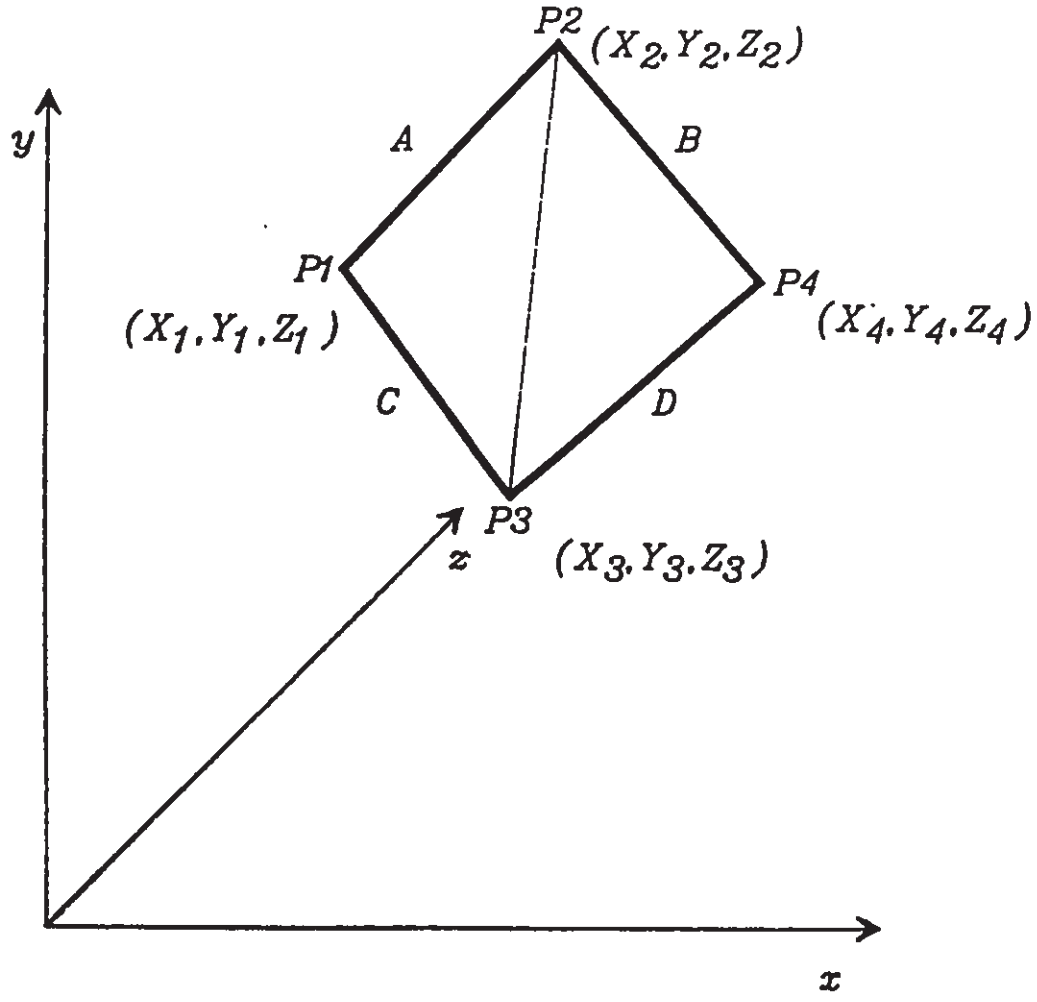


Figure 5.1. Example of a Four Corner Non-Planar Target.

are collinear. Second, it is possible to define a plane which intersects the sides A, B, C, and D as shown in Figure 5.2. Using these two observations a set of 11 linear and one non-linear equations can be formulated which can be solved to find the three dimensional position of the four non-planar points.

As was the case for the four point planar solution, eight equations can be derived from the relationship between the actual three dimensional position of the points and their image projection points.

$$X_1 = X_{1p} * Z_1 / f_e \quad [5.1]$$

$$Y_1 = Y_{1p} * Z_1 / f_e \quad [5.2]$$

$$X_2 = X_{2p} * Z_2 / f_e \quad [5.3]$$

$$Y_2 = X_{2p} * Z_2 / f_e \quad [5.4]$$

$$X_3 = X_{3p} * Z_3 / f_e \quad [5.5]$$

$$Y_3 = Y_{3p} * Z_3 / f_e \quad [5.6]$$

$$X_4 = X_{4p} * Z_4 / f_e \quad [5.7]$$

$$Y_4 = Y_{4p} * Z_4 / f_e \quad [5.8]$$

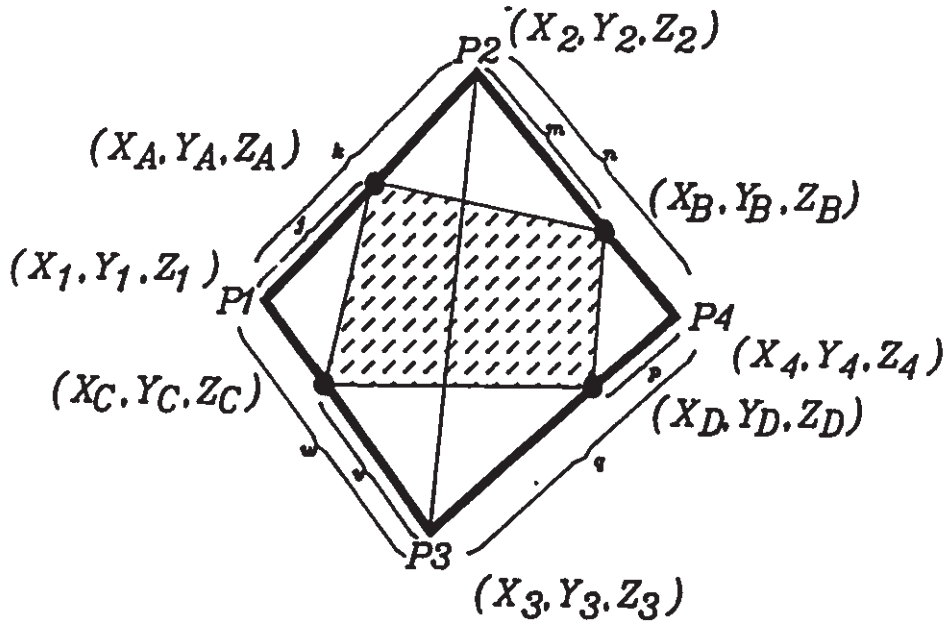
In contrast to the planar case, the next three linear equations are not readily apparent. The lines from P1 to P4 and P2 to P3 do not intersect and thus do not provide a common intersection point from which to derive the equations. The equations can be derived if the second observation is applied. Consider again a plane intersecting the four sides as defined as in Figure 5.2. The four intersection points are related to the three dimensional position of the four projection points by the ratios R, S, T, and U as shown. This results in equations 5.9 to 5.12 :

$$\begin{aligned} X_A &= R * (X_2 - X_1) + X_1 \\ Y_A &= R * (Y_2 - Y_1) + Y_1 \\ Z_A &= R * (Z_2 - Z_1) + Z_1 \end{aligned} \quad [5.9]$$

$$\begin{aligned} X_B &= S * (X_4 - X_2) + X_2 \\ Y_B &= S * (Y_4 - Y_2) + Y_3 \\ Z_B &= S * (Z_4 - Z_2) + Z_3 \end{aligned} \quad [5.10]$$

$$X_C = T * (X_1 - X_3) + X_3$$
$$Y_C = T * (Y_1 - Y_3) + Y_3 \qquad [5.11]$$
$$Z_C = T * (Z_1 - Z_3) + Z_3$$

$$X_D = U * (X_3 - X_4) + X_4$$
$$Y_D = U * (Y_3 - Y_4) + Y_4 \qquad [5.12]$$
$$Z_D = U * (Z_3 - Z_4) + Z_4$$



$$R = j / k$$
$$S = m / n$$
$$T = v / w$$
$$U = p / q$$

Figure 5.2. Ratios Derived for a Four Point Non-Planar Pattern.

Given the plane defined by the points A,B,C,D the relationship for the intersection of the diagonals AD and BC can be derived and the equations using the ratios R, S, T, and U are:

$$V * (X_A - X_D) + X_D = W * (X_B - X_C) + X_C \quad [5.13]$$

$$V * (Y_A - Y_D) + Y_D = W * (Y_B - Y_C) + Y_C \quad [5.14]$$

$$V * (Z_A - Z_D) + Z_D = W * (Z_B - Z_C) + Z_C \quad [5.15]$$

Substituting into equations 5.13 to 5.15 using equations 5.9 to 5.12 yields :

$$E * X_2 + F * X_3 = G * X_4 + H * X_1 \quad [5.16]$$

$$E * Y_2 + F * Y_3 = G * Y_4 + H * Y_1 \quad [5.17]$$

$$E * Z_2 + F * Z_3 = G * Z_4 + H * Z_1 \quad [5.18]$$

where :

$$E = V*R - (1 - S)*W$$

$$F = (1 - V)*U - (1 - W)*(1 - T)$$

$$G = W*S - (1 - V)*(1 - U)$$

$$H = (1 - W)*T - V*(1 - R)$$

Thus equations 5.1 to 5.8 and 5.16 to 5.18 form a system of 11 linear equations with 12 unknowns. The twelfth equation is again one which incorporates the absolute size of the shape similar to equation 3.45.

This solution for non-planar four point patterns makes it possible to determine analytically the position and orientation of objects in space given the projections of object feature points in a single camera view. It should be noted that recently, a similar approach was reported by Linnainmaa et al [5.4]. In this paper, 2 triangle pairs are used to determine the 3-D position of an object. The primary difference in this

approach is the uncertainty attached to the various possible positions in which a triangle can produce the same projection.

One of the primary future tasks is to develop reliable techniques to derive and match three dimensional feature points from the image to those in a data base. One method would be to recognize shapes and their approximate orientations using shape descriptors as was the case for planar shapes.

APPENDIX A

NRC PHOTOGRAMMETRY EQUATIONS

The National Research Council of Canada (NRC) developed a real-time tracking system for the space shuttle arm based on the use of a fixed target of 4 dots attached to the object to be manipulated by the arm [1.13][1.14][1.15]. The camera-target setup is illustrated in Figure 1.3. This system employs individual processors to process image data within feature windows of the digitized video image. Each processor reports the relative position within the window of the target dot. This information is then used by another processor in order to report a photogrammetric solution to the object position. This photogrammetric solution is derived using the least squares approach described in section 3.3.1 of this thesis. By using a number of simplifying assumptions about the target position relative to the camera a set of position estimating equations were derived from the least squares. The assumptions are the result of the assertion that the position of the target with respect to the camera is close to the nominal position. (i.e. the image and target planes are nearly parallel and the z distance is close to the grapple position). This leads to the substitution into the equations of rotation angle sines equivalent to the angles and cosines of 1. The first derivatives of the collinearity equations w.r.t. x and y are estimated as the slope of the ray passing through the focal

point of the camera and the target point. Using these assumptions a set of six equations used to estimate the object (x,y,z) position and pitch, yaw and roll rotations were derived. These equations are:

$$x_{co} = a_x(t)\ XPS\ /\ XP_{2413}$$

$$y_{co} = a_y(t)\ YPS\ /\ YP_{3412}$$

$$z_{co} = 4\ f_e\ (a_x(t) + a_y(t))\ /\ (\ XP_{2413} + YP_{3412}\ )$$

$$\Omega_{co} = \beta_y^2(t)\ f_e\ YP_{1423}\ /\ 4 + 2\ \Phi_{co}(t)\ \theta_{co}(t)$$

$$\theta_{co} = \beta_x^2(t)\ f_e\ XP_{2314}\ /\ 4 + 2\ \Phi_{co}(t)\ \Omega_{co}(t)$$

$$\Phi_{co} = \frac{1}{2(1 + \Phi^2_{co}(t)/3)} \left[\frac{YP_{2413}}{XP_{2413}} - \frac{XP_{3412}}{YP_{3412}} \right]$$

where:

x_{co}, y_{co}, z_{co} - x,y,z position of the target w.r.t. the camera

Ω_{co}, θ_{co}, Φ_{co} - pitch,yaw,roll of the target w.r.t. the camera

XPS(,YPS) - sum of the x coordinates(,y coordinates) of the centroids of the projections of the target dots.

$XP_{klmn} = X_k + X_l - X_m - X_n$

X_k, X_l, X_m, X_n are the x coordinates of the centroids of the projections of the target dots.

$YP_{klmn} = Y_k + Y_l - Y_m - Y_n$

Y_k, Y_l, Y_m, Y_n are the y coordinates of the centroids of the projections of the target dots.

f_e - the effective focal length of the system.

$$a_x(t) = a(1 - \Phi^2_{co}(t)/2)(1 - \Omega^2_{co}(t)/2)$$

$$a_y(t) = a(1 - \Phi^2_{co}(t)/2)(1 - \theta^2_{co}(t)/2)$$

$$\beta_x(t) = z_{co}(t) \; / \; a_x(t)$$

$$\beta_y(t) = z_{co}(t) \; / \; a_y(t)$$

These equations are applied to the projection centroids of the target dots for successive frames of video. The use of the time symbol (t) in the right hand side identifies terms for use with filtered or previous frame values. These were included because the results of NRC experiments showed that previous frame values could be used to refine the estimates for the current frame.

REFERENCES

Chapter 1

[1.1] Fahim, A.E.F., and R.M.H. Cheng, "An On-Line System for Identifying the Angular Orientation of a Class of Engineering Components", IEEE Transactions on Industrial Electronics and Control Instrumentation, Vol IECI-27, No. 3, 1980, pp. 189-196.

[1.2] Capson, D.W., Techniques for the Recognition of Silhouettes, Master's Thesis, McMaster University, 1981, pp. 45-48.

[1.3] Holland, S.W., L. Rossol, and M.R. Ward, "CONSIGHT-I: A Vision-Controlled Robot System for Transferring Parts from Belt Conveyors", Computer Vision and Sensor-Based Robots, (ed. Dodd, G.G. and L. Rossol), Plenum Press, New York, 1979, pp. 81-97.

[1.4] Williams, T.D., "Depth from Camera Motion in a Real World Scene", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-2, No. 6, November, 1980, pp. 511-515.

[1.5] Mitiche, A, S. Seida, and J.K. Aggarwal, "Using Constancy of Distance to Estimate Position and Displacement in Space", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-10, No. 4, July, 1988, pp. 594-599.

[1.6] Mutch, K., "Determining Object Translation Information Using Stereoscopic Motion", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-8, No. 6, November, 1986, pp. 750-

[1.7] Huang, T.S., "Three Dimensional Motion Analysis By Direct Matching", Journal of the Optical Society of America, Vol. 3, No. 9, September, 1986, pp. 1501-1503.

[1.8] Sethi, I.K., and R. Jain, "Finding Trajectories of Feature Points in a Monocular Image Sequence", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-9, No. 1, January, 1987, pp. 56-73.

[1.9] Augusteijn, M.F., and C.R. Dyer, "Recognition and Recovery of Three Dimensional Orientation of Planar Point Patterns", Computer Graphics and Image Processing, 36, 1986, pp. 76-99.

[1.10] Horaud, R., "New Methods for Matching 3-D Objects with Single Perspective Views", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-9, No. 3, May, 1987, pp. 401-412.

[1.11] Topa, L.C., and R.J. Schalkoff, "An Analytic Approach to the Determination of Planar Surface Orientation Using Active-Passive Image Pairs", Computer Vision Graphics and Image Processing, 35, 1986, pp. 404-418.

[1.12] Gordon, S.J., and W.P. Seering, "Real-Time Part Position Sensing", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-10, No. 3, May, 1988, pp. 373-386.

[1.13] Pinkney, H.F.L., "Theory and Development of an On-Line 30Hz Video Photogrammetry System for Real Time Three Dimensional Control", Proceedings of the ISP Symposium on Photogrammetry for Industry, Stockholm, August, 1978.

[1.14] Pinkney, H.F.L., and C.I. Perrat, "A Flexible Machine Vision Guidance System for 3-Dimensional Control Tasks", Advances in CAD/CAM and Robotics : NRC Contributions, National Research Council of Canada, May, 1987, pp. 401-411.

[1.15] Kratky, V., "Real Time Photogrammetric Support of Dynamic Three Dimensional Control", Photogrammetric Engineering and Remote Sensing, Vol. 45, No. 9, September, 1979, pp. 1231-1242.

[1.16] Thompson, M.M., Manual of Photogrammetry, American Society of Photogrammetry, Falls Church, Va., 1966, pp. 46-65.

[1.17] Wallace, T.P., and O.R. Mitchell, "Analysis of Three-Dimensional Movement Using Fourier Descriptors", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-2, No. 6, November, 1980, pp. 583-588.

[1.18] Rosenfeld, A., "SURVEY Picture Processing :1986", Computer Vision, Graphics and Image Processing, 38, 1987, pp.147-225.

Chapter 2

[2.1] Rosenfeld, A., and A.C. Kak, Digital Picture Processing, Academic Press, New York, Vol. 2, 1982, pp. 224-225.

[2.2] Cederberg, R.L.T., "Chain-Link Coding and Segmentation for Raster Scan Devices", Computer Graphics and Image Processing, Vol. 10, 1979, pp. 224-234.

[2.3] Agrawala, A.K. and A.V. Kulkarni, "A Sequential Approach to the Extraction of Shape Features", Computer Graphics and Image Processing, Vol. 6, 1977, pp. 538-557.

[2.4] Capson, D.W., "An Improved Algorithm for Sequential Extraction of Boundaries from a Raster Scan", Computer Vision Graphics and Image Processing, Vol. 28, 1984, pp. 109-125.

[2.5] Ma, J., C. Wu, and X. Lu, "A Fast Shape Descriptor", Computer Vision, Graphics and Image Processing, Vol. 34, 1986, pp.282-291.

[2.6] Frendo, M.J., D.W. Capson, and R. Kitai, "Instrumentation for High-Speed Image Connectivity", IEEE Transactions on Instrumentation and Measurement, Vol. IM-36, No. 1, March 1987, pp. 71-76.

[2.7] Weszka, J.S., "A Survey of Threshold Techniques", Computer Vision, Graphics and Image Processing, Vol. 7, 1978, pp.259-265.

[2.8] Rosenfeld, A., "Image Analysis: Problems, Progress and Prospects", Pattern Recognition, Vol. 17, No. 1, 1984, pp. 3-12.

[2.9] Reeves,A.P. et al., "Three Dimensional Analysis Using Moments and Fourier Descriptors", Seventh International Conference on Pattern Recognition, 1984, pp.447-450.

[2.10] Persoon,E. and K.S. Fu, "Shape Discrimination Using Fourier Descriptors", IEEE Transactions on Systems, Man And Cybernetics, Vol. SMC-7, No. 3, March 1977, pp.170-179.

[2.11] Zahn,C.T. and R.Z. Roskies, "Fourier Descriptors for Closed Plane Curves", IEEE Transactions on Computers, Vol. C-21, No. 3, March 1972, pp.269-281.

[2.12] Crimmins,T.R., "A Complete Set of Fourier Descriptors for Two-Dimensional Shapes", IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-12, No.6, November/December 1982, pp.848-855.

[2.13] Chellappa,R. and R. Bagdazian, "Fourier Coding of Image Boundaries", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-6, No.1, January 1984, pp.102-105.

[2.14] Searle,N.H., "Shape Analysis by Use of Walsh Functions", Machine Intelligence 5, Meltzer and Michic, New York, 1970, pp.395-410.

Chapter 3

[3.1] Davis, L.S., Z. Wu, and H. Sun, "Contour-Based Motion Estimation", Computer Vision, Graphics and Image Processing, Vol. 23, 1982, pp. 313,326.

[3.2] Moffitt, F.H. and E.M. Mikhail, Photogrammetry, Harper and Row, New York, 1980, pp. 137-143.

Chapter 5

[5.1] iAPX 286 Hardware Reference Manual, 1983, Intel Corporation, pp.6-1 - 6-7.

[5.2] TMS320C25 Users Guide, SPRU012, 1986, Texas Instruments Corporation.

[5.3] Digital Signal Processing Applications with the TMS320 Family, SPRA012A, 1986, Texas Instruments Corporation.

[5.4] Linnainmaa, S., D. Harwood, and L.S. Davis, "Pose Determination of a Three-Dimensional Object Using Triangle Pairs", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-10, No.5, September 1988, pp.634-647.