

**MICROCOMPUTER BASED VISION ALGORITHMS
FOR
OBJECT CLASSIFICATION**

**A Thesis Presented to the
Faculty of Graduate Studies
University of Manitoba**

**In Partial Fulfillment
of the Requirements for the Degree
Master of Science
in
Mechanical Engineering**

**presented by
Kristin Elaine Johnson**

(c) January 1989

National Library
of Canada

Canadian Theses Service

Bibliothèque nationale du
Canada

Service des thèses
canadiennes

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-315-48104-8

**MICROCOMPUTER BASED VISION ALGORITHMS
FOR OBJECT CLASSIFICATION**

BY

KRISTIN ELAINE JOHNSON

A thesis submitted to the Faculty of Graduate Studies of
the University of Manitoba in partial fulfillment of the requirements
of the degree of

MASTER OF SCIENCE

© **1989**

Permission has been granted to the LIBRARY OF THE UNIVERSITY OF MANITOBA to lend or sell copies of this thesis, to the NATIONAL LIBRARY OF CANADA to microfilm this thesis and to lend or sell copies of the film, and UNIVERSITY MICROFILMS to publish an abstract of this thesis.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.

ACKNOWLEDGEMENT

I would like to thank my advisor, Professor S. Balakrishnan, for his guidance and support towards the completion of this degree.

I am indebted to my father, Professor Richard A. Johnson, whose advice and infinite patience and assistance over the years have given me an invaluable education.

I also wish to express my sincere gratitude and thanks to my parents for their financial and moral support throughout my years of study at the University of Manitoba.

ABSTRACT

An efficient and affordable vision system was developed for object recognition. The system consisted of a charge coupled device (CCD) camera, an IBM AT compatible microcomputer, and a Frame Grabber board. The objects to be identified consisted of a square, a rectangle, a circle, an equilateral triangle and an isosceles triangle.

The results obtained from this system yielded consistent identification of all the objects used, and rejected those objects that were not within the scope of the project. A number of software based variables were developed. The optimum values for these variables was also obtained for the application. In addition to identifying the objects correctly, the software algorithms provided information regarding the geometrical attributes of the objects and their orientation relative to the screen coordinate system.

The software algorithms were written using the Microsoft Fortran77 language and compilers, and the execution time required to obtain results was estimated to be less than eight seconds. The methods and algorithms used in the software programs were found to be dependent on the types of objects to be identified. Limitations of the software include the types of objects it can identify, and extrapolation methods to find the corner coordinates. In addition, some possible extensions to the program are discussed and methods by which the execution speed may be enhanced are outlined.

TABLE OF CONTENTS

<u>CHAPTER</u>	<u>PAGE</u>
ACKNOWLEDGEMENT	i
ABSTRACT	ii
CHAPTER I: INTRODUCTION	1
CHAPTER II: IMAGE THEORY AND PROBLEM DESCRIPTION	3
2.1 The Digital Image Processing System	3
2.2 Image Processing Techniques	4
2.3 The Object Recognition Problem	6
2.4 Hardware and Program Structure	7
2.5 Thresholding	8
2.6 Edge Extraction	9
2.7 Feature Extraction and Object Recognition ..	10
CHAPTER III: VISION APPLICATION PROGRAM	11
3.1 General Structure of the program	11
3.2 Obtaining the Image and Extracting the Edge	12
3.3 Location and Accessing of Image	14
3.4 Edge Extraction	18
3.5 Feature Extraction and Object Recognition ..	26
3.5.1 Geometric Center	27
3.5.2 Detection of Maximum Radii	31
3.5.3 Identifying Transition Points	38
3.5.4 QUADRILATERAL Test Section	43
3.5.5 TRIANGLE Test Section	44
3.5.6 CIRCLE Test Section	46
3.6 Summary	46
CHAPTER IV: EXPERIMENTAL RESULTS	48
4.1 Optimum Value of NUMAVE	48
4.2 Optimum Value of STLEN	54
4.3 Optimum Value of RADITOL	56
4.4 Optimum Value of CIRCL	62
4.5 Optimum Value of SIDETOL	64
4.6 Summary	65
CHAPTER V: CONCLUSIONS AND RECOMMENDATIONS	68
5.1 Limitations of the Software Routines	68
5.1.1 Other Limitations of the System	71
5.2 Recommendations for Future Enhancements	72
5.2.1 Enhancing Execution Speed and Accuracy	75
5.3 Application of the Developed Algorithms	79
5.4 Conclusions	80
REFERENCES	81
APPENDICES	
A: RESULTS DATA	82
B: TSTLOAD LISTING	98
C: TSTIMAGE LISTING	111

LIST OF FIGURES

<u>FIGURE</u>	<u>PAGE</u>
FIGURE 1: Illustration of Window Updating Process ..	16
FIGURE 2: Locating the Initial Point	19
FIGURE 3: Eight Surrounding Pixels	21
FIGURE 4: Eight Possible Directions to Start Search for Next Pixel	21
FIGURE 5: Flow Chart of Edge Extraction Process ...	24
FIGURE 6: Sample Edge Following Method	25
FIGURE 7: Axes for Green's Theorem and Application	30
FIGURE 8: Setting Up the Array DSTNC	33
FIGURE 9: Setting Up the Array TRANS	35
FIGURE 10: Ideal TRANS Values for Typical Objects ..	35
FIGURE 11: Modified TRANS Arrays	37
FIGURE 12: Illustration of Trend Search	40
FIGURE 13: Intersection of Two Lines	40
FIGURE 14: Points on Same Line	40
FIGURE 15: Orientation Angle for a QUADRILATERAL ...	45
FIGURE 16: Orientation for a TRANGLE	45
FIGURE 17: Corner Location With Increasing NUMAVE ..	51
FIGURE 18: Abnormal Objects	70
FIGURE 19: Analysis of a NOise "Spike"	74
FIGURE 20: Transition Point Calculation	78

LIST OF TABLES

<u>TABLE</u>	<u>PAGE</u>
TABLE 1: Initial Directions and Next Directions ...	22
TABLE 2: Sample Edge Following Steps	25
TABLE 3: Calculation of DSTNC Array	33
TABLE 4: Effects of Increasing NUMAVE on a CIRCLE	50
TABLE 5: Corner Location Data for Increasing NUMAVE	51
TABLE 6: Maximum Percentage Difference of Sides / Radius	53
TABLE 7: Effects of Increasing RADITOL on a Circle	58
TABLE 8: Radii Length Information - Circle Parts	59
TABLE 9: Radii Length Information - Square Corner	60
TABLE 10: Radii Length Information - Square Side Midpoint	61
TABLE 11: Typical Total's of Zero's in Objects	63
TABLE 12: Typical Percentage of Zero's Over Ranges of Distances from Camera	63
TABLE 13: Differences Between Equal Sides	66

CHAPTER I: INTRODUCTION

The gradual introduction of robots in the area of manufacturing provided tireless, efficient, and consistent performance of many tasks. However, there were limits to the tasks they could perform. For reliable performance of robotized work cells, the entire work area had to be accurately set. The parts had to be inspected for any abnormalities before entering the automated manufacturing cell. The next step toward full automation became apparent: the robot needed to be equipped with the eyes of the supervising operator that oversaw its work. Thus the possibilities of robotic vision were introduced to the industrial world.

The interaction of computer vision and robotics provided the manufacturing cell with the ability to adjust to situations, such as aligning pieces to be welded or reorienting a part for machining. Currently vision is heavily used in conjunction with robots in the automotive and shipbuilding industries for welding, painting, sorting, and accurate part orientation, and also in the electronics industry for the manufacture of integrated circuit chips [1,2]. In addition, it is used to evaluate conditions in hazardous environments such as the ocean floor which is attainable by unmanned submersibles [3]. More recently, the agriculture field has been using vision to sort grain

automatically according to variables such as size, shape, color, and texture.

The field of vision has also created a variety of new possibilities in the realm of flexible manufacturing. It permits a less constrained work environment for the functioning of a robot. A good understanding of the basic concepts behind vision is hence imperative. The scope of this thesis was then:

- 1) to develop microcomputer based algorithms for object recognition and classification of simple objects;
- 2) to extract geometrical attributes which will define the location and orientation of the object; and
- 3) to accomplish these objectives in an efficient manner to create an affordable, microcomputer based vision system.

A thorough problem description and associated image processing theory is presented in Chapter II. Chapter III details the algorithms developed to achieve object recognition, and the results obtained from these routines are presented in Chapter IV. The limitations of the developed algorithms, future modifications and conclusions are discussed in Chapter V.

CHAPTER II: IMAGE THEORY AND PROBLEM DESCRIPTION

This chapter discusses the basic elements of an image processing system. It briefly describes some of the image processing techniques widely used today with particular emphasis on methods used in this project. A more detailed presentation of these topics is dealt with in references [4,5,6,7].

2.1 The Digital Image Processing System

There are three basic elements of a digital processing system: the digitizer, the processor, and the display device.

The digitizer receives a continuous analog signal from the image sensor in the camera, which represents the spatial brightness, or light intensities, in the scene. The digitizer samples and quantizes the signal into a discrete representation of the image, usually in a numerical form which is proportional to the light intensities detected by the sensor. These numbers then serve as input to a digital computer where processing may be carried out quickly. There are many types of digitizers, the most common of which are microdensitometers or flying spot scanners, which use photographs or other negatives as input. The output of these devices is then fed to image dissectors or television cameras, which can use almost any medium as input and then

display the digitized image on a monitor.

Image processors are devices that provide both the capacity for storing the digitized image and the capability of manipulating the image for specific applications. They range from specialized microprocessors to full computer systems that are capable of performing a large number of simultaneous calculations on large arrays for real time processing. The efficiency of the system is dependent on the complexity of the problem, the size of the input images, and the amount of image processing required.

The display device is a medium by which the converted image arrays may be viewed by the human eye. These include cathode ray tubes (CRT's), television monitors, and printers. The reconstruction of such a digitized, processed image can be used to compare with the original image to monitor accuracy qualitatively.

2.2 Image Processing Techniques

Image processing methods fall into four major categories: image digitization, image enhancement, image encoding, and image segmentation and representation.

Image digitization converts a continuous analog image signal into discrete, spatial and grey level components. Image enhancement improves the quality of the image to the required level necessary for the application. The techniques used are entirely dependent upon the desired

application and, thus, the field is a large one. However, some of the fundamental techniques developed should be mentioned, namely: histogram modification and equalization, image smoothing, and image sharpening.

Histogram techniques modify the frequency range of the grey levels of an image to create a more usable image for certain applications. Image smoothing methods reduce the noise in an image, and include neighborhood averaging of the grey levels to produce a more balanced image. Low pass filters which attenuate the high frequency or sharpness (edges, noise) attributes of the image, are also employed to create a more blurred look. Image sharpening, on the other hand, utilizes high pass filters which attenuate the low frequency components of an image, thus accenting the edges present in the scene.

Image encoding techniques are also dictated by the particular application. The image must be coded in such a manner so as to maintain an acceptable representation of the original image while preserving the information required for the application. The techniques also employ as little storage as possible so as to facilitate a high processing speed.

Image segmentation and representation classifies the features of an image to enhance the speed and quality of subsequent feature extraction and scene analysis operations. The most common techniques in this category

include edge extraction methods by grey level thresholding which scan the image for significant changes in adjacent grey levels to enhance the edges in the image, as well as template matching which utilizes a mask of a known object or feature to detect a common feature pattern in an image.

2.3 The Object Recognition Problem

The objective of this thesis was to develop a set of microcomputer based vision algorithms for object recognition, and then to determine the orientation and shape parameters associated with an object. The ultimate goal was to use this information module as an interface to the robot manufacturing cell presently in place at the Automation / Manufacturing Laboratory of the University of Manitoba. The objects used in this project were restricted to simple and regular shapes such as a square, a rectangle, a circle, an equilateral triangle, and an isosceles triangle. It also assumed that at any given time only one object would be present in the scene. The objects appeared before the camera at distances varying from 7" to 24" from the camera lens. As a result approximately one quarter of the camera's field of view is utilized. These restrictions were imposed primarily to standardize the experimental setup and can be modified in the future.

2.4 General Hardware and Program Structure

The camera used in this project was a Panasonic color video camera (model no. WV-3240). The image capturing and subsequent display was achieved through the use of a Frame Grabber board manufactured by Data Translation (model no. DT-2853-SQ-60Hz), which provided for square pixel display (1:1 pixel aspect ratio). The Frame Grabber board contains a 16-bit analog to digital (A/D) converter on-board which has a throughput of 100,000 samples per second, and a frame can be grabbed in approximately 1/30 of a second. This board also contains two memory buffers, each capable of containing an entire 512 x 512 pixel image with 256 grey levels.

The Frame Grabber board is IBM compatible and is used in a MIND AT microcomputer (IBM compatible) with a 8087 processor and a more powerful 80287 coprocessor. The images are displayed on an RS-170 standard RGB video display monitor.

The program developed to recognize the objects was written using Microsoft Fortran V3.30 [8]. These programs were linked to a specific software module supplied with the Data Translation board. The subroutine library (IRIS) has been designed for real time image processing [9]. The library provides not only the communication between the camera and the computer, but also has the capability to perform a number of operations and image manipulations very

quickly. The routines developed as part of this work may be divided into four categories: thresholding, edge detection, feature extraction, and object recognition. The overall development of the program will be described in Chapter III in this thesis.

2.5 Thresholding

The image dealt with in this study is a 512 x 512 pixel array with 256 grey levels of image. Grey levels, in this case, range from "black" represented traditionally by a value of "0" to "white" represented as "255". The intermediate values represent different tones of grey present in the image. Thresholding is used to reduce the grey level image to a binary one. Since the application deals only with simple shapes, it is sufficient to use a binary image as opposed to a detailed multi-leveled grey image. Thus a reduced data set is used without losing any information. Since lighting and contrast were fixed, consistent results were obtained with a fixed threshold value. If there was less contrast between the object and the background, or if shadows created a significant effect in the scene, an adaptive thresholding routine may have to be incorporated into the program to alleviate these problems.

Thresholding an image involves selecting a grey level, above which all grey levels are assigned the level 255, or

"white". All grey levels below this limit are assigned the value 0, or "black". For the present study the objects were made out of black cardboard and were firmly attached to white cards to provide a sharp contrast between the object and the background. Thus, the black objects on the white cards produced a maximum contrast, and a threshold level of 127 was chosen as the boundary point between the two. If there was a lot less contrast between the background and the object, the most suitable threshold level could be determined by experimental methods.

2.6 Edge Detection

Once thresholded, an image detection algorithm is used to scan the image for an object, namely 0's, and then follow the contour of the object to extract the coordinates of the edge points. It was assumed that the image is a reasonable one and that all 0's represent part of the object only and do not belong to any isolated smudge or noise outside the edges of the object. This process greatly reduces the amount of data used to represent the image. In addition, it reduces the number of times the image must be accessed to obtain all the information necessary for object recognition. Further, all the features required in this application may be determined using the edge points only.

Other alternative methods such as run length encoding or scan line encoding could be used to locate the object and

would also serve as an image compression mechanism at the same time.

2.7 Feature Extraction and Object Recognition

The feature extraction and object recognition algorithms analyze the edge points to obtain the information required to determine the type of object present and its appropriate features and orientation characteristics. The features extracted for this study are the area, the geometric center, and the coordinates of the corners of the object.

Object recognition utilizes the number of corners, the length of sides, and the radial lengths from the geometric center to the corners, or "half-diagonals". These then determine the type of object present in the field of view of the camera.

The two categories, feature extraction and object recognition, overlap when distinguishing the circle from the rest of the objects. Since a circular object does not contain any corners, the presence of a circular object is determined before all the aforementioned features are extracted. The percentage of equal radial lengths determines whether the program is diverted to the circle test section or continues to calculate the corner coordinates of one of the other objects.

The algorithms required to extract the features and orientation of the objects are detailed in the next chapter.

CHAPTER III: VISION APPLICATION PROGRAM

3.1 General Structure of the Program

Once a raw image has been obtained by a camera, the steps required for the development of object recognition and feature identification begin. This chapter details the set of routines required to distinguish the given objects from one another. All of the techniques used are spatial-domain methods; that is, they manipulate the individual pixels of the image, as contrasted to frequency-domain methods (such as Fourier transform methods) which modify the transform of an image. The choice was made after a careful review of material on related subjects [10].

The application programs were developed in two parts. The lack of sufficient memory in the microcomputer prevented the programs from being combined. The first module referred to as " **TSTLOAD** " hereafter, contains the steps employed to acquire an adequately thresholded binary image from the Frame Grabber. It also contains routines used to obtain the edge coordinates of the object. The edge coordinates are then stored in a data file. The second module referred to as " **TSTIMAGE** ", performs retrieval of the file containing the coordinates and also contains the algorithms required to identify the object. This chapter focuses on various aspects related to the two modules of the software program.

3.2 Obtaining an Image and Extracting the Edge

As previously stated, the first step toward object recognition is to acquire an image from the camera. To access the camera through the Data Translation (DT) Frame Grabber hardware, the program utilizes the subroutine library (IRIS) provided with the DT software. These subroutines contain the information required to set up the system to accept images, and provide access to the optional filtering, thresholding, and various other image processing utilities. This section explains only those routines utilized to obtain the image required for the analysis developed in this thesis. These subroutines are identified throughout this chapter as those in bold print beginning with the letters "IS" and their full description can be found in the DT IRIS Manual [9].

Before the camera can be accessed, the system must be initialized. This is achieved by calling the subroutine with the syntax **ISINIT()** which initiates access to the DT Frame Grabber. From now on the DT IRIS subroutines will be denoted by their appropriate syntax. Next, the system is commanded to accept the signal from the external source, namely the camera, by calling the routine **ISSYNC(1)** (where the number "1" denotes the external source).

The on-board Data Translation hardware provides two frame buffers labelled 0 and 1. These are used to hold the 512 x 512 pixel image received by the camera (input frame buffer),

and the processed image displayed by the software (output frame buffer). However, the algorithms developed here require very little processing of the image before extracting features. Therefore, to increase the overall processing speed, the same buffer (0) is used to input the image and display the image. This is accomplished by calling the **ISINFR(0)** and **ISOTFR(0)** routines which set the input and output frame buffers, respectively. Frame buffer 0 is then cleared by calling the **ISFCLR(0)** routine.

Once the hardware is configured to accept an image, the next step is to ready the system to threshold the captured image. This can be accomplished by making use of one of the seven input and output look-up tables (**LUT's**). These LUT's allow the user to manipulate the 256 grey levels to suit a variety of purposes, such as creating negatives of a scene or assigning multi-level thresholding values. The software must use an input and output LUT, and if no manipulation is required, the system defaults these values to the identity table (0) which leaves the input and output scenes intact. The table used to convert the image to a binary one is LUT 6, and is called through the **ISINTS(6)** subroutine. The table converts pixels in the intensity grey level range of 0 to 127, to a value of 0. This corresponds to the black object in the scene. Any pixel with an intensity grey level in the 128 to 255 range is converted to 255. This corresponds to the full intensity of the white

used in each scene. In this manner, each individual pixel in the image is converted to black or white, creating the desired binary image required for further processing, and reducing the amount of memory required to hold an image.

Since no further manipulation of the pixels is required, the output LUT used in the program corresponds to the identity table, LUT 0. The output LUT is the table by which the scene in buffer 0 is filtered through before being displayed on the monitor. This function is called from the routine **ISOUTS(0)**, and leaves the thresholded binary image as is. Thus, when the display of the output frame buffer is enabled, the image seen on the RGB monitor is the thresholded binary image processed through the input LUT.

Finally, the program uses the "pass-thru" mode of image acquisition by calling the **ISPASS()** routine, which commands the camera and Frame Grabber board to monitor the scene. The camera continues to do so until it is interrupted by the **ISFREZ()** routine which instructs the system to "freeze" the current image. The frozen image, or frame, is grabbed and held, and subsequently thresholded by the **ISINTS(6)** command. The data is then stored in buffer 0 and automatically displayed on the monitor. The data stored is then ready for edge extraction.

3.3 Location and Accessing of Image

The image residing in the on-board buffer 0 will be

designated as **IMAGE(ROW,COL)**, where both **ROW** and **COL** (column) are 512 pixels each as shown in Figure 1. The object typically occupies no more than one third of the pixels in **IMAGE**, and the computer memory available to run the program is limited. In order to maintain the speed of processing, the program must use the least amount of data necessary without losing any pertinent information. This is accomplished by utilizing a **WINDOW** array with **IMAGE** that holds 64 rows of 512 columns. The **WINDOW** array reduces the amount of **IMAGE** data by removing those rows of the **IMAGE** that do not contain part of the object. This increases the speed of processing as the on-board buffer need only be accessed a few times rather than for each pixel. It could be expanded to remove all the unnecessary columns, but this would not create any fundamental challenge. If a boundary of **WINDOW** is accessed by the program, **WINDOW** is updated to hold the information area required. The **WINDOW** updating process is illustrated in Figure 1.

For example, the edge point "n" in Figure 1, which has a **WINDOW** array row coordinate of 64 corresponding to an **IMAGE** array row value of 72, is located and its **IMAGE** coordinates are extracted and put into their row and column arrays designated as **BORDR** and **BORDC** respectively. Thus **BORDR(n)** is 72 and **BORDC(n)** is "a" in the **IMAGE** array. The search begins for the next point on the edge of the object. Since point "n" is on the lower boundary of the **WINDOW** array

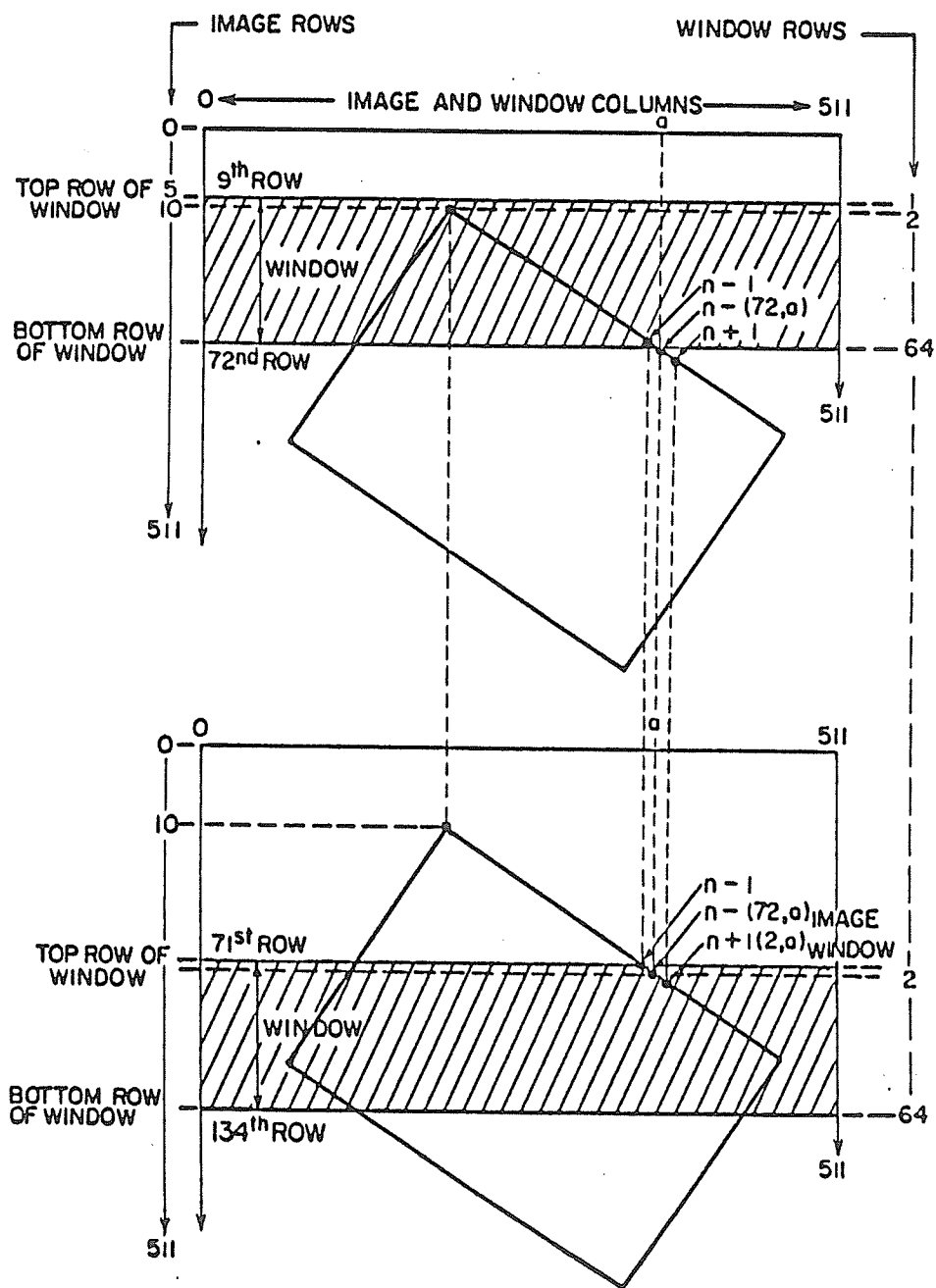


FIGURE 1: Illustration of Window Updating Process

containing the edge information, the array must be updated to provide continuous edge data for the contour following algorithm. The algorithm requires that all pixels surrounding a located edge point "n" must be present in the WINDOW array at the same time. Thus the WINDOW array is updated in such a manner as to contain the located edge point "n", and all of the eight surrounding points. The top row in the WINDOW is therefore updated to row 71 of the IMAGE array, and the bottom row of the WINDOW is row 134 of the IMAGE array. The edge point "n" now resides at coordinate (2,a) of the WINDOW array, and the data is now present to continue the search for the next edge point "n+1".

The array WINDOW(64,512) is filled by calling the DT IRIS routine ISGETP . The full syntax of ISGETP is:

ISGETP(BUFFER,ROW,COL,COUNT,IBUFF)

where **BUFFER** is the DT on-board buffer in which the image resides, namely buffer 0. **ROW** and **COL** are the coordinates of the point from which the retrieval is to start, **COUNT** is the number of bytes, or pixels, to be retrieved and **IBUFF** is the array in which the information is to be stored. There were problems associated with the hardware and software system that temporarily restricted the number of pixels that could be retrieved at one time. Therefore, each row is individually retrieved and stored in IBUFF rather than the entire 64 rows at a time. This however can be quickly

modified in the future. Thus, groups of 64 rows of IMAGE(512,512) are accessed individually by calling:

ISGETP(0,ROW,0,512,IBUFF)

64 times. Each row called is then stored in IBUFF and transferred to its appropriate position in WINDOW(64,512).

3.4 Edge Detection

This routine accesses the array IMAGE in buffer 0 and extracts the edge coordinates of the single object found therein. It is assumed that the image is a reasonable one and that all 0's represent part of the object, and not an isolated smudge outside the edges of the object.

First, a row scan is performed to find the first point on the edge of the object. The rows are accessed starting from row 0 of IMAGE using the ISGETP routine which stores a row in IBUFF(512). The row is searched for a pixel whose intensity level is 0, or "black". If no 0 is located in a row, the next row is scanned from left to right. This continues until the first pixel whose intensity is 0 is located. This point is on the edge of the object and its coordinates are labelled FROW,FCOL (row and column coordinates respectively). Two arrays, BORDR and BORDC, are allocated to contain the row and column coordinates respectively. The initial coordinates, BORDR(1) and BORDC(1) are set to FROW and FCOL respectively. The search for the first point on the edge is illustrated in Figure 2. The first

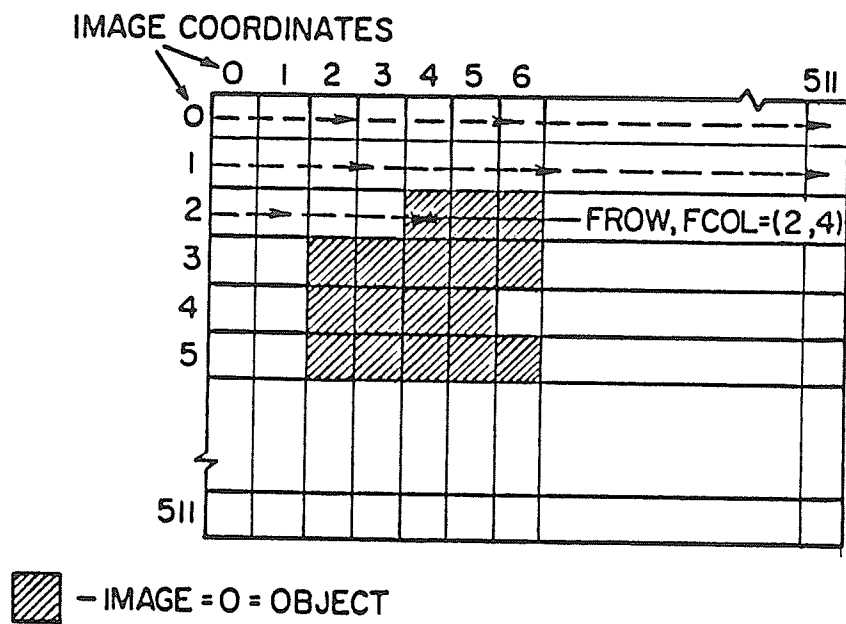


FIGURE 2: Locating the Initial Point

point located in Figure 2 corresponds to FROW=2 and FCOL=4.

Once the first point is located, a contour following algorithm is used to follow the edge in a clockwise direction, extracting the coordinates of each successive edge point as and when it is found. The WINDOW(64,512) array is utilized to provide access to the IMAGE(512,512) residing in the on-board DT memory buffer 0, and is updated by those methods outlined in the previous section. For solid shapes, the edge of the object consists of a set of contiguous pixels. Therefore, the contour algorithm will continue until the first point is relocated.

To effect this, the first problem is to determine which neighboring pixel is the next one on the edge. For each point, there are eight surrounding pixels. Figure 3 shows the nine pixels with the "current" or last confirmed edge point, or pixel, in the center. Since the edge is a set of contiguous points, it follows that at least two of the surrounding eight pixels are also part of the edge. These two pixels denote the previous point on the edge and the next point on the edge. At each stage of the edge point extraction process, this block of nine pixels is mapped to the edge with the known or current edge point in the center. The next step of the search is to find which of the surrounding eight pixels is the next edge point to be found in a clockwise direction from the current point. The clockwise search direction, which was arbitrarily chosen,

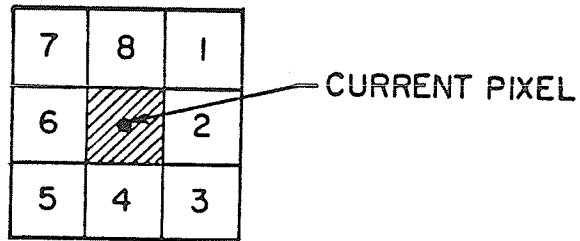


FIGURE 3: Eight Surrounding Pixels

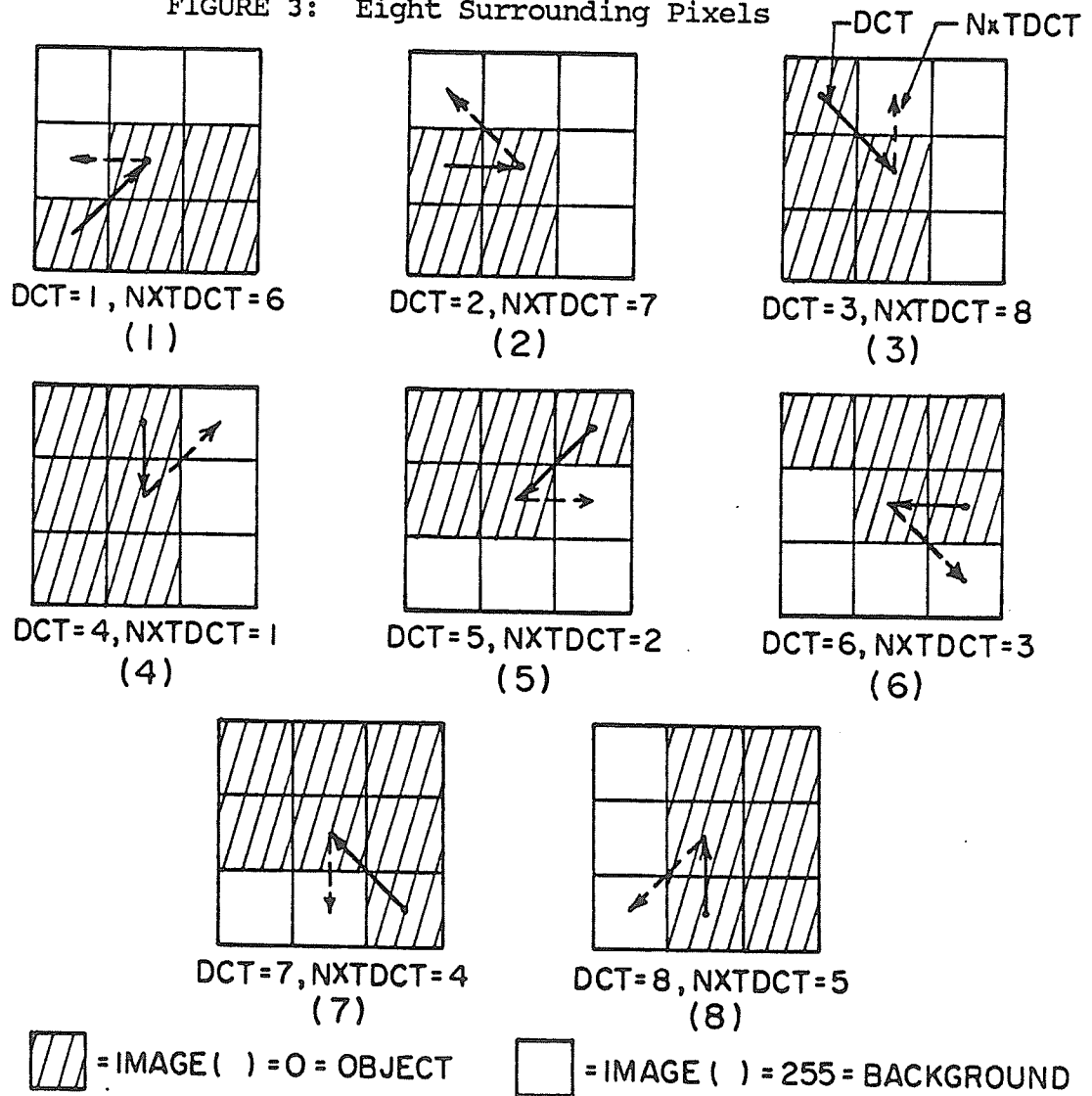


FIGURE 4: Eight Possible Directions to Start
Search for Next Pixel

ensures that the object will always be to the right of the edge and the background to the left of the edge.

The direction by which the search is initiated, from the previous pixel to the current pixel, is known and this can be used to determine the direction to begin the search for the next pixel. Figure 4 shows the eight possible directions denoted as "DCT" by which a point can be found, and the corresponding initial direction to begin the search for the next edge point referred to as "NXTDCT". In the illustration shown, the object is shown cross-hatched. The initial direction is shown as a bold line and the next direction is shown as a dotted line in Figure 4. In each case, the next direction is shifted clockwise by one direction with respect to the initial direction. Similar approaches have been discussed in references [11,12].

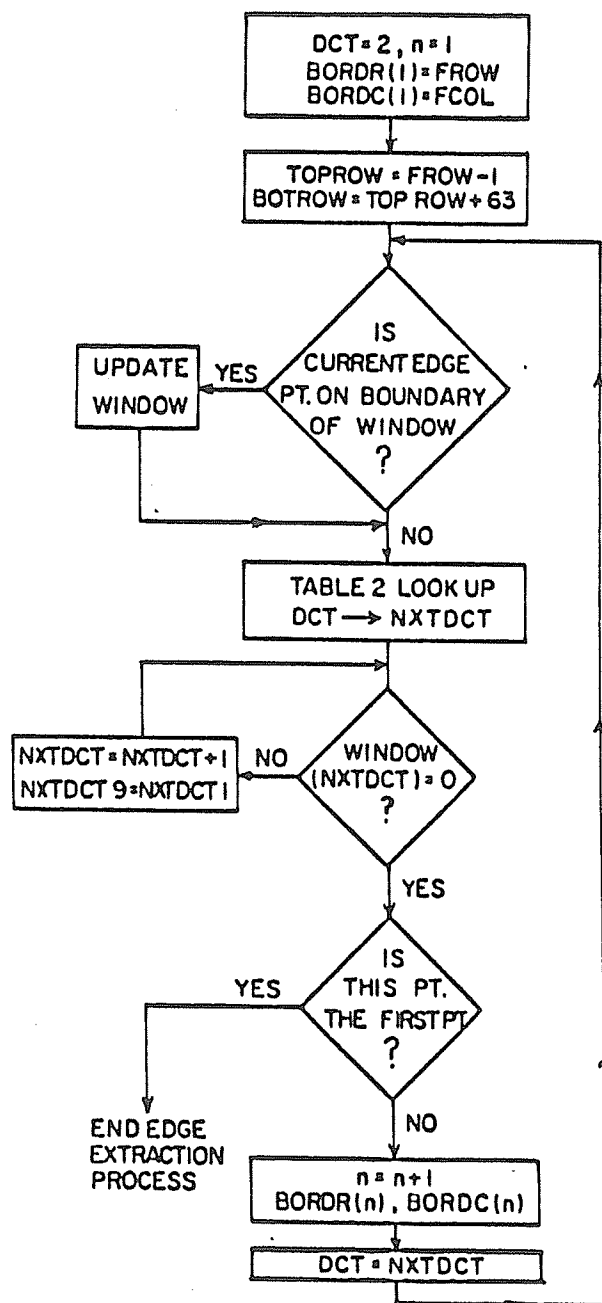
TABLE 1: Initial Directions and Next Directions

Last direction taken to locate current edge point (DCT)	First direction taken to start search for next edge point (NXTDCT)
1	6
2	7
3	8
4	1
5	2
6	3
7	4
8	5

Table 1 shown above lists these pairs of directions and next directions. For example, if the current point was found by moving from the surrounding point direction number 7, as shown in Figure 4 number 3, it is found by moving in direction 3 (DCT=3) from the previous point. According to Table 1, the search for the next edge point must begin by moving in direction 8. However, this particular pixel is not part of the object or edge. Thus the direction is incremented by one direction clockwise, and the search continues in this manner until the next edge point is located. The coordinates of this point are placed in their respective BORDR and BORDC arrays. The next point is then updated to the status of "current" point, and the search continues. A flow chart depicting this process is illustrated in Figure 5.

Figure 6 illustrates a sample object contour. The directions taken to locate each edge point are shown as arrows pointing into the center of the point in question. The arrows emanating from the center of the points indicate the initial direction used to start the search for the next edge point.

Edge point number 4 is found by moving from the current point, located at coordinate (2,6), and searching consecutive directions shown in line 4 of Table 2. The arrow pointing out of the current point is the next direction to take to start the search for point 4, and



- 1st edge pt. located by row scan (DCT=2).
- n= edge point number.
- BORDR(1), BORDC(1) set to coords. of 1st pt. (FROW, FCOL).
- Set WINDOW(64,512) boundaries:
TOPROW = row above 1st pt.
BOTROW = TOPROW + 63.
- Is current edge pt. on a boundary of WINDOW ?
- YES: WINDOW updated to contain current pixel & ALL eight surrounding pixels.
- NO: From current DCT obtain NXTDCT (See TABLE 2 for these).
- Is Pixel at location NXTDCT part of object? (See FIGURE 3 for location of pixels).
- NO: Increment NXTDCT by one direction clockwise.
- YES: Has first point been relocated?
- YES: All pts. found; end edge extraction.
- NO: Increment edge pt. no. & store edge pt. coords. in BORDR, BORDC.
- Update current pt. and direction DCT.
- Repeat Loop until 1st pt. relocated.

FIGURE 5: Flow Chart of Edge Extraction Process

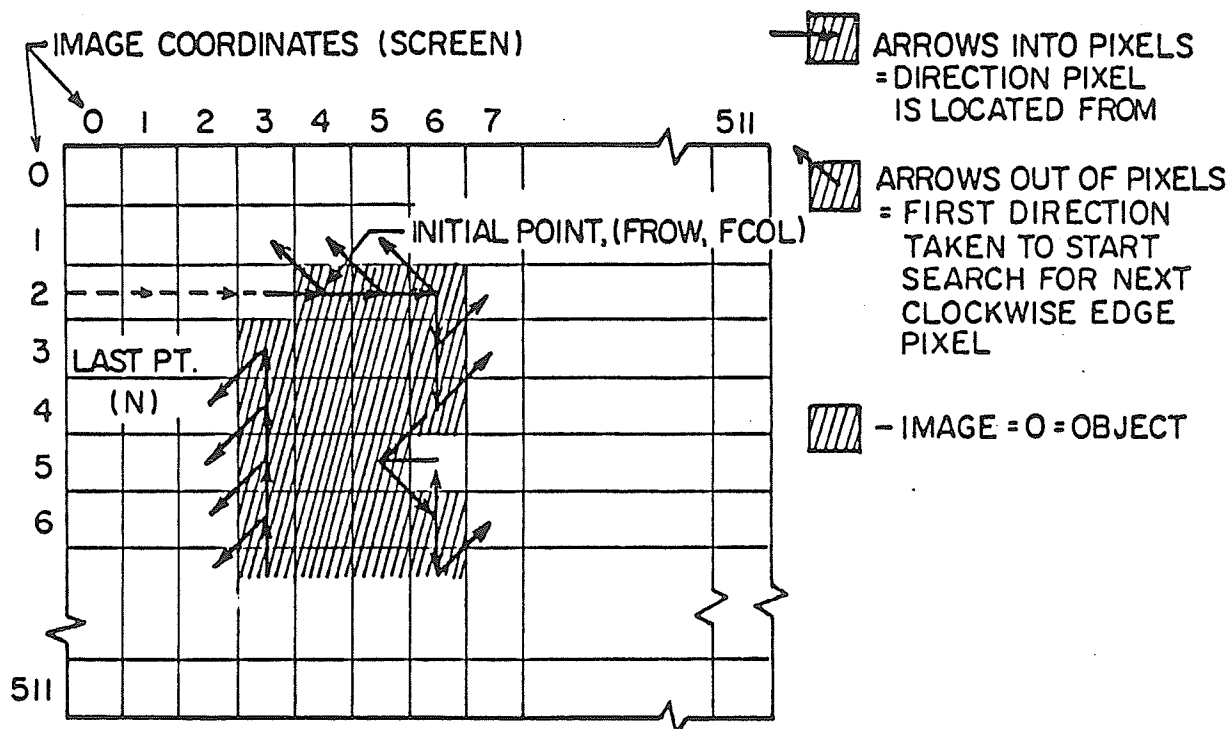


FIGURE 6: Sample Edge Following Method

TABLE 2: Sample Edge Following Steps

Point #	Directions Taken to locate current Point (DCT's)	Last DCT	Point Coords.	Next DCT to start search for NEXT pt.
1	2	2	(2,4)	7
2	7,8,1,2	2	(2,5)	7
3	7,8,1,2	2	(2,6)	7
4	7,8,1,2,3,4	4	(3,6)	1
5	1,2,3,4	4	(4,6)	1
6	1,2,3,4,5	5	(5,5)	2
7	2,3	3	(6,6)	8
.	.	.	.	.
.	.	.	.	.
N-2	5,6,7,8	8	(5,3)	5
N-1	5,6,7,8	8	(4,3)	5
N	5,6,7,8	8	(3,3)	5
N+1	5,6,7,8,1	1	FROW, FCOL	End List

corresponds to direction 7 listed in column 5 of Table 2 for point 3. It is also the first direction listed in the second column of Table 2 for pixel 4. However, this location does not reveal a point on the object. Therefore the direction is incremented clockwise by one direction and the process is repeated until an object point is located. The list of directions searched before locating the next point is shown in Table 2, column 2 for point 4. After searching in directions 7, 8, 1, 2 and 3, and edge point is located by searching in direction 4 from the current pixel (2,6). The coordinates of this point are stored in the BORDR and BORDC arrays. The current point is updated to the 4th point, and the direction taken to locate this point (direction 4) is used to determine the first direction to start searching for point 5, which is direction 1. These steps are all shown in line 4 of Table 2. This table also shows the steps taken to find the rest of the points illustrated in Figure 6.

3.5 Feature Extraction and Object Recognition

In order to recognize an object, a set of parameters must be chosen to differentiate uniquely one object from another. The objects dealt with in this thesis are simple, regular shapes, and are limited to a square, a rectangle, a circle, an equilateral triangle, and an isosceles triangle. Any other object is considered as unknown. The number of

maximal radii from the geometric center is one of the fundamental parameters used to decide which object is present. They represent the number of corners present in the object. Once the number of corners present has been determined, other aspects such as lengths of sides and half-diagonals can be calculated and compared to determine the type of object present in the image.

The features required for object recognition are extracted from the edge coordinates stored in arrays BORDR and BORDC. From these arrays, the geometric center (X_G, Y_G) is calculated and the distances from that center to points along the edge are used to find all maxima or corners.

3.5.1 Geometric Center

The geometric center of an object can be found by applying Green's formula [10] for extracting features using discrete border points. The edge coordinates can also be used to determine the area of an object.

The general form of Green's formula for a planar region is given by:

$$\iint_R \left[\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y} \right] dx dy = \int_{\vec{C}} [P dx + Q dy] \quad (3-1)$$

where $\vec{C}$ is along the contour of the region R taken in a counterclockwise direction. The area A of an object can be defined as:

$$A = \iint_R dx dy \quad (3-2)$$

To get the area from Green's formula, the variables Q and P in equation (3-1) are set to x and zero respectively. Thus for N points on the border (the (N+1)th is the first), area A becomes:

$$\begin{aligned} A &= \int_C x \cdot dy = \sum_{n=1}^N \left[\frac{(x_{n+1} + x_n)}{2} \frac{(y_{n+1} - y_n)}{2} \right] \\ &= 1/2 \sum_{n=1}^N [x_n y_{n+1} - x_{n+1} y_n] + \underbrace{1/2 \sum_{n=1}^N [x_{n+1} y_{n+1} - x_n y_n]}_B \end{aligned} \quad (3-3)$$

An expansion of the second term denoted as "B" in equation (3-3) yields:

$$\begin{aligned} B &= 1/2 [(x_2 y_2 - x_1 y_1) + (x_3 y_3 - x_2 y_2) + (x_4 y_4 - x_3 y_3) + \dots \\ &\dots + (x_{N-1} y_{N-1} - x_{N-2} y_{N-2}) + (x_N y_N - x_{N-1} y_{N-1}) + (x_{N+1} y_{N+1} - x_N y_N)] \end{aligned} \quad (3-4)$$

This can be further simplified as:

$$B = 1/2 [(-x_1 y_1) + (x_{N+1} y_{N+1})] \quad (3-5)$$

Point (N+1) is the first point for an enclosed edge. Hence B = 0, and area A reduces to a simplified equation:

$$A = 1/2 \sum_{n=1}^N [x_n y_{n+1} - x_{n+1} y_n] \quad (3-6)$$

Similarly, Green's formula is used to calculate the coordinates of the geometric center (XG,YG) of the enclosed object. The equations for XG and YG over an entire region are:

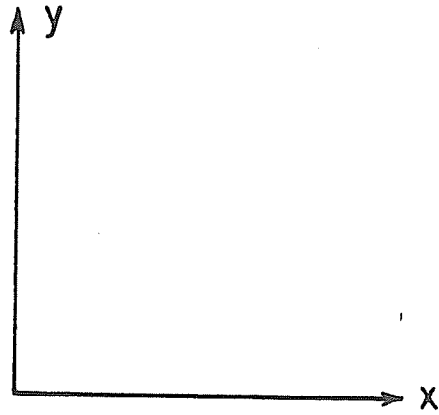
$$XG = 1/A \iint_R x \cdot dx \cdot dy \quad \text{and} \quad YG = 1/A \iint_R y \cdot dx \cdot dy \quad (3-7)$$

Conversion of these equations using Green's formula gives XG and YG as:

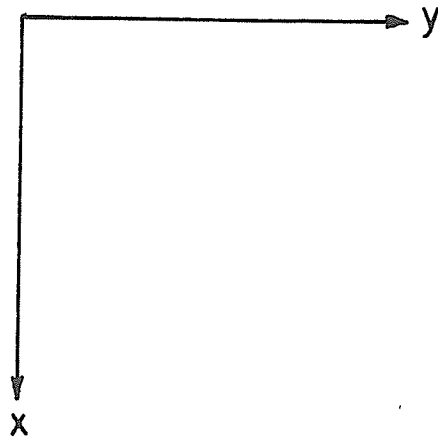
$$XG = 1/(2A) \sum_{n=1}^N x_n^2 (y_{n+1} - y_n); \quad (3-8)$$

$$\text{and} \quad YG = -1/(2A) \sum_{n=1}^N y_n^2 (x_{n+1} - x_n). \quad (3-9)$$

Thus, the calculations for the geometric center of an object can be performed using equations (3-6), (3-8) and (3-9) given above. The x and y in th equations are replaced by the array values of BORDR and BORDC, respectively. The orientation of the axes in Green's formula shown in Figure 7 part A consists of x being horizontal and Y being vertically upward. Figure 7 part B indicates the orientation of these axes required for this application. It can be seen from this diagram that the orientation required by Green's formulae is maintained.



(A) Green's Theorem Configuration



(B) Thesis Application Configuration

FIGURE 7: Axes for Green's Theorem and Application

3.5.2 Detection of Maximal Radii

The next step of the object recognition problem is to find the maximal radii of the object, which are defined as the points further away from the geometric center than their surrounding edge points. To differentiate these points from the rest, the distances from the edge points to the center (X_G, Y_G) are first calculated.

It is assumed that a relatively reasonable image may be obtained from the Frame Grabber with minimal noise. This is a reasonable assumption since vision systems generally require a very controlled lighting environment. However the image can still be non-ideal. The edge of an object will likely contain some noise that may not be eliminated by the pre-processing algorithms. The digitization produces this. Therefore, calculating distances to each individual point could yield inaccurate results, such as a false corner. To eliminate this, groups of a fixed number of adjacent points along an edge, represented by parameter "NUMAVE", are averaged together to find the average row and column coordinates (X_{AVE}, Y_{AVE}), associated with each group. It is this average point that is used to calculate the distance to the geometric center.

The optimum number of consecutive points (NUMAVE) to be averaged was determined through analysis of typical data, which is presented in Chapter IV. The number must be kept to a minimum to produce high speed computations. In order

to obtain an accurate result, each averaged point (real number coordinate) is associated with an actual point on the edge (integer coordinate). Therefore the averaging of points must be done in an unbiased manner so that the center point has a true averaged coordinate associated with it. This implies that NUMAVE must be an odd whole number. The distances from the geometric center to the averaged coordinates are calculated and stored in an array **DSTNC(n)**, where n is the edge point number. An example to illustrate this is shown in Figure 8. The corresponding data is listed in Table 3. The software routine wraps the ends of the coordinate list together to assure continuity along the end points when calculating the DSTNC array.

Once the distances are calculated, the algorithm developed must search them for the maximums, or corners. These can be found by first extracting the increasing and decreasing trends from the data. A **TRANS(n)** array is set up to indicate how the distance is changing from one point to the next. The distances for two successive edge points are compared to determine whether point "n" is longer, shorter, or equal to point "n-1". If it is found to be longer, the **TRANS(n)** value is set to +1. If the distance is found to be shorter, the **TRANS(n)** value is set to -1, and if they are equal, **TRANS(n)** is set to 0. A variable "**RADITOL**" is introduced to represent the tolerance allowed for two distances to be deemed as equal.

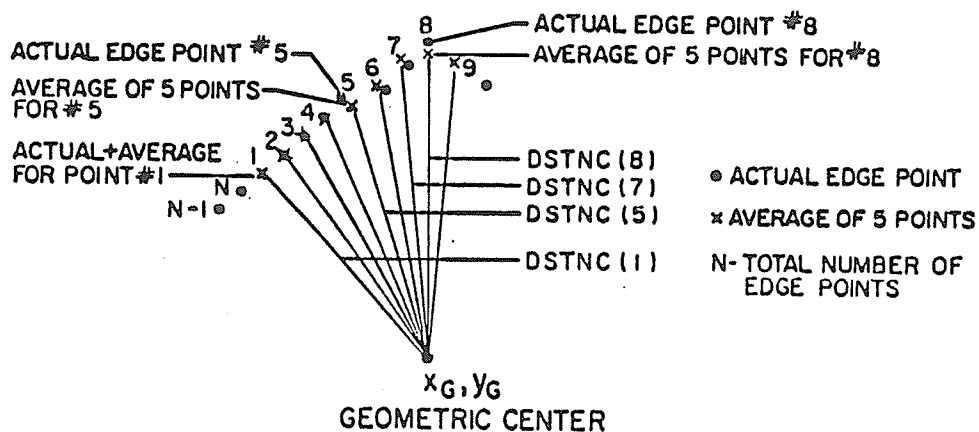


FIGURE 8: Setting Up the Array DSTNC

TABLE 3: Calculation of DSTNC Array

Edge pt. n	Coordinates		Distance $x_g y_g$ to $\bar{x}_n \bar{y}_n$ DSTNC(n)
	Actual point x_n, y_n	Average point $\bar{x}_n, \bar{y}_n$ (NUMAVE = 5)	
N-1	x_{N-1}, y_{N-1}	$\bar{x}_{N-1}, \bar{y}_{N-1}$	DSTNC(N-1)
N	x_N, y_N	$\bar{x}_N, \bar{y}_N$	DSTNC(N)
1	x_1, y_1	$\bar{x}_1, \bar{y}_1$	DSTNC(1)
2	x_2, y_2	$\bar{x}_2, \bar{y}_2$	DSTNC(2)
3	x_3, y_3	$\bar{x}_3, \bar{y}_3$	DSTNC(3)
4	x_4, y_4	$\bar{x}_4, \bar{y}_4$	DSTNC(4)
5	x_5, y_5	$\bar{x}_5, \bar{y}_5$	DSTNC(5)
6	x_6, y_6	$\bar{x}_6, \bar{y}_6$	DSTNC(6)

For example, an object with several edge points and their corresponding distances is illustrated in Figure 9. The following relations are used to determine the values of TRANS(n) for the various points shown in the figure.

$$\frac{DSTNC(N+1)-DSTNC(n)}{DSTNC(n+1)} > +RADITOL : TRANS(n+1)=1 \quad (3-10)$$

$$\frac{DSTNC(N+2)-DSTNC(n+1)}{DSTNC(n+2)} < -RADITOL : TRANS(n+1)=1 \quad (3-10)$$

$$-RADITOL \leq \frac{DSTNC(m+1)-DSTNC(m)}{DSTNC(m+1)} \leq +RADITOL : TRANS(m+1)=0 \quad (3-11)$$

$$-RADITOL \leq \frac{DSTNC(m+2)-DSTNC(m+1)}{DSTNC(m+2)} \leq +RADITOL : TRANS(m+2)=0 \quad (3-12)$$

Ideally all TRANS element values would be zero for a circle, since the distance from the geometric center to every point would equal the radius. Similarly, the TRANS array for the square and triangle would consist entirely of chains of 1's and -1's with no 0's. This is illustrated in Figure 10.

In reality, due the inherent discrete nature of the data, the circle's TRANS array may contain some 1's or -1's. Similarly, the array for the square or triangle may contain some 0's. However, unless the image data are extremely corrupted, these irregularities in the TRANS array will make up only a small percentage of the entries. The image from the Frame Grabber was analyzed to verify this assumption. Hence, accounting for a small percentage of non-zero numbers in the TRANS array, the object may be

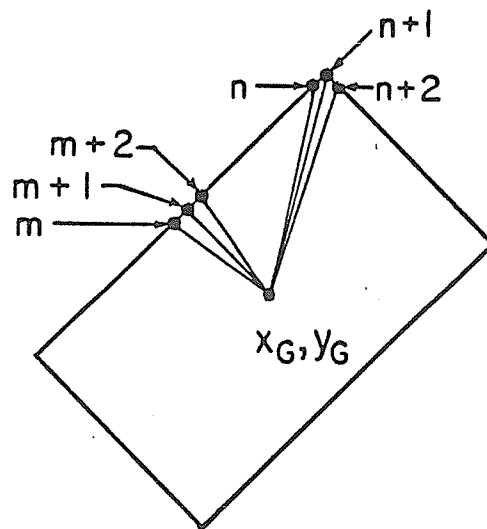


FIGURE 9: Setting Up the Array TRANS

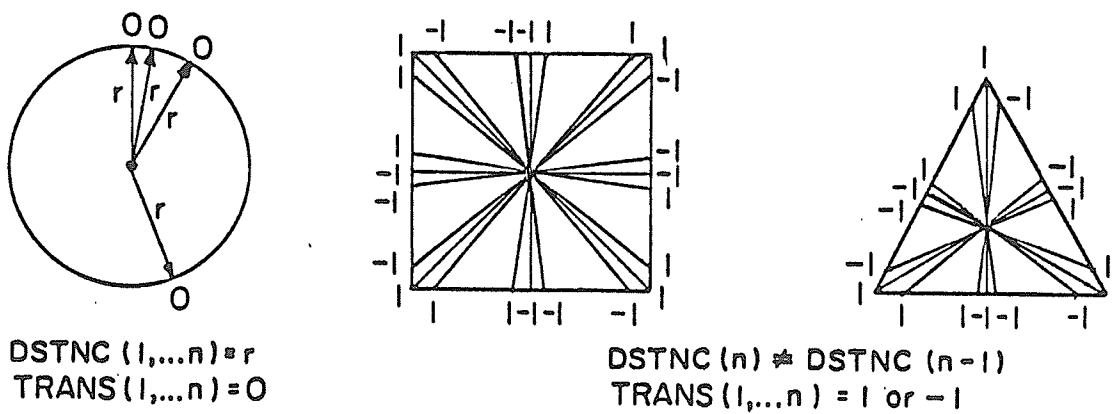


FIGURE 10: Ideal TRANS Values for Typical Objects

identified as being circular. Parameter "CIRCL" was introduced as the minimum percentage of zero's a circle will have. Its optimum value was determined experimentally and is discussed in Chapter IV.

If the percentage of zero's in the TRANS array is less than parameter CIRCL, the object will be identified as not being a circle, and ideally, the TRANS array should not contain any zero's. However, as mentioned above, it may have a few 0's present due to some noise or resolution problems. Thus, a scan is performed on the TRANS array to eliminate all 0's that are imbedded between identical non-zero TRANS avalues and replace them with the non-zero value as shown in box B in Figure 11. Those zero's not imbedded between identical values such as those marked in box A of Figure 11 are left alone. In this manner, irregularities can be eliminated without distorting the true image.

From the structure of the TRANS array, it is obvious that all single 1's or -1's imbedded between identical values in the array can only be noise or digitization errors. Therefore a second scan, which acts like a noise filter, is performed on the TRANS array to eliminate transient 1's and -1's, as shown in boxes D and E in Figure 11. All transient 1's or -1's embedded between non-identical TRANS values, such as those shown within box C in Figure 11, are left alone. This process is valid for straight edges, but if

ORIGINAL
TRANS
ARRAY → 1, 1, 1, 1, 1, A
0, -1, -1, -1, -1, -1, B
0, 0, -1, -1, ...

TRANS
ARRAY
AFTER
ADJUSTMENT → 1, 1, 1, 1, 1, 0, -1, -1, -1, -1, -1, -1, -1, -1, -1, ...

ORIGINAL
TRANS
ARRAY → 1, 1, 1, C
-1, -1, 1, 1, 1, D
-1, 1, 1, -1, -1, -1, -1, E
1, -1, -1, ...

TRANS
ARRAY
AFTER
ELIMINATION → 1, 1, 1, -1, -1, 1, 1, 1, 1, 1, 1, -1, -1, -1, -1, -1, -1, -1, ...

FIGURE 11: Modified TRANS Arrays

more complex shapes are used this method may break down by locating maxima's in curved edges.

The resulting TRANS array for polygons essentially contains chains of 1's and -1's, with a small percentage of 0's interspersed, mostly around the midpoints of the sides. The ideal case would consist of eight chains of alternating 1's and -1's for a square, and similarly six strings for a triangle as shown in Figure 10. The chains of 1's would denote an increase in radial length towards a corner, and the chains of -1's would indicate a decrease in the radial distance moving away from a corner and toward a side midpoint. Therefore, the next step in the algorithm is to search for these chains or **trends** in the TRANS array.

3.5.3 Identifying Transition Points

A trend is defined as increasing or decreasing based on the values of a chain of adjacent points. For example, an increasing trend would have a chain of consecutive points each with TRANS values of 1, and a decreasing trend would have a chain of consecutive points with TRANS values of -1. The number of points included in each chain is represented by a variable "STLEN". For the analysis in this thesis, STLEN has a value of 8. The justification for this is discussed in Chapter IV.

Once a trend is identified, the search continues until the opposite trend has been located. That is, if one is on

an increasing trend (+1's), the scan continues until a chain of points indicating a decreasing trend (-1's) is found. A transition area has now been located and is defined by the last chain of points of the first trend and the first chain of points of the second trend. There may be a mixture of 0's, 1's or -1's between the two trends. The coordinates of the first and last points of the first trend (X1,Y1 and X2,Y2), and the endpoints of the second trend (X3,Y3 and X4,Y4), are extracted and placed consecutively in arrays ARX and ARY (row and column respectively). Their respective TRANS values are put into array TREND. The search continues until the first trend identified has been relocated. A sample scan is illustrated in Figure 12. The table below indicates the points whose coordinates are extracted from the list and are used as markers (n) for the transition areas.

POINT n	ARX(n),ARY(n)	TREND(n)
1	a	1
2	b	1
3	c	-1
4	d	-1
5	e	-1
6	f	-1
7	g	1
8	h	1

After all the transition areas are located and entered into arrays ARX and ARY, the transition points are calculated. Whenever two lines have a definite

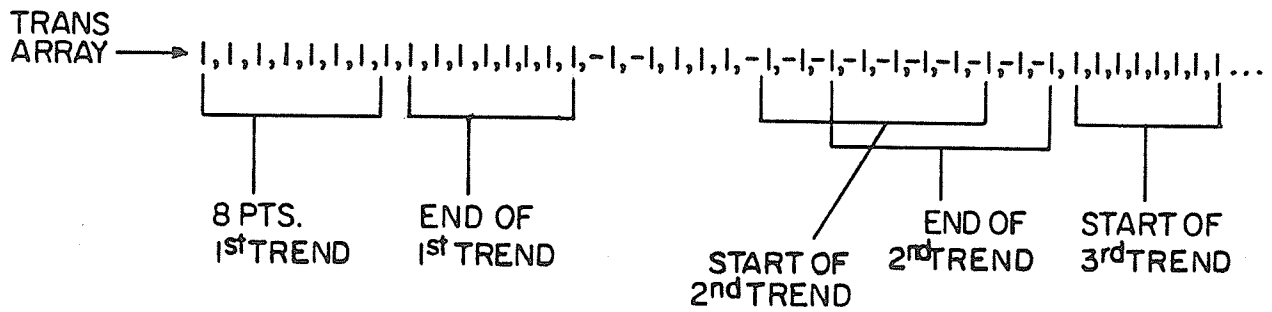


FIGURE 12: Illustration of Trend Search

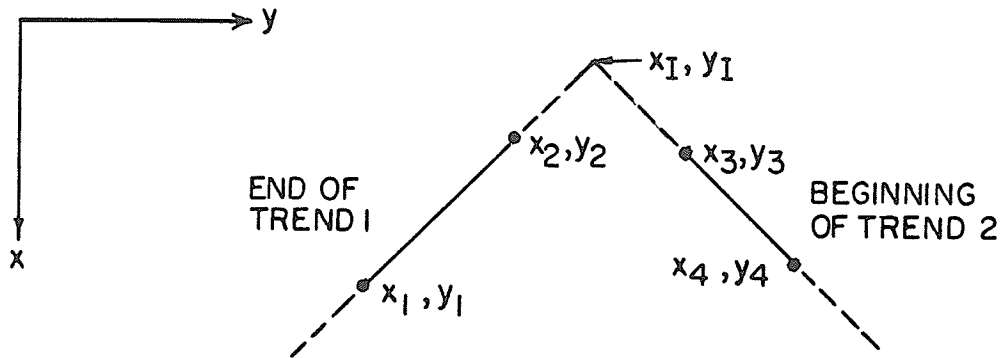


FIGURE 13: Intersection of Two Lines

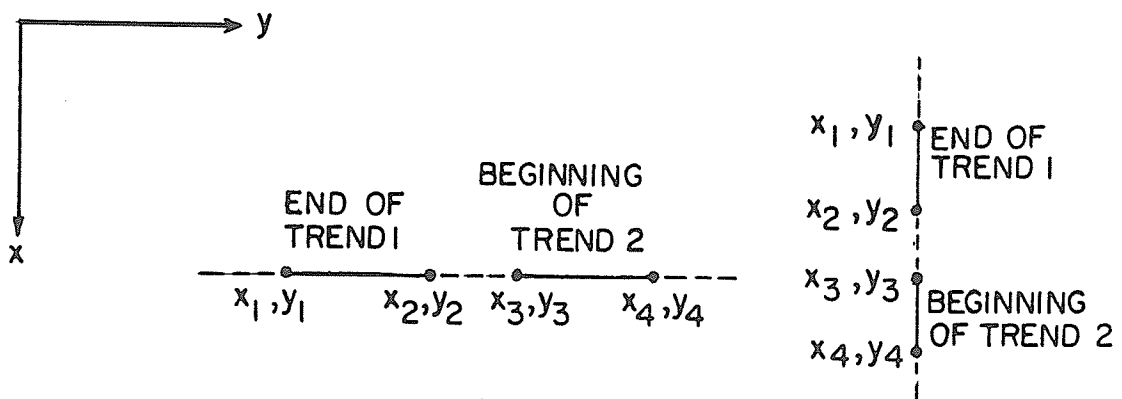


FIGURE 14: Points on Same Line

intersection, such as the corner illustrated in Figure 13, the coordinates of their intersection can be determined by solving the equations for the two lines simultaneously. The four points identifying each transition area are assumed to define two separate lines that have an intersection point (XI,YI). The method to calculate the intersection point can be derived as follows:

Lines A-B and C-D are represented by the following equations:

$$x_1 = m_1 y_1 + b_1 \quad (3-14) \qquad x_3 = m_2 y_3 + b_2 \quad (3-16)$$

$$x_2 = m_1 y_2 + b_1 \quad (3-15) \qquad x_4 = m_2 y_4 + b_2 \quad (3-17)$$

Solving these equations simultaneously, one can obtain:

$$m_1 = (x_1 - x_2) / (y_1 - y_2); \quad (3-18)$$

$$m_2 = (x_3 - x_4) / (y_3 - y_4); \quad (3-19)$$

$$\begin{aligned} b_1 &= x_1 - m_1 y_1 \\ &= (x_2 y_1 - x_1 y_2) / (y_1 - y_2); \text{ and} \end{aligned} \quad (3-20)$$

$$\begin{aligned} b_2 &= x_3 - m_2 y_3 \\ &= (x_4 y_3 - x_3 y_4) / (y_3 - y_4). \end{aligned} \quad (3-21)$$

Similarly, the intersection of the two lines (XI,YI) is obtained by:

$$XI = m_1 YI + b_1 ; \quad (3-22)$$

$$XI = m_2 YI + b_1 ; \text{ and} \quad (3-23)$$

$$YI = \frac{b_2 - b_1}{m_1 - m_2} . \quad (3-24)$$

Therefore:

$$XI = (m_1 b_2 - m_2 b_1) / (m_1 - m_2) ; \quad \text{and} \quad (3-25)$$

$$YI = \frac{b_2 - b_1}{m_1 - m_2} . \quad (3-26)$$

Substituting and simplifying equations (3-25) and (3-26):

$$YI = \frac{(x_4 y_3 - x_3 y_4)(y_1 - y_2) - (x_2 y_1 - x_1 y_2)(y_3 - y_4)}{(x_1 - x_2)(y_3 - y_4) - (x_3 - x_4)(y_1 - y_2)} \quad (3-27)$$

$$XI = \frac{(x_4 y_3 - x_3 y_4)(x_1 - x_2) - (x_2 y_1 - x_1 y_2)(x_3 - x_4)}{(x_1 - x_2)(y_3 - y_4) - (x_3 - x_4)(y_1 - y_2)} \quad (3-28)$$

As can be seen from the equations, the denominators equal zero when the lines are parallel or when all four points are on the same line. This ideally occurs at the midpoint of the sides of a polygon (minimal radii) as shown in Figure 14. In this case the two middle points are simply averaged to define the transition point as follows:

$$XI = \frac{x_2 + x_3}{2} , \text{ and } YI = \frac{y_2 + y_3}{2} \quad (3-29)$$

Special consideration is given to these areas in the computer program to eliminate any "division by zero" errors.

To determine which equation to use, the array TREND must

first be examined. A maximum is found when a transition area goes from $TREND(n)=1$ to $TREND(n)=-1$. Conversely, a minimum is found when a transition area goes from -1 to 1 . Therefore, by examining the $TREND$ array for the second and third points of each transition area, it may be determined whether a transition point is a maximum or not. These maxima are separated from the rest and placed in arrays **CORNR** and **CORNC** (row and column respectively).

Once all the maxima have been found (assuming no false corners exist) the object is routed to the **TRIANGLE** or **QUADRILATERAL** test sections according to the number of true maxima. If the number of maxima, or corners, is anything other than 3 or 4, the object is rejected as "unknown". Future enhancements could add other tests at this point.

3.5.4 QUADRILATERAL Test Section

In the quadrilateral test section, the lengths of sides are first calculated, and are then compared for equality. The half-diagonals, which are the distances from the geometric center to each corner, are also calculated and compared for equality. If all four sides and all four half-diagonals are within a certain tolerance defined as "**SIDETOL**", then the object is recognized as a **SQUARE**. If the sides are not within **SIDETOL**, they are not all equal, and the opposite pairs of sides and opposite pairs of half-diagonals are then compared for equality. If they are both

within SIDETOL of each other, the object is recognized as being a **RECTANGLE**. If neither comparison is proven, the object is rejected as an unknown quadrilateral. Further expansion of this test section could include additional objects such as a parallelogram or rhombus. The optimum value of SIDETOL is discussed in Chapter IV.

The orientation of a quadrilateral is defined in this study as the smallest, clockwise, positive angle of any side with respect to the positive y-axis as shown in Figure 15. The calculation of this angle takes advantage of the fact that for any regular quadrilateral, one need only calculate the four-quadrant angles of two of any two adjacent sides, and then add 180 degrees to each to determine the directed angles of the other two sides. The possible angles are limited to the range 0 to 360 degrees. The four angles are then compared to determine the smallest positive angle, **MINPOS**, which is designated as the angle of orientation of the object.

3.5.5 TRIANGLE Test Section

In the triangle test section, the lengths of the sides are calculated and compared for equality. If all three sides are within tolerance SIDETOL, the object is found to be an **EQUILATERAL TRIANGLE**. If they are not equal, pairs of sides are compared for equality. If two sides are found to be within tolerance SIDETOL, the object is an **ISOSCELES**

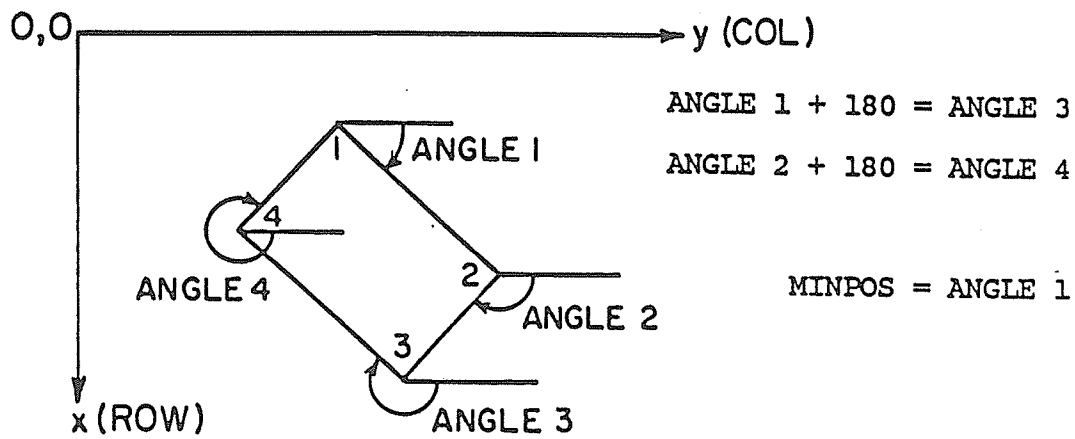


FIGURE 15: Orientation Angle for a Quadrilateral

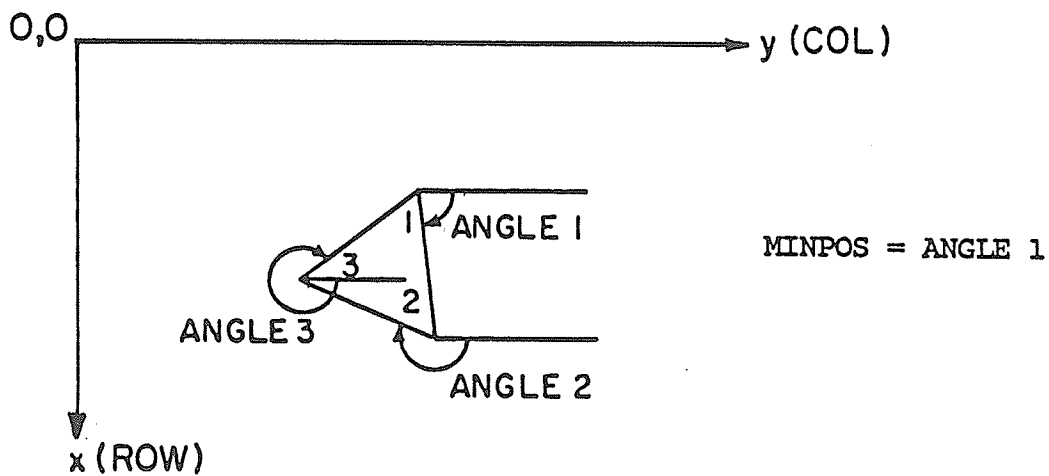


FIGURE 16: Orientation Angle for a Triangle

TRIANGLE. Any other type of triangle is rejected as scalene. The angle of orientation for a triangle is defined in the same way as that for the quadrilaterals. This is shown in Figure 16. However all three angles are calculated separately and then compared to determine the minimum positive angle, MINPOS.

3.5.6 CIRCLE Test Section

In this section, the difference between the maximum and minimum radii, **DIFFER**, is compared to **SIDETOL**. If **DIFFER** is within **SIDETOL**, the object is recognized as a **CIRCLE**, otherwise it is rejected as an unknown object.

3.6 Summary

There are some variables whose optimum value can only be determined by considering the interaction between the camera, the scene, the frame grabber board and the associated software. A number of data has to be collected and analyzed to arrive at the qualitative and quantitative aspects of the objects' features. The variables that are influenced by this interaction are:

- 1) **NUMAVE**: optimum number of consecutive points on the edge of an object whose average row and column coordinates are used to determine radial distances.

- 2) **RADITOL:** the percentage tolerance level used to compare adjacent radii, and thus determine increasing and decreasing information.
- 3) **CIRCL:** the percentage of zero's in the objects' TRANS array above which indicates the presence of a circle and the absence of a simple polygon.
- 4) **STLEN:** the number of consecutive points on an edge whose TRANS values must be identical 1's or -1's to determine the presence of a trend.
- 5) **SIDETOL:** the percentage tolerance level associated with the equality of sides or "half-diagonals" of a polygon.

The optimum values for these variables were determined from analysis of actual image data obtained from the Frame Grabber. The various tests used to obtain the data as well as a discussion of the analysis is presented in the next chapter.

CHAPTER IV: EXPERIMENTAL RESULTS

A number of tests were conducted to verify the software based algorithms and to find the optimum values for the variables used therein. The raw data for all these tests are presented in Appendix A, and appropriate extracts from these data are then used throughout this chapter. The objects used in the tests were cut out of black cardboard and then mounted on a white background to provide a sharp contrast. They were presented in the camera's field of view, over a range of 7" to 24" away from the lens. Only the ambient light of the room was used to illuminate the scene. However, because of the sharp contrast between the object and the background, the lighting was found to be adequate for this study.

This chapter discusses the optimum values found for the various variables.

4.1 Optimum Value of NUMAVE

NUMAVE is the number of adjacent points on an edge. The average coordinate of these points is used to calculate the distance to the geometric center. It is always an odd, whole number that directly associates the distance computed to an actual point on the edge without any geometrical bias.

Qualitatively, it may be seen that, if too few points are

used for the average, the results are sensitive to the digitization and transmission noises. On the other hand, as more and more points are used in obtaining the average coordinate value, the corners become increasingly rounded. The resulting corner coordinates extracted from this data will tend to produce shorter and shorter side lengths.

These generalizations are evident when examining the numerical results. For the polygons used, when NUMAVE equals one, the results yield erroneous corner locations, resulting in unidentifiable objects. This can also be seen by examining the data for the circle. Table 4 indicates that by using NUMAVE values of one or three, corners were erroneously located resulting in incorrect identification of the circles. Thus the optimum value for NUMAVE must be at least five.

Figure 17 illustrates a square whose corner coordinates are plotted for NUMAVE values of seven and twenty-one. These data are taken from DATA A4 part 3 in Appendix A. The corners in these results are compared to corners extracted from the raw edge coordinates. A comparison of the values in Table 5 indicates the change in the length of the sides as NUMAVE increases. The largest difference in side lengths from the actual for NUMAVE equal to seven is 2.35 pixels (0.91%), and for NUMAVE equal to twenty-one it is 6.94 pixels (2.7%).

NUMAVE	I.D.	RADII		DIFFERENCE MAX - MIN
		MAXIMUM	MINIMUM	
1	-	-	-	-
3	-	-	-	-
5	CIRCLE	97.415	100.685	3.270
7	CIRCLE	97.424	100.658	3.234
9	CIRCLE	97.435	100.615	3.180
11	CIRCLE	97.435	100.574	3.139
13	CIRCLE	97.442	100.547	3.105
15	CIRCLE	97.441	100.510	3.069
17	CIRCLE	97.446	100.463	3.017
19	CIRCLE	97.427	100.413	2.986
21	CIRCLE	97.411	100.364	2.953
49	CIRCLE	96.695	99.201	2.506
69	CIRCLE	95.368	98.101	2.733
89	CIRCLE	96.692	96.626	2.934
AVE:		97.431	100.537	3.106
SPREAD:		0.04%	0.32%	10.21%

COMMENTS:

- 1) All averages and spreads done from NUMAVE = 5 to 21 only.
- 2) All dashes (-) indicate the erroneous location of corners.

TABLE 4: Effects of Increasing NUMAVE on a Circle

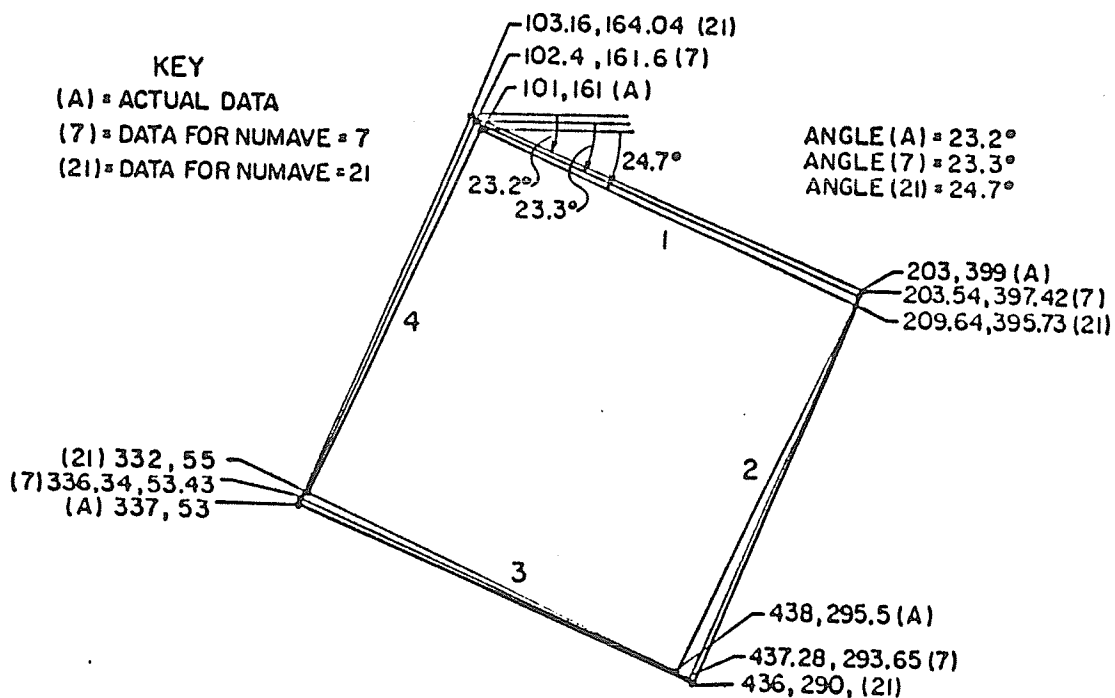


FIGURE 17: Corner Location with Increasing NUMAVE

TABLE 5: Corner Location Data with Increasing NUMAVE

	Length of Sides (pixels)				Angle of Orientation (degrees)
	1	2	3	4	
ACTUAL	258.94	256.78	262.69	259.54	23.2
NUMAVE=7	256.59	255.74	260.57	257.74	23.21
NUMAVE=21	254.99	249.84	256.98	253.49	24.68
ACTUAL - NUMAVE=7	2.35	1.04	2.12	1.8	0.01
(%)	0.91%	0.41%	0.81%	0.69%	0.04%
ACTUAL - NUMAVE=21	3.95	6.94	5.71	6.05	1.48
(%)	1.32%	2.7%	2.17%	2.33%	6.38%

NOTE: ACTUAL data is taken from RAW data.

Further examination indicates a clockwise shift of 1.47 degrees in the orientation angle when moving from NUMAVE values of seven to twenty-one. The difference in angle between the true and calculated angle for a NUMAVE value of seven is 0.01 degrees (0.04%). The difference in the angles for a NUMAVE value of twenty-one is 1.48 degrees (6.4%), and hence the NUMAVE value of twenty-one is too large.

From scanning the data for intermediate NUMAVE values, there seems to be nothing gained from using large values of NUMAVE as opposed to smaller values of, say, seven or nine. The speed of processing using large values will also be reduced.

These results stem from data taken at a fixed distance from the camera lens. Similar tests were performed to investigate the effects of increasing NUMAVE at distances of approximately 7", 14", 20" and 24" from the front of the camera. It was found that the gain by using larger values of NUMAVE was negligible and also the speed of processing will be considerably reduced. Hence attention was given to the lower possible NUMAVE values, namely five, seven and nine, at the different distances.

Table 6 shows the maximum percentage difference between equal sides (or radii) of each shape, for NUMAVE values of five, seven and nine over the given range of distances. It was found that in general, the maximum percentage

OBJECT	NUMAVE	DISTANCES FROM OBJECT TO CAMERA (APPROXIMATE)			
		7"	14"	20"	24"
SQUARE	5	4.7 %	4.0 %	4.11 %	2.0 %
	7	4.14	4.14	4.22	2.5
	9	4.2	4.2	4.22	2.5
RECTANGLE	5	3.1	2.6	1.4	1.3
	7	2.3	2.3	1.5	2.8
	9	2.6	3.5	1.2	3.2
ISOSCELES TRIANGLE	5	0.8	0.4	0.25	0.2
	7	1.9	1.8	0.3	0.11
	9	1.9	1.9	0.4	0.03
EQUILATERAL TRIANGLE	5	1.4	1.0	1.4	1.0
	7	1.9	1.1	1.3	1.5
	9	2.11	1.0	1.3	1.5
CIRCLE	5	5.43	5.5	5.0	4.8
	7	6.08	5.2	4.8	4.8
	9	5.91	4.8	4.7	4.6
MAXIMUM	5	5.43	5.5	5.0	4.8
	7	6.08	5.2	4.8	4.8
	9	5.91	4.8	4.7	4.6

NOTE: Differences are between the longest and shortest sides or radii, and are expressed as a percentage of the shortest side or radii.

TABLE 6: Maximum Percentage Difference of
Sides / Radius

difference occurred at the shorter distance of 7". It was evident that the circle yielded the highest difference, and it was found that a significantly larger number of changes in adjacent radii were present at the shorter distances. Therefore the value of variables CIRCL and SIDETOL could not be kept constant in order to obtain these results for constant NUMAVE. These effects are discussed later.

Thus, the optimum value of NUMAVE was chosen to be seven. This value produced the least amount of distortion of the various characteristics, and also kept the processing speed close to a minimum.

4.2 Optimum Value of STLEN

A trend is either increasing towards a corner or decreasing towards a midpoint of a side. At a corner, for example, the trend will increase and then decrease. This can be easily seen by considering the distances from the geometric center to the corner point. The length decreases as one moves away from the corner toward the midpoint of the side. Each increasing and decreasing trend must be accurately represented in order to reach the objective of pinpointing a corner coordinate. The TRANS array is such an indicator. A TRANS array value is either a 1, -1, or 0, to represent increasing, decreasing, or constant distance lengths of adjacent points on the edge to the geometric center. Thus, chains of 1's would represent an increasing

trend, and so on.

The number of adjacent points included in each trend chain is **STLEN**. It is obvious that a trend could not be accurately designated using only one or two points as indicators. They could be subject to noise or digitization error and would not necessarily reveal the true nature of the trend present. It is conceivable that pairs or trios of points of the opposite trend could be imbedded in a trend, and therefore produce inaccurate results. Since the endpoints of two chains are used to linearly extrapolate the corner coordinates, they must contain enough points to produce accurate results. Therefore, the problem focuses on the choice of a value for **STLEN** that would indicate the type of trend present, and would predict corner coordinates accurately.

Qualitatively, if very large values of **STLEN** were used, the trend would be identified correctly, but the endpoints would be so far away from each other that a line between them would not necessarily be a valid one for all points contained between them. A value of **STLEN**, therefore, must be chosen that is small enough to balance this problem, without allowing the chain of points to be subjected to a significant amount of noise or digitization error.

The data presented in Tables A3, B3, C3 and D3 of Appendix A show the effects of increasing **STLEN** values from one to twenty. Incorrect identifications of the objects

occurred frequently when STLEN was less than four. These data also indicate that there is little or no change found in corner coordinates, with all shifts being typically under 0.75 pixels. There is little change in the angle of orientation which was a maximum change of 0.112 degrees or 2.66%. The lengths of the sides only change by a maximum of 0.68 pixels. This accuracy would certainly seem to be good enough for all practical purposes.

These data indicate that no extra accuracy is gained from using higher values of STLEN. These also increase the processing time required for the calculations. The same results were found not only for the fixed distances from the camera from which the data was taken, but also for the range of distances of approximately 7" to 24". Thus the value of STLEN was more dependent on the desired accuracy of the corner coordinates extrapolated using its endpoints. The optimum value for STLEN was chosen to be eight to take into account all these factors.

4.3 Optimum Value of RADITOL

RADITOL is the tolerance, expressed as a percentage, to which the radii of an object, located in the **DSTNC** array, are compared for equality. Clearly the size of the objects used is an important factor when determining this variable. The smaller the object, the more significant the differences between adjacent radii will be. If more pixels are used to

represent the edge of an object, the less significant these differences become. Therefore, the value of RADITOL is expressed as a percentage of the radius rather than as a specific number of pixels. This enables the program to identify the objects consistently, independent of the size of the object.

Table 7 shows the effect of different values of RADITOL on a circle. Values under 0.2% yielded "object unidentifiable" results. Thus, a minimum value of 0.2% was required. As RADITOL increased, the number of zero's significantly jumped until all (100%) of the radii were deemed as equal at a RADITOL value of 0.35%.

Since the objects used are regular shapes, the critical areas, or areas where the least amount of radial change occurs, will be those around the midpoint of a side or near the corner of a polygon, or between adjacent radii of a circle. Table 8 lists for three groups of points, the radii of a circle (**DISTANCE**), their adjacent radial differences (**DIFF. A**), and their corresponding percentage difference (**DIFF. B**). Tables 9 and 10 show the same information for the midpoint of a side and the corner of a square. Similar data were gathered for all the shapes used, and it would found that using a RADITOL value of 0.2% yielded accurate distinction between adjacent radii for both the circle and polygons. The TRANS array values for this RADITOL value are given in the tables under the TRANS

RADITOL (%)	I.D.	# ZERO'S (CNTZERO)	# EDGE POINTS	%AGE ZEROS (%)	RADII		DIFF. MAX-MIN (PIXEL)
					MAX	MIN (PIXELS)	
* 0.1	-	-	-	-	-	-	-
* 0.15	-	-	-	-	-	-	-
0.2	CIR.	552	561	98.4	97.4	100.7	3.3
0.25	CIR.	560	561	99.8	97.4	100.7	3.3
0.3	CIR.	561	561	100.0	97.4	100.7	3.3
0.35	CIR.	561	561	100.0	97.4	100.7	3.3
0.4	CIR.	561	561	100.0	97.4	100.7	3.3

* NOTE: Dashed lines had incorrect identifications.

TABLE 7: Effects of Increasing RADITOL on a Circle

EDGE PT. #	COORDS. ROW COL	DISTANCE TO XG,YG	DIFF. OF DIST.		TRANS #
			A	B	

362	154 170	87.16639	-	-	0
363	153 170	87.13692	0.00034	0.034%	0
364	152 170	87.11893	0.00021	0.021%	0
365	151 170	87.3124	0.00222	0.222%	1
366	150 170	87.51734	0.00234	0.234%	1
367	149 169	87.73367	0.00247	0.247%	1
368	148 169	87.96127	0.00259	0.259%	1
369	147 169	88.20007	0.00271	0.271%	1
370	146 169	88.25031	0.00057	0.057%	0
371	145 169	88.31184	0.00070	0.070%	0
372	144 169	88.38464	0.00082	0.082%	0

417	99 187	87.57246	-	-	0
418	98 187	87.53069	0.00048	0.048%	0
419	97 188	87.50764	0.00026	0.026%	0
420	96 189	87.3779	0.00148	0.148%	0
421	95 190	87.10846	0.00309	0.309%	-1
422	95 191	86.85707	0.00289	0.289%	-1
423	94 192	86.62388	0.00269	0.269%	-1
424	93 193	86.55754	0.00077	0.077%	0
425	92 194	86.64186	0.00097	0.097%	0
426	91 194	86.74502	0.00119	0.119%	0
427	90 195	86.72675	0.00021	0.021%	0

41	78 301	84.94751	-	-	0
42	79 302	84.95844	0.00013	0.013%	0
43	79 303	84.98537	0.00032	0.032%	0
44	80 304	85.02829	0.00050	0.050%	0
45	81 305	85.08717	0.00069	0.069%	0
46	81 306	85.16199	0.00088	0.088%	0
47	82 307	85.25268	0.00106	0.106%	0

COLUMN DESCRIPTION FOR TABLES 8, 9, AND 10.

EDGE PT. #: Edge point number in list.
ROW,COL: Coordinates of edge point (true data).
DISTANCE: Radius from geometric center (XG,YG) to
the average coordinate (NUMAVE=5).
DIFF of DIST. A: DISTANCE[i]-DISTANCE[i-1]=DIFF[i]
DIFF of DIST. B: DIFF[i]/DISTANCE[i] AS A PERCENTAGE.
TRANS: TRANS array value of edge point based on
RADITOL = 0.2%.

TABLE 8: Radii Length Information - Circle Parts

CORNER DATA - SQUARE						
EDGE	COORDS.		DISTANCE	TRANS	DIFF OF DIST.	
PT.#	ROW	COL	TO XG,YG	#	A	B
275	376	388	172.1880	1		
276	377	388	172.8254	1	0.6374	0.369%
277	378	388	173.4662	1	0.6408	0.369%
278	379	388	174.1104	1	0.6442	0.370%
279	380	388	174.7580	1	0.6476	0.371%
280	381	388	175.4088	1	0.6508	0.371%
281	382	388	176.0630	1	0.6542	0.372%
282	383	388	176.7203	1	0.6573	0.372%
283	384	388	177.3809	1	0.6606	0.372%
284	385	388	178.0446	1	0.6637	0.373%
285	386	388	178.7115	1	0.6669	0.373%
286	387	388	179.3815	1	0.67	0.374%
287	388	388	180.0545	1	0.673	0.374%
288	389	388	180.7305	1	0.676	0.374%
289	390	388	181.2631	1	0.5326	0.294%
290	391	388	181.5171	0	0.254	0.140%
291	392	387	181.4924	0	0.0247	0.014%
292	392	386	181.1889	0	0.3035	0.168%
293	392	385	180.6065	-1	0.5824	0.322%
294	392	384	179.8885	-1	0.718	0.399%
295	392	383	179.1731	-1	0.7154	0.399%
296	392	382	178.4605	-1	0.7126	0.399%
297	392	381	177.7507	-1	0.7098	0.399%
298	392	380	177.0437	-1	0.707	0.399%
299	392	379	176.3395	-1	0.7042	0.399%
300	392	378	175.6382	-1	0.7013	0.399%
301	392	377	174.9398	-1	0.6984	0.399%
302	392	376	174.2443	-1	0.6955	0.399%
303	392	375	173.5518	-1	0.6925	0.399%
304	392	374	172.8624	-1	0.6894	0.399%
305	392	373	172.1759	-1	0.6865	0.399%
306	392	372	171.4926	-1	0.6833	0.398%
307	392	371	170.8123	-1	0.6803	0.398%
MAX:			181.5171		0.7180	
MIN:			170.8123		0.0247	

CORNER COORD. (DATA ABOVE): 392,388

TABLE 9: Radii Length Information - Square Corner

MIDPOINT OF SIDE DATA - SQUARE															
EDGE PT. #	COORDS.		DISTANCE TO XG,YG	TRANS #	DIFF OF DIST.		EDGE PT. #	COORDS.		DISTANCE TO XG,YG	TRANS #	DIFF OF DIST.			
	ROW	COL			A	B		ROW	COL			A	B		
115	216	385	139.4092	-1			165	266	386	130.9243	0				
116	217	385	139.0498	-1	0.3594	0.258%	166	267	386	130.9240	0	0.0003	0.000%		
117	218	385	138.6966	-1	0.3532	0.255%	167	268	386	130.9312	0	0.0072	0.005%		
118	219	385	138.3498	-1	0.3468	0.251%	168	269	386	130.9462	0	0.015	0.011%		
119	220	385	138.0093	-1	0.3405	0.247%	169	270	386	130.9687	0	0.0225	0.017%		
120	221	385	137.6753	-1	0.334	0.243%	170	271	386	130.9989	0	0.0302	0.023%		
121	222	385	137.3477	-1	0.3276	0.239%	171	272	386	131.0367	0	0.0378	0.029%		
122	223	385	137.0266	-1	0.3211	0.234%	172	273	386	131.0821	0	0.0454	0.035%		
123	224	385	136.7121	-1	0.3145	0.230%	173	274	386	131.1352	0	0.0531	0.040%		
124	225	385	136.4042	-1	0.3079	0.226%	174	275	386	131.1958	0	0.0606	0.046%		
125	226	385	136.1030	-1	0.3012	0.221%	175	276	386	131.2641	0	0.0683	0.052%		
126	227	385	135.8084	-1	0.2946	0.217%	176	277	386	131.3399	0	0.0758	0.058%		
127	228	385	135.5206	-1	0.2878	0.212%	177	278	386	131.4232	0	0.0833	0.063%		
128	229	385	135.2396	-1	0.281	0.208%	178	279	386	131.5142	0	0.091	0.069%		
129	230	385	134.9653	-1	0.2743	0.203%	179	280	386	131.6126	0	0.0984	0.075%		
130	231	385	134.6980	0	0.2673	0.198%	180	281	386	131.7186	0	0.106	0.080%		
131	232	385	134.4376	0	0.2604	0.194%	181	282	386	131.8321	0	0.1135	0.086%		
132	233	385	134.1841	0	0.2535	0.189%	182	283	386	131.9530	0	0.1209	0.092%		
133	234	385	133.9376	0	0.2465	0.184%	183	284	386	132.0815	0	0.1285	0.097%		
134	235	385	133.6981	0	0.2395	0.179%	184	285	386	132.2173	0	0.1358	0.103%		
135	236	385	133.4657	0	0.2324	0.174%	185	286	386	132.3606	0	0.1433	0.108%		
136	237	385	133.2404	0	0.2253	0.169%	186	287	386	132.5113	0	0.1507	0.114%		
137	238	385	133.0222	0	0.2182	0.164%	187	288	386	132.6693	0	0.158	0.119%		
138	239	385	132.8112	0	0.211	0.159%	188	289	386	132.8347	0	0.1654	0.125%		
139	240	385	132.6034	0	0.0078	0.006%	189	290	386	133.0074	0	0.1727	0.130%		
140	241	385	132.4033	0	0.0001	0.000%	190	291	386	133.1873	0	0.1799	0.135%		
141	242	386	132.2112	0	0.0079	0.006%	191	292	386	133.3746	0	0.1873	0.140%		
142	243	386	132.0268	0	0.0156	0.012%	192	293	386	133.5690	0	0.1944	0.146%		
143	244	386	132.8503	0	0.0235	0.018%	193	294	386	133.7707	0	0.2017	0.151%		
144	245	386	132.6842	0	0.1661	0.125%	194	295	386	133.9795	0	0.2088	0.156%		
145	246	386	132.5255	0	0.1587	0.120%	195	296	386	134.1954	0	0.2159	0.161%		
146	247	386	132.3742	0	0.1513	0.114%	196	297	386	134.4184	0	0.223	0.166%		
147	248	386	132.2302	0	0.144	0.109%	197	298	386	134.6485	0	0.2301	0.171%		
148	249	386	132.0937	0	0.1365	0.103%	198	299	386	134.8856	0	0.2371	0.176%		
149	250	386	131.9646	0	0.1291	0.098%	199	300	386	135.1297	0	0.2441	0.181%		
150	251	386	131.8429	0	0.1217	0.092%	200	301	386	135.5742	1	0.4445	0.328%		
151	252	386	131.7287	0	0.1142	0.087%	201	302	386	135.8318	1	0.2576	0.190%		
152	253	386	131.6221	0	0.1066	0.081%	202	303	387	136.0962	1	0.2644	0.194%		
153	254	386	131.5229	0	0.0992	0.075%	203	304	386	136.3674	1	0.2712	0.199%		
154	255	386	131.4313	0	0.0916	0.070%	204	305	386	136.6455	1	0.2781	0.204%		
155	256	386	131.3472	0	0.0841	0.064%	205	306	386	136.7387	1	0.0932	0.068%		
156	257	386	131.2707	0	0.0765	0.058%	206	307	386	137.0306	1	0.2919	0.213%		
157	258	386	131.2018	0	0.0689	0.053%	207	308	386	137.3292	1	0.2986	0.217%		
158	259	386	131.1405	0	0.0613	0.047%	208	309	386	137.8246	1	0.4954	0.359%		
159	260	386	131.0867	0	0.0538	0.041%	209	310	386	138.3257	1	0.5011	0.362%		
160	261	386	131.0406	0	0.0461	0.035%	210	311	387	138.8326	1	0.5069	0.365%		
161	262	386	131.0021	0	0.0385	0.029%	211	312	387	139.1560	1	0.3234	0.232%		
162	263	386	130.975	0	0.0271	0.021%	212	313	387	139.5841	1	0.4281	0.312%		
163	264	386	130.9479	0	0.0542	0.041%	213	314	386	140.0102	1	0.0542	0.040%		
164	265	386	130.9323	0	0.0156	0.012%	214	315	387	140.3523	1	0.3421	0.244%		
MAX:			139.4092			0.3594						140.3523			0.8542
MIN:			130.9323			0.0001						130.924			0.0003

MIDPOINT OF SIDE: 265.5 , 385.5

TABLE 10: Radii Length Information - Square Side Midpoint

column in Figures 8, 9 and 10.

4.4 Optimum Value of CIRCL

The change in radii of a circle is ideally 0. However, detectable differences between radii can be created when using a digitized image of the circle. Hence the TRANS array, whose values show the number of different or constant radii, will conceivably contain -1 and/or +1 values, but the majority of this array will be 0's. Therefore a percentage value, represented by "CIRCL", must be chosen to differentiate between a very noisy polygon and a circular object.

A number of tests were performed on the circle and the other polygons to determine the minimum percentage of 0's present in each shape. The results for a fixed distance from the camera are listed in Table 11. For the circle, this number ranged from 510 to 552 in 561 circumference points which correspond to the range 90% to 98%. The polygon tests revealed that the majority of 0's were located around the midpoint of a side. The number of zero's present in these shapes ranged from 56 out of a total of 715 perimeter points of an equilateral triangle (7.8%), to 301 in 1034 perimeter points in a square (29.1%).

Table 12 shows the number of zero's present in the TRANS array for different distances from the camera ranging from

OBJECT		# ZEROS	# EDGE POINTS	%AGE ZEROS
SQUARE	1	301	1034	29.11%
	2	298	1033	28.85
	3	251	953	26.34
ISOS. TRIANGLE	1	74	775	9.55
	2	78	782	9.97
	3	68	757	8.98
EQUIL. TRIANGLE	1	56	715	7.83
	2	70	714	9.80
	3	65	710	9.15
RECTANGLE	1	159	752	21.14
CIRCLE	1	552	561	98.40

TABLE 11: Typical Totals of Zero's in Objects

OBJECT	NUMAVE (PIXELS)	DISTANCE FROM CAMERA (APPROX.)			
		7"	14"	20"	24"
SQUARE	5	31.7	20.1	15.0	11.5
	7	31.2	20.5	15.2	10.4
	9	31.3	20.8	16.7	11.2
RECTANGLE	5	21.9	14.5	12.4	11.0
	7	25.4	17.0	12.5	10.4
	9	25.9	17.1	12.5	10.4
ISOSCELES TRIANGLE	5	9.8	7.1	4.3	4.9
	7	11.3	7.6	4.9	5.2
	9	11.6	7.6	5.4	5.2
EQUIL. TRIANGLE	5	10.2	6.5	5.0	5.3
	7	10.2	6.1	5.8	4.6
	9	10.4	6.7	5.8	3.9
MAXIMUM MINIMUM		31.7	20.8	16.7	11.5
		9.8	6.1	4.3	3.9
CIRCLE	5	95.5	75.4	69.0	72.0
	7	96.5	85.3	72.3	69.0
	9	98.4	90.7	82.3	72.0

TABLE 12: Typical Percentage of Zero's Over Ranges of Distances from Camera

7" to 24". It was found that the maximum number of zero's was 31.7% for a square located approximately 7" from the camera. The minimum number of zero's was 3.9% for an equilateral triangle located at approximately 24". The circle tests revealed that a maximum number of zero's (98.4%) was present at approximately 7" and a minimum percentage of 69% was found at both 20" and 24".

Thus the optimum value for CIRCL was largely based on the variance of the circle's radii. A value of 65% was arbitrarily chosen to give a broader tolerance for detection of circular objects, while clearly rejecting other shapes over the range of distances.

4.5 Optimum Value of SIDETOL

Once the corners of a polygon have been determined, the program compares lengths of sides and lengths of half-diagonals to determine the type of object present in the image. **SIDETOL** is the tolerance, expressed as a percentage, within which these lengths are said to be equal. The difference between maximum and minimum radii of a circle is also compared to SIDETOL for final confirmation of the object being a circle as opposed to being elliptical or oval in shape.

This variable is very much dependent upon the amount of image area taken up by the object. If the object occupies much of the 512 x 512 image grid, the sides will be longer

and will be less sensitive to quantization error and noise, than an object using less image space. In addition, SIDETOL cannot be too large otherwise sides that are not supposed to be equal could be deemed to be as such. Consequently, the smallest optimum value must be determined.

The data in Table 13, which is summarized from Table 6, indicates the maximum percentage difference between equal sides. This information indicates that to distinguish all the objects from one another the minimum percentage difference between the longest equal side and the unequal side of the isosceles triangle being tested was 9.5%, this is denoted as the maximum value of SIDETOL.

It is still conceivable that some of the data may be corrupted somewhat due to noise or digitization error. Thus, an arbitrary value of 6.5% was chosen to differentiate between equal and unequal lengths. While this percentage seems high, it was found that anything lower resulted in incorrect identification of some of the triangles. On the other hand, this number has no significant effect on computing time.

4.6: Summary

The optimum values of the variables of interest are:

NUMAVE: 7 pixels

RADITOL: 0.2%

OBJECT	DISTANCE FROM CAMERA (APPROX.)			
	7"	14"	20"	24"
SQUARE	4.7%	4.7%	4.22%	2.5%
RECTANGLE	3.1	3.5	1.5	3.2
EQUIL. TRIANGLE	2.11	1.1	1.35	1.49
ISOS. TRIANGLE	1.9 (10.1)	1.93 (10.4)	0.31 (9.5)	0.2 (9.5)
CIRCLE	6.08	5.5	5.0	4.8

NOTE:

- 1) All values are maximum percentages based on the maximum difference between equal sides.
- 2) Those values in parenthesis () indicate the maximum percentage between the maximum equal side and the unequal side.
- 3) Values for the circle are percentage differences between the maximum and minimum radii.

TABLE 13: Differences Between Equal Sides

CIRCL: 65%
STLEN: 8 pixels
SIDETOL: 6.5%

These values were found to yield accurate results over a range of distances from the camera from approximately 7" to 24".

It was also found that the optimum value of one variable was dependent upon some of the other variables in order to get consistent and accurate results over the range of distances. Thus an optimum combination of variables NUMAVE, CIRCL and SIDETOL was required. The optimum values of variables RADITOL and STLEN were not affected by the change in distances.

CHAPTER V: CONCLUSIONS AND RECOMMENDATIONS

The objective of this thesis was to describe and demonstrate the necessary steps to effect object recognition using a vision camera through the development of a fast, simple method of identifying certain objects. As with any "simple" method, there are limitations. Indeed the very word implies a trade-off of such limitations in order to achieve the stated objective(s). The computer algorithms developed to demonstrate the object recognition tasks had to incorporate many trade-offs, and there were certainly many ways to expand the program to accommodate more complex problems.

This chapter discusses the known limitations of the software implementation of the algorithms. It also suggests possible ways to improve the program such as through the extension of the routines for the enhancement of speed and accuracy. A possible application in the area of robotics is also discussed in this chapter.

5.1 Limitations of the Software Routines

The most obvious limitation of the program lies in the type of objects it can identify. This stems from the method used, itself. The radii are measured from the geometric center to an averaged point on the edge, and each difference is compared against the tolerance, **RADITOL**, to

determine whether the edge is increasing towards a corner or decreasing away from one. If the object's shape is irregular or non-symmetrical, a corner may not necessarily be a maximum. This is illustrated for a quadrilateral in part A of Figure 18. This is also the case when corners are cut off from the regular shaped objects, such as the isosceles triangle in part B of Figure 18. These figures were tested for rejection. Although they were rejected as unidentifiable objects, the characteristics of the shapes revealed incorrect corner coordinates due mostly to complete omission of one of the cut off corners.

In addition, it is not known as to how many sides the regular shaped objects can have before the optimum RADITOL value should be reduced. The more sides the object has, the more the object could resemble a circle. This will result in a greater number of zero values in the TRANS array. This question can only be answered through further testing. Thus, the program, as it stands, is critically dependent upon the symmetrical qualities of the objects.

Another limitation is evident in the radial comparison technique used. Adjacent radii 1 and 2 are compared against RADITOL for equality. If they are within this tolerance, there is no significant change registered. The next comparison, between radii 2 and 3, may also be within the tolerance, but it is conceivable that radii 3 and 1 may differ by more than the tolerance. This is the case, for

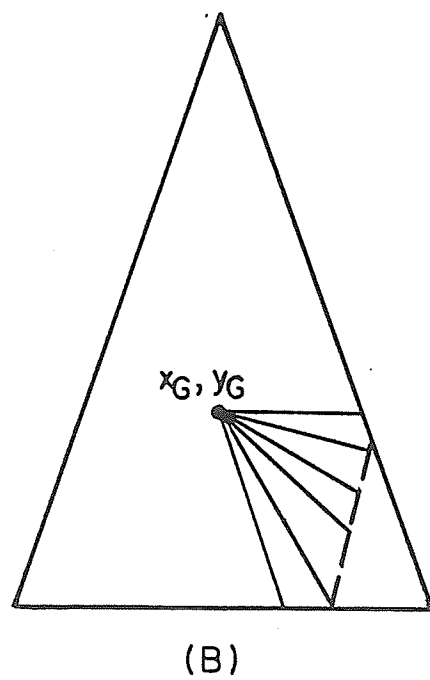
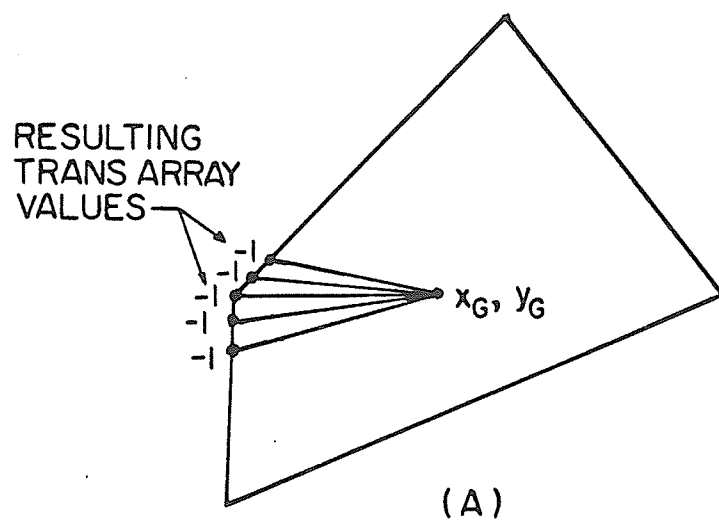


FIGURE 18: Abnormal Objects

example, for an oval or ellipse, where the incremental change in adjacent radii could be within RADITOL but the edge moves gradually away from or towards the geometric centre as the edge is traversed. Although this particular example would be unusual or rare for the shapes considered, future refinement of the program could be incorporated to account for these incremental possibilities.

5.1.1 Other Limitations of the System

Digitization of an image for processing will always create a less than ideal representation of a scene. Nevertheless, accurate results may indeed be obtained even with this limitation. The 512 x 512 pixel grid sufficiently represents the simple, regular shapes used in this study, and may be expected to do so as long as the shapes occupy a reasonable part of the full scene. But if more complex shapes were used, it may be desirable to have a finer resolution grid. With increased resolution, there is an increase in storage space required to hold the image and an increase in processing time to manipulate the data. Thus the hardware used and the software required is entirely dependent on the complexity of the objects used and the application of the problem, expressed in terms of the speed and accuracy required for the results.

In this study, only the ambient light of the room was used. Furthermore, the objects used were merely shapes cut

out of black cardboard attached to a white card. Thus, any dark shadows created by the slightly elevated shape became part of the object when thresholded. As it turned out in the tests performed, this did not create inaccurate identification. Indeed the tolerance for equality of sides, SIDETOL, could take any variances caused by this into account. However, in order to adapt to a manufacturing environment, where these shadows could create significant disturbances in the data, more testing would be necessary to determine the optimum lighting required, and the optimum thresholding level needed to reduce these disturbances to a workable level.

There was some overheating and accessing problems associated with the DT IRIS hardware and software. Combined with the memory available in the microcomputer, this caused the programs to reside as two separate routines. This greatly increased the time required to get the final results to about 15 to 20 seconds because of the time required to type in the commands. This however could be rectified in the future.

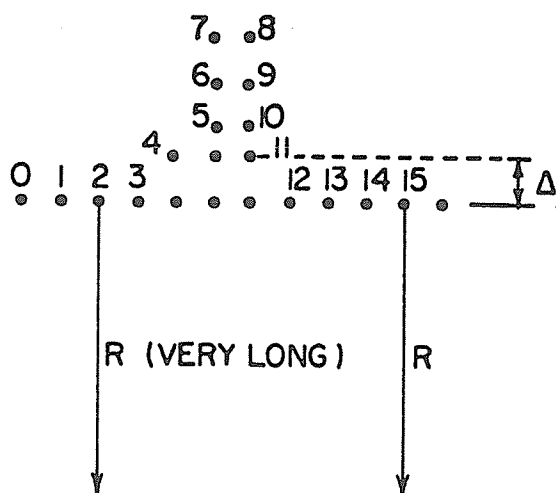
5.2 Recommendations for Future Enhancements

Initially it was assumed that only one object was present in the scene at one time, and that all black pixels are part of the object. The program, as it stands finds the first edge pixel and then follows a contiguous set of

pixels around a single object contour, and other such contours in the scene are ignored. The program could be modified to search for more than one object in the scene. Once the minimum number of pixels required to represent the edges of the objects is known, the number representing an identified edge may be compared to the minimum number to conclude that it is an object rather than a noisy blotch of the background.

A particular useful purpose of such a modification is evident. In order for a robot to grasp the object, it must know its physical location. This may be achieved by expanding the program to include calibration or reference points. Such a development would require the identification and location of a likely smaller reference point included in the scene with the object, and the ability to distinguish this point from blotches of noise shadows that could also be present in the scene. Alternatively, a partial frame could be used to contain the reference point.

One other problem that requires further work is the ability to identify "**spikes**" that do not represent the edge of an object. This is illustrated in Figure 19. The figure shows that the smoothing process, which uses the average of NUMAVE points along the edge to calculate the radius, is still unable to remove this "spike". The resulting TRANS array values, which are displayed at the



Note: DIAGRAM ENLARGED FOR CLARITY

PT #	(AVE. 5 PTS) APPROX. LENGTH	PT #	(AVE. 5 PTS) APPROX. LENGTH
0	R	8	$R + \frac{16\Delta}{5}$
1	R	9	$R + \frac{14\Delta}{5}$
2	$R + \frac{\Delta}{5}$	10	$R + 2\Delta$
3	$R + \frac{3\Delta}{5}$	11	$R + \frac{6\Delta}{5}$
4	$R + \frac{6\Delta}{5}$	12	$R + \frac{3\Delta}{5}$
5	$R + 2\Delta$	13	$R + \frac{\Delta}{5}$
6	$R + \frac{14\Delta}{5}$	14	R
7	$R + \frac{16\Delta}{5}$	15	R

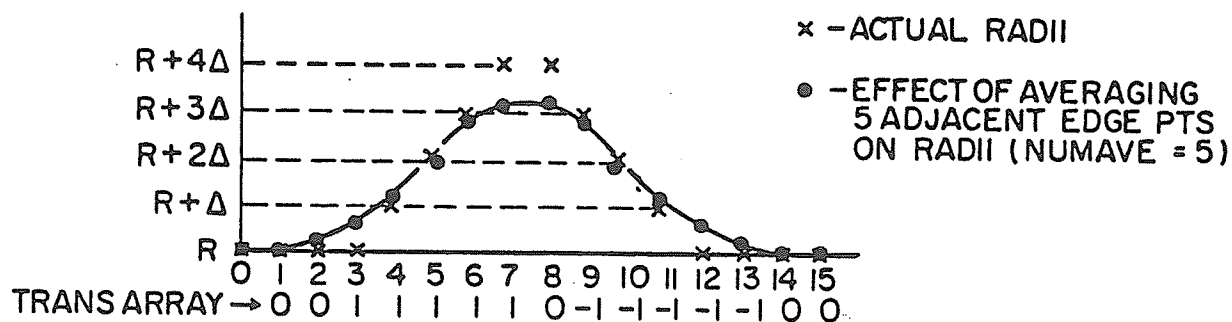


FIGURE 19: Analysis of Noise "Spike"

bottom of Figure 19, indicate the presence of a maximum. The program as it stands cannot detect whether this maximum is part of the object or is caused by extraneous noise. An extension of the program to examine the pattern of maxima and minima in an object could be introduced to remove this limitation.

5.2.1 Enhancing Execution Speed and Accuracy

Although the processing speed was taken into consideration throughout the program development, there are areas that could be modified to further reduce the number of computations. One improvement could be made with respect to averaging of points along the edge. Currently, every point on the edge has an average point associated with it in order to achieve subpixel accuracy. Further testing will likely reveal that it is unnecessary to use every point on the edge. By skipping to every third or fourth point to locate trends, a sufficiently accurate and efficient location of all transition areas may be obtained. More detailed averaging in these areas could then provide more accurate information required to extract the coordinates of the maximum or minimum points therein. Thus, the average points could be calculated for every third or fourth edge pixel, and comparison of the resulting radii could be performed for every third or fourth pixel as well. This would reduce considerably the number of

calculations and iterations required in the program.

Also, the manner in which two adjacent radii are compared could also be modified. Currently, the program compares the difference between two adjacent radii to variable RADITOL to determine whether the two radii are equal. This creates the possibility of errors due to incremental changes discussed in section 5.1. This might be alleviated by comparing the two radial distances on either side of the pixel in question, to determine whether it is lodged between equal radii, or whether it is located along an increasing or decreasing trend.

In addition, the software algorithms, which were written in the Fortran language, could be rewritten in the C language which lends itself to more efficient data manipulation and therefore to increased speed of operations. A majority of image work and image related graphics is presently written in the C language.

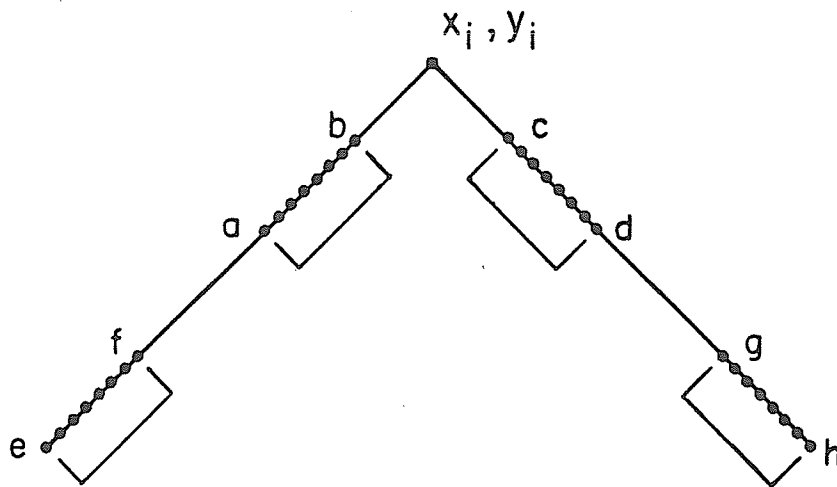
The program currently resides in two separate programs and must be called individually. By combining the two the execution time required can be reduced. Presently the program requires approximately 15 seconds to run both parts and obtain results, but most of this time is spent in typing in the program commands. It is estimated that when combined, the program will take no more than 8 seconds to obtain results. This could be further reduced if the objects residing in the scene utilized a smaller fraction

of the entire image. Currently they occupy approximately $1/4-1/3$ of the camera's field of view.

An expansion of the extrapolation process could provide more accurate results. The current program provides a transition point based on the calculations performed on the intersection of two lines, which are each represented by only the endpoints of each trend line (eight pixels in length). Although this method provides fairly accurate results because the images were reasonable ones, this accuracy could be increased by using more points to identify these lines.

This could be accomplished by two methods. A methodology could be developed to make use of more points between the two endpoints of each line used and some line fitting subroutine. Another method could use the other trend endpoints to produce the intersection. At present, the lines represented by the endpoints contain only eight edge points (eight is the optimum value of STLEN), and these eight points are located either at the end or beginning of a trend. The program could be modified to include the other ends of each trend, which would increase the number of points used and provide a more accurate calculation of the transition point. This is illustrated in Figure 20.

These methods would provide greater accuracy in identifying intersections of straight lines. However a



- a-b and c-d represent lines currently used to find x_i, y_i .
- expanding lines to e-f-a-b and c-d-g-h could improve accuracy in finding x_i, y_i .

FIGURE 20: Transition Point Calculation

more complex and time-consuming algorithm, such as a b-spline, would have to be used to represent curved edges. Additional testing would have to be performed to find the optimum method for distinguishing specific forms which would provide the accuracy desired without sacrificing too much processing speed. In order to include curved objects in the program's repertoire of identifiable shapes, a much more complex method of extrapolation must be incorporated into the present software.

5.3 Application of the Developed Algorithms

A conceptual although realistic application of the vision algorithms developed in this thesis would involve orienting the gripper of a robot arm to the proper alignment for the object to be grasped.

The vision system presented in this study provides the data required by the controller to perform a task. It enables the robot to accurately grasp an object regardless of its orientation and location, and place it in an appropriate, predetermined location with predetermined orientation. The information obtained from the vision algorithms, such as the type of object present, the location of the geometric center of the object (XG , YG), the corner coordinates, the lengths of the sides, and the orientation of the object, MINPOS, can be effectively used in such a control scheme.

5.4 Conclusions

In summary, the study set out to investigate the fundamentals of a microcomputer based vision system to create a simple object recognition module was achieved. From the results of Chapter IV, it may be fairly concluded that this objective has been achieved, albeit with accompanying limitations to its application.

The known limitations of the software were mainly due to the types of objects it can identify combined with the nature of the radial comparison technique employed by the program. Execution speed limitations were also present and included the nature of the Fortran language and the systems inability to contain the entire image at one time. It was also found that the accuracy could be enhanced in the module that calculates the intersection of two lines by introducing more points and/or incorporating a more sophisticated line interpolation and curve matching method.

Recommendations to alleviate the limitations found were discussed. Additional aspects such as false corner detection and reduction of computations required to obtain the results were also outlined.

REFERENCES

1. Chin, Roland T., "Machine Vision for Discrete Part Handling in Industry: A Survey." In IEEE: Industrial Applications of Machine Vision, pp. 26-32. IEEE Publications, 1982.
2. Hara, Y., Akiyama, N., Karasaki, K., "Automatic Inspection System for Printed Circuit Boards." In IEEE: Industrial Applications of Machine Vision, pp. 62-70. IEEE Publications, 1982.
3. Russell, G. T., "The Automatic Guidance and Control of an Unmanned Submersible." In Robotic Technology, pp. 52-62. Peter Peregrinus Ltd., London, 1983.
4. Gonzalez, Rafael C., Wintz, Paul, Digital Image Processing, Addison-Wesley Publishing Co., Reading, Massachusetts, 1977.
5. Faugeras, O. D., ed., Fundamentals in Computer Vision, Cambridge University Press, Cambridge, UK., 1983.
6. Ballard, D. H., Brown, C. M., Computer Vision, Prentice-Hall Inc., Englewood Cliffs, NJ., 1982.
7. Horn, Berthold K. P., Robot Vision, MIT Press, Cambridge, Massachusetts, 1986.
8. Microsoft Fortran77 V3.30 Reference Manual, Microsoft Corporation, 1985.
9. IRIS Tutor User Manual, V01.01, Data Translation Inc., Marlboro, MA., 1987.
10. Elgazzar, S., Green, D., O'Hara, D., "A Vision-Based Robot System Using a Multiprocessor Controller", NRCC no. 23485, June 1985.
11. Pugh, A., "Processing of Binary Images." In Robotic Technology, pp. 63-87. Edited by A. Pugh. Peter Peregrinus Ltd., London. UK., 1983.
12. Freeman, Herbert, "Computer Processing of Line-Drawing Images", Computing Surveys, Vol. 6, No. 1, March 1974.
13. Kreyszig, Erwin, Advanced Engineering Mathematics, 2nd edition, pp. 313-317. New York: John Wiley & Sons, Inc., 1967.

APPENDIX A

RESULTS DATA

KEY TO APPENDIX A

I.D. : Identification type.

ORIENT. ANGLE (DEG.) : Angle of Orientation, in degrees.

CORNER COORDINATES : Coordinates of each corner.

ROW, COL = ROW coordinate, COLumn coordinate.

LENGTH OF SIDES : Side lengths.

ASPECT RATIO (A.R.) : Ratio of the height to the width of an object. Height uses the length of one side.

AVERAGE A.R. : Ratio as above except the average to two sides is used for height.

AVE. : Average of column.

For NUMAVE data : average for NUMAVE = 5 to 21.

For STLEN data : average for STLEN = 4 to 20.

SPREAD : Maximum - minimum column value, as a percentage.

STD. : Standard deviation (N method).

ACTUAL : Measured aspect ratios.

% DIFFERENCE : Percentage difference between actual and average aspect ratios.

MAXIMUM : Maximum column value.

MINIMUM : Minimum column value.

MAX-MIN : MAXIMUM - MINIMUM column values.

CNTZERO : Number of zeros in TRANS array.

EDGE PTS. : Number of points on the edge.

NOTE: All data is taken at a fixed distance from the camera lens.

DATA A1: CONSISTENCY TEST - SQUARE

TEST	I.D. ORIENT.	CORNER COORDINATES				LENGTH OF SIDES				ASPECT AVERAGE			
		1	2	3	4	1	2	3	4	1	2	3	4
0	ANGLE (DEG.)	ROW	COL.	ROW	COL.	ROW	COL.	ROW	COL.	ROW	COL.	ROW	COL.
(A.R.)													

1	89.552	137.00	385.00	393.00	387.00	396.00	125.00	138.00	127.00	256.01	262.02	258.01	258.00	0.977	0.988
2	90	138.00	385.00	393.00	387.00	395.60	124.00	138.00	127.00	255.01	263.01	257.62	258.00	0.970	0.984
3	90	138.00	385.00	393.00	387.00	396.05	125.15	138.00	127.00	256.01	263.01	258.05	259.00	0.974	0.985
4	90	138.00	385.00	393.00	386.06	396.00	125.00	138.00	127.00	253.81	261.09	258.01	258.14	0.972	0.986
5	90.01	138.05	385.14	391.85	387.02	396.00	125.00	138.00	127.00	253.81	261.09	258.01	257.00	0.969	0.986
6	90	138.00	384.00	391.85	387.02	396.00	125.00	138.00	127.00	253.81	262.05	258.01	257.00	0.969	0.986
7	90	138.00	384.00	392.00	387.00	396.00	125.00	138.00	127.00	254.02	262.02	257.01	257.00	0.973	0.987
8	89.325	137.00	384.00	392.00	387.00	395.00	125.00	138.00	127.00	255.02	262.02	257.01	257.00	0.969	0.987
9	89.325	137.00	384.00	392.00	387.00	395.00	125.00	138.00	127.00	255.02	262.02	257.01	257.00	0.973	0.987
10	90	138.00	384.00	392.00	387.00	395.00	125.00	138.00	127.00	254.02	262.02	257.01	257.00	0.969	0.985
11	89.325	137.00	384.00	392.00	387.00	395.00	125.00	138.00	127.00	255.02	262.02	257.01	257.00	0.973	0.987
12	89.325	137.00	384.00	392.00	386.00	395.00	125.00	138.00	127.00	255.01	261.02	257.01	257.00	0.977	0.988
13	89.550	137.00	384.00	392.00	386.00	395.00	125.00	138.00	127.00	255.01	261.02	257.01	257.00	0.977	0.988
14	89.550	137.00	384.00	392.00	386.00	395.00	125.00	138.00	127.00	255.01	261.02	257.01	257.00	0.977	0.988
15	89.550	137.00	384.00	392.00	386.00	395.00	125.00	138.00	127.00	255.01	261.02	257.01	257.00	0.977	0.988
16	89.550	137.00	384.00	392.00	386.00	395.00	125.00	138.00	127.00	255.01	261.02	257.01	257.00	0.977	0.988
17	89.550	137.00	384.00	392.00	386.00	395.00	125.00	138.00	127.00	255.01	261.02	257.01	257.00	0.977	0.988
18	89.550	137.00	384.00	392.00	386.00	395.00	125.00	138.00	127.00	255.01	261.02	257.01	257.00	0.977	0.988
19	89.325	137.00	384.00	392.00	387.00	395.00	125.00	138.00	127.00	255.02	262.02	257.01	257.00	0.973	0.987
20	89.550	137.00	384.00	392.00	386.00	395.00	125.00	138.00	127.00	255.01	261.02	257.01	257.00	0.977	0.988

AVE.	89.652	137.35	384.25	392.18	386.55	395.33	124.95	138	126.95	254.84	261.61	257.34	257.30	0.9741	0.9870
SPREAD	0.761	0.761	0.761	0.761	0.761	0.761	0.761	0.761	0.761	0.761	0.761	0.761	0.761	0.761	0.761
STD.	0.2709	0.480	0.446	0.410	0.495	0.461	0.222	0.000	0.218	0.508	0.576	0.461	0.566	0.0030	0.0014
ACTUAL	1.0000	1.0013	1.0000	1.0013	1.0000	1.0013	1.0000	1.0013	1.0000	1.0013	1.0000	1.0013	1.0000	1.0013	1.0000
1 DIFFERENCE	138.05	385.14	393.00	387.02	396.05	125.15	138.00	127.00	256.01	263.01	258.05	259.00	0.977	0.988	
MINIMUM	137.00	384.00	391.85	386.00	395.00	124.00	138.00	126.00	253.81	261.02	257.01	257.00	0.969	0.984	
MAX-MIN	1.05	1.14	1.15	1.02	1.05	1.15	0.00	1.00	2.20	2.20	1.04	2.00	0.008	0.005	

DATA A3: INCREASING STLEN - SQUARES

STLEN	I.D.	ORIENT. ANGLE (DEG.)	CORNER COORDINATES								LENGTH OF SIDES				ASPECT AVERAGE	
			1	1	2	2	3	3	4	4	1	2	3	4	RATIO	A.R.
			ROW	COL.	ROW	COL.	ROW	COL.	ROW	COL.	A	B	C	D	(A/B)	(B/D)
1	SQ.	60.029	150.50	193.00	197.50	109.50	114.50	62.50	69.00	146.00	95.82	95.38	95.09	94.08	1.005	1.008
2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
3	SQ.	89.551	137	384	392	386	395	125	138	127	255.01	261.02	257.01	257.00	0.977	0.988
4	SQ.	89.551	137	384	392	386	395	125	138	127	255.01	261.02	257.01	257.00	0.977	0.988
5	SQ.	89.551	137	384	392	386	395	125	138	127	255.01	261.02	257.01	257.00	0.977	0.988
6	SQ.	89.551	137	384	392	386	395	125	138	127	255.01	261.02	257.01	257.00	0.977	0.988
8	SQ.	89.551	137	384	392	386	395	125	138	127	255.01	261.02	257.01	257.00	0.977	0.988
10	SQ.	89.551	137	384	392	386	395	125	138	127	255.01	261.02	257.01	257.00	0.977	0.988
15	SQ.	89.551	137	384	392	386	395.01	125.07	138	127	255.01	260.95	257.02	257.00	0.977	0.989
20	SQ.	89.539	136.99	383.95	392	386	395.01	125.05	138	127	255.02	260.97	257.02	256.95	0.977	0.989
AVE.		89.549	137.00	383.99	392.00	386.00	395.00	125.02	138.00	127.00	255.01	261.00	257.01	256.99	0.9770	0.9884
SPREAD		0.01%	0.01%	0.01%	0.00%	0.00%	0.00%	0.06%	0.00%	0.00%	0.00%	0.03%	0.00%	0.02%	0.03%	0.02%
STD.		0.004	0.003	0.017	0.000	0.000	0.005	0.028	0.000	0.000	0.004	0.028	0.004	0.017	0.0001	0.0001
ACTUAL															1.0000	1.0013
% DIFFERENCE															2.30%	1.28%
MAXIMUM		89.551	137.00	384.00	392.00	386.00	395.01	125.07	138.00	127.00	255.02	261.02	257.02	257.00	0.977	0.989
MINIMUM		89.539	136.99	383.95	392.00	386.00	395.00	125.00	138.00	127.00	255.01	260.95	257.01	256.95	0.977	0.988
MAX-MIN		0.012	0.01	0.05	0.00	0.00	0.01	0.07	0.00	0.00	0.01	0.07	0.01	0.05	.000	.000
1	SQ.	0.598	140.50	84.50	90.00	166.00	172.50	214.00	218.50	131.00	95.88	95.45	94.89	90.81	1.005	1.024
2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
3	SQ.	0.224	179	172	180	428	435	428	436	167	256.00	255.00	261.00	257.05	1.004	1.010
4	SQ.	0.224	179	172	180	428	434.6	428.2	436	167	256.00	254.60	261.20	257.05	1.006	1.011
5	SQ.	0.224	179	172	180	428	434.73	428.09	436	167	256.00	254.73	261.09	257.05	1.005	1.010
6	SQ.	0.224	179	172	180	428	434.79	428.05	436	167	256.00	254.79	261.05	257.05	1.005	1.010
8	SQ.	0.224	179	172	180	428	434.85	428.02	436	167	256.00	254.85	261.02	257.05	1.005	1.010
10	SQ.	0.224	179	172	180	428	434.89	428.01	436	167	256.00	254.89	261.01	257.05	1.004	1.010
15	SQ.	0.224	179	172	180	428	434.93	428.01	436	167	256.00	254.93	261.01	257.05	1.004	1.010
20	SQ.	0.224	179	172	180	428	434.95	428.01	436	167	256.00	254.95	261.01	257.05	1.004	1.010
AVE.		0.224	179.00	172.00	180.00	428.00	434.82	428.06	436.00	167.00	256.00	254.82	261.06	257.05	1.0046	1.0101
SPREAD		0.00%	0.00%	0.00%	0.00%	0.00%	0.08%	0.04%	0.00%	0.00%	0.00%	0.14%	0.07%	0.00%	0.14%	0.11%
STD.		.000	0.000	0.000	0.000	0.000	0.115	0.065	0.000	0.000	0.000	0.115	0.066	.000	0.0005	0.0004
ACTUAL															1.0000	1.0013
% DIFFERENCE															0.46%	0.88%
MAXIMUM		0.224	179.00	172.00	180.00	428.00	434.95	428.20	436.00	167.00	256.00	254.95	261.20	257.05	1.006	1.011
MINIMUM		0.224	179.00	172.00	180.00	428.00	434.60	428.01	436.00	167.00	256.00	254.60	261.01	257.05	1.004	1.010
MAX-MIN		0	0.00	0.00	0.00	0.00	0.35	0.19	0.00	0.00	0.00	0.35	0.19	0.00	0.001	0.001
1	SQ.	81.185	140.50	84.50	90.00	166.00	172.50	214.00	218.50	131.00	95.88	95.45	94.89	90.81	1.005	1.024
2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
4	SQ.	23.231	101.75	161.25	203.2	397.6	437.5	295.5	337	53.33	257.20	255.58	262.20	258.82	1.006	1.010
5	SQ.	23.264	101.67	161.33	203.18	397.46	437.4	295.6	336.43	53.29	257.02	255.41	262.51	258.43	1.006	1.011
6	SQ.	23.319	101.69	161.23	203.57	397.67	437.33	295.67	336.43	53.43	257.45	255.04	262.41	258.31	1.009	1.013
8	SQ.	23.33	101.67	161.22	203.54	397.42	437.42	295.48	336.34	53.43	257.23	255.13	262.31	258.24	1.008	1.012
10	SQ.	23.345	101.65	161.22	203.62	397.46	437.47	295.4	336.3	53.42	257.31	255.15	262.28	258.23	1.008	1.012
15	SQ.	23.357	101.66	161.28	203.69	397.47	437.47	295.32	336.27	53.46	257.28	255.12	262.18	258.20	1.008	1.012
20	SQ.	23.363	101.66	161.19	203.75	397.53	437.48	295.29	336.25	53.47	257.45	255.11	262.15	258.14	1.009	1.012
AVE.		23.315	101.68	161.25	203.51	397.52	437.44	295.47	336.43	53.40	257.28	255.22	262.29	258.34	1.0080	1.0117
SPREAD		0.57%	0.10%	0.09%	0.28%	0.06%	0.04%	0.13%	0.22%	0.34%	0.17%	0.21%	0.13%	0.26%	0.31%	0.29%
STD.		0.046	0.031	0.044	0.211	0.083	0.053	0.130	0.241	0.063	0.139	0.181	0.121	0.215	0.0012	0.0009
ACTUAL															1.0000	1.0013
% DIFFERENCE															0.81%	1.04%
MAXIMUM			101.75	161.33	203.75	397.67	437.50	295.67	337.00	53.47	257.45	255.58	262.51	258.82	1.009	1.013
MINIMUM			101.65	161.19	203.18	397.42	437.33	295.29	336.25	53.29	257.02	255.04	262.15	258.14	1.006	1.010
MAX-MIN			0.10	0.14	0.57	0.25	0.17	0.38	0.75	0.18	0.43	0.53	0.35	0.68	0.003	0.003

DATA A4: INCREASING NUMAVE - SQUARES

NUM- AVE	I.D.	ORIENT. ANGLE (DEG.)	CORNER COORDINATES								LENGTH OF SIDES				ASPECT AVERAGE	
			1	1	2	2	3	3	4	4	1	2	3	4	RATIO	A.R.
			ROW	COL.	ROW	COL.	ROW	COL.	ROW	COL.					(A/B)	(A+C)/ (B+D)
1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
3	SO.	89.326	137	384	392	387	396	125	138	126	255.02	262.03	258.00	258.00	0.973	0.987
5	SO.	89.551	137	384	392	386	395	125	138	127	255.01	261.02	257.01	257.00	0.977	0.988
7	SO.	89.549	138	384	392	386	395	125	138	127	254.01	261.02	257.01	257.00	0.973	0.986
9	SO.	90	138	384	392	385	394	125	138	128	254.00	260.01	256.02	256.00	0.977	0.988
11	SO.	90.224	139	384	392	385	394	125	138	128	253.00	260.01	256.02	256.00	0.973	0.986
13	SO.	90	139	384	392	384	393	125	138	129	253.00	259.00	255.03	255.00	0.977	0.988
15	SO.	90	140	384	392	384	393	125	138	129	252.00	259.00	255.03	255.01	0.973	0.986
17	SO.	90	140.44	383.78	392	383	392	125	138	130	251.56	258.00	254.05	253.79	0.975	0.988
19	SO.	90.552	140.44	383.78	392	383	391	125	138	130	251.56	258.00	253.05	253.79	0.975	0.986
21	SO.	90.554	140.44	383.78	392	382	390	125	138	131	251.57	257.01	252.07	252.79	0.979	0.988
Ave.		90.047	139.15	383.93	392.00	384.22	393.00	125.00	138.00	128.78	252.86	259.23	255.03	255.15	0.9754	0.9873
SPREAD		1.12%	2.47%	0.06%	0.00%	1.04%	1.27%	0.00%	0.00%	3.11%	1.36%	1.55%	1.94%	1.65%	0.60%	0.25%
STD.		0.341	1.203	0.104	0.000	1.315	1.633	0.000	0.000	1.315	1.205	1.319	1.613	1.398	0.002	0.001
ACTUAL															1.0000	1.0013
% DIFFERENCE															2.46%	1.39%
MAXIMUM		90.554	140.44	384.00	392.00	386.00	395.00	125.00	138.00	131.00	255.01	261.02	257.01	257.00	0.979	0.988
MINIMUM		89.549	137.00	383.78	392.00	382.00	390.00	125.00	138.00	127.00	251.56	257.01	252.07	252.79	0.973	0.986
MAX-MIN		1.005	3.44	0.22	0.00	4.00	5.00	0.00	0.00	4.00	3.45	4.01	4.94	4.21	0.006	0.002
1	SO.	0	179.00	171.00	179.00	428.00	434.86	429.00	437.00	167.00	257.00	255.86	262.01	258.03	1.004	1.010
3	SO.	0.223	179.00	171.00	180	428.00	434.86	429	436	167.00	257.00	254.86	262.00	257.03	1.008	1.014
5	SO.	0.224	179.00	172.00	180	428.00	434.86	428.02	436	167.00	256.00	254.86	261.02	257.05	1.004	1.010
7	SO.	0.448	179.00	172.00	181	428.00	434.85	427.07	435	167.00	256.01	253.85	260.07	256.05	1.008	1.012
9	SO.	0.449	179.00	173.00	181	428.00	434.84	426.12	435	167.00	255.01	253.85	259.12	256.07	1.005	1.008
11	SO.	0.674	179.00	173.00	182	428.00	434.84	426.12	434	167.00	255.02	252.85	259.12	255.07	1.009	1.012
13	SO.	0.677	179.00	174.00	182	428.00	436	423.67	434	167.00	254.02	254.04	256.68	255.10	1.000	1.003
15	SO.	0.902	179.00	174.00	183	428.00	436	423.67	433	167.00	254.03	253.04	256.69	254.10	1.004	1.007
17	SO.	0.906	179.00	175.00	183	428.00	436	424	432	167.00	253.03	253.03	257.03	253.13	1.000	1.008
19	SO.	1.137	179.00	176.00	184	428.00	436	424	432	167.00	252.05	252.03	257.03	253.16	1.000	1.008
21	SO.	1.364	179.00	176.00	185	428.00	436	423	431	167.00	252.07	251.05	256.05	252.16	1.004	1.010
Ave.		0.7534	179.00	173.89	182.33	428.00	435.49	425.07	433.56	167.00	254.14	253.18	258.09	254.65	1.0037	1.0086
SPREAD		151.31%	0.00%	2.30%	2.74%	0.00%	0.27%	1.18%	1.15%	0.00%	1.56%	1.50%	1.93%	1.92%	0.86%	0.91%
STD.		0.341	0.000	1.449	1.491	0.000	0.573	1.680	1.571	0.000	1.429	1.078	1.667	1.533	0.003	0.003
ACTUAL															1.0000	1.0013
% DIFFERENCE															0.38%	0.73%
MAXIMUM		1.364	179.00	176.00	185.00	428.00	436.00	428.02	436.00	167.00	256.01	254.86	261.02	257.05	1.009	1.012
MINIMUM		0.224	179.00	172.00	180.00	428.00	434.84	423.00	431.00	167.00	252.05	251.05	256.05	252.16	1.000	1.003
MAX-MIN		1.14	0.00	4.00	5.00	0.00	1.16	5.02	5.00	0.00	3.96	3.81	4.97	4.89	0.009	0.009
1	SO.	23.242	101.66	160.20	203.54	397.42	438.34	295.80	336.34	53.43	258.17	255.85	262.96	257.83	1.009	1.015
3	SO.	23.175	101.66	161.2	203.18	398.35	438.34	295.8	336.82	52.65	257.97	256.55	263.49	259.00	1.006	1.011
5	SO.	23.33	101.66	161.22	203.54	397.42	437.42	295.48	336.34	53.43	257.24	255.13	262.31	258.25	1.008	1.012
7	SO.	23.213	102.4	161.6	203.54	397.42	437.28	293.65	336.34	53.43	256.59	255.74	260.57	257.74	1.003	1.007
9	SO.	23.691	102.75	163.42	205.36	397.27	437.23	292.54	336.34	53.43	255.37	254.42	259.52	258.19	1.004	1.004
11	SO.	23.691	102.75	163.42	205.36	397.27	437.23	292.54	334.58	53.75	255.37	254.42	259.92	256.46	1.004	1.009
13	SO.	23.934	102.75	163.42	206.52	397.21	437.23	292.54	334.58	53.75	255.79	253.34	259.92	256.46	1.010	1.012
15	SO.	24.026	102.81	164.56	206.52	397.21	436.62	293.15	334.69	54.28	254.72	252.54	259.71	256.77	1.009	1.010
17	SO.	23.961	102.81	164.56	206.04	396.84	436.62	293.15	332	55	254.19	252.82	260.12	254.03	1.005	1.015
19	SO.	23.842	103.16	164.04	206.04	396.84	436	290	332	55	254.52	253.57	256.98	253.49	1.004	1.009
21	SO.	24.682	103.16	164.04	209.64	395.73	436	290	332	55	254.99	249.84	256.98	253.49	1.021	1.017
Ave.		23.818	102.69	163.36	205.84	397.02	436.85	292.56	334.32	54.12	255.42	253.54	259.56	256.10	1.0074	1.0105
SPREAD		6.17%	1.46%	2.04%	2.96%	0.43%	0.33%	1.87%	1.30%	2.90%	1.19%	2.33%	2.05%	1.86%	1.72%	1.26%
STD.		0.403	0.425	1.130	1.715	0.500	0.526	1.617	1.785	0.671	0.932	1.641	1.578	1.834	0.005	0.004
ACTUAL															1.0000	1.0013
% DIFFERENCE															0.75%	0.92%
MAXIMUM		24.682	103.16	164.56	209.64	397.42	437.42	295.48	336.34	55.00	257.24	255.74	262.31	258.25	1.021	1.017
MINIMUM		23.213	101.66	161.22	203.54	395.73	436.00	290.00	332.00	53.43	254.19	249.84	256.98	253.49	1.003	1.004
MAX-MIN		1.469	1.50	3.34	6.10	1.69	1.42	5.48	4.34	1.57	3.05	5.90	5.32	4.76	0.017	0.013

DATA B1: CONSISTENCY TEST - RECTANGLE

TEST #	I.D.	ORIENT.	CORNER COORDINATES								LENGTH OF SIDES				ASPECT AVERAGE	
			ANGLE (DEG.)	1 ROW	1 COL.	2 ROW	2 COL.	3 ROW	3 COL.	4 ROW	4 COL.	1	2	3	4	RATIO (A.R.)
1	RECT.	70.269	225.33	308.57	310.15	87.92	147.83	31.07	67.96	252.13	236.39	171.99	235.05	167.18	0.707	0.719
2	RECT.	70.316	225.42	309.04	310.15	87.92	147.6	31.6	67.9	252.69	236.80	172.03	235.02	167.30	0.706	0.719
3	RECT.	70.316	225.42	309.04	310.15	87.92	147.6	31.6	67.9	252.69	236.80	172.03	235.02	167.30	0.706	0.719
4	RECT.	70.316	225.42	309.04	310.15	87.92	147.6	31.6	67.9	252.69	236.80	172.03	235.02	167.30	0.706	0.719
5	RECT.	70.339	225.42	309.04	310.15	87.92	147.6	31.6	66.89	252.4	236.80	172.03	235.09	168.34	0.711	0.721
6	RECT.	70.339	225.42	309.04	310.15	87.92	147.67	31.43	66.89	252.4	236.80	172.02	235.27	168.34	0.711	0.721
7	RECT.	70.339	225.42	309.04	310.15	87.92	147.67	31.43	66.89	252.4	236.80	172.02	235.27	168.34	0.711	0.721
8	RECT.	70.339	225.42	309.04	310.15	87.92	147.67	31.43	66.89	252.4	236.80	172.02	235.27	168.34	0.711	0.721
9	RECT.	70.339	225.42	309.04	310.15	87.92	147.67	31.43	66.89	252.4	236.80	172.02	235.27	168.34	0.711	0.721
10	RECT.	70.339	225.42	309.04	310.15	87.92	147.67	31.43	66.89	252.4	236.80	172.02	235.27	168.34	0.711	0.721
11	RECT.	70.339	225.42	309.04	310.15	87.92	147.67	31.43	66.89	252.4	236.80	172.02	235.27	168.34	0.711	0.721
12	RECT.	70.339	225.42	309.04	310.15	87.92	147.6	31.6	66.89	252.4	236.80	172.03	235.09	168.34	0.711	0.721
13	RECT.	70.339	225.42	309.04	310.15	87.92	147.67	31.43	66.89	252.4	236.80	172.02	235.27	168.34	0.711	0.721
14	RECT.	70.339	225.42	309.04	310.15	87.92	147.67	31.43	66.89	252.4	236.80	172.02	235.27	168.34	0.711	0.721
15	RECT.	70.339	225.42	309.04	310.15	87.92	147.67	31.43	66.89	252.4	236.80	172.02	235.27	168.34	0.711	0.721
16	RECT.	70.339	225.42	309.04	310.15	87.92	147.67	31.43	66.89	252.4	236.80	172.02	235.27	168.34	0.711	0.721
17	RECT.	70.339	225.42	309.04	310.15	87.92	147.67	31.43	66.89	252.4	236.80	172.02	235.27	168.34	0.711	0.721
18	RECT.	70.339	225.42	309.04	310.15	87.92	147.67	31.43	66.89	252.4	236.80	172.02	235.27	168.34	0.711	0.721
19	RECT.	70.339	225.42	309.04	310.15	87.92	147.67	31.43	66.89	252.4	236.80	172.02	235.27	168.34	0.711	0.721
20	RECT.	70.339	225.42	309.04	310.15	87.92	147.67	31.43	66.89	252.4	236.80	172.02	235.27	168.34	0.711	0.721
AVE.		70.335	225.42	309.04	310.15	87.92	147.65	31.474	67.049	252.44	236.79	172.02	235.21	168.17	0.7102	0.7207
SPREAD		0.03%	0.00%	0.00%	0.00%	0.00%	0.05%	0.54%	1.51%	0.11%	0.00%	0.01%	0.11%	0.62%	0.62%	0.29%
STD.		0.008	.000	.000	ERR	ERR	0.031	0.075	0.368	0.106	.000	0.005	0.102	0.382	0.002	0.001
ACTUAL																0.7102 0.7074
% DIFFERENCE																.00% -1.89%
MAXIMUM		70.339	225.42	309.04	310.15	87.92	147.67	31.6	67.9	252.69	236.79	172.03	235.27	168.34	0.711	0.721
MINIMUM		70.316	225.42	309.04	310.15	87.92	147.6	31.43	66.89	252.4	236.79	172.01	235.01	167.30	0.706	0.719
MAX-MIN		0.023	0.00	0.00	0.00	0.00	0.07	0.17	1.01	0.29	0.00	0.01	0.26	1.05	0.004	0.002

DATA B3: INCREASING STLEN - RECTANGLE

STLEN	I.D.	ORIENT.	CORNER COORDINATES								LENGTH OF SIDES				ASPECT AVERAGE		
		ANGLE (DEG.)	1 ROW	1 COL.	2 ROW	2 COL.	3 ROW	3 COL.	4 ROW	4 COL.	1	2	3	4	RATIO (A,R.)	A.R. (A+C) / (B+D)	
											A	B	C	D			
1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
2	RECT.	69.959	67	253	226	311	301	89	148.5	31.5	169.25	234.33	162.98	236.02	0.722	0.706	
3	RECT.	70.118	67	252	226	309.5	301	88	147.8	31.4	169.08	233.85	163.32	234.93	0.723	0.709	
4	RECT.	70.385	66.9	252.3	226	309	301	88.33	147.91	31.64	168.90	233.07	163.25	235.06	0.725	0.710	
5	RECT.	70.307	66.88	252.47	225.44	309.22	301.22	88.11	147.8	31.4	168.41	233.74	163.57	235.41	0.721	0.708	
6	RECT.	70.282	66.93	252.17	225.59	309.04	301.22	87.89	147.62	31.45	168.54	233.72	163.64	235.01	0.721	0.709	
8	RECT.	70.339	66.89	252.4	225.42	309.04	301.29	87.98	147.67	31.43	168.34	233.72	163.70	235.27	0.720	0.708	
10	RECT.	70.381	66.89	252.3	225.4	308.8	301.23	87.99	147.65	31.55	168.28	233.47	163.62	235.06	0.721	0.708	
15	RECT.	70.346	66.91	252.25	225.46	308.88	301.33	87.83	147.64	31.51	168.36	233.71	163.68	235.04	0.720	0.708	
20	RECT.	70.365	66.89	252.34	225.41	308.88	301.36	87.87	147.63	31.5	168.30	233.70	163.74	235.14	0.720	0.708	
AVE.			70.343	66.921	252.31	225.53	308.98	301.23	88	147.70	31.497	168.44	233.58	163.59	235.14	0.7211	0.7084
SPREAD			0.15%	0.18%	0.12%	0.27%	0.14%	0.12%	0.57%	0.20%	0.76%	0.37%	0.29%	0.30%	0.17%	0.63%	0.27%
STD.			0.035	0.044	0.091	0.200	0.130	0.109	0.160	0.102	0.075	0.202	0.230	0.152	0.138	0.001	0.001
ACTUAL																0.7102	0.7074
% DIFFERENCE																1.54%	0.14%
MAXIMUM			70.385	67.00	252.47	226.00	309.22	301.36	88.33	147.91	31.64	168.90	233.74	163.74	235.41	0.725	0.710
MINIMUM			70.282	66.88	252.17	225.40	308.80	301.00	87.83	147.62	31.40	168.28	233.07	163.25	235.01	0.720	0.708
MAX-MIN			0.103	0.12	0.30	0.60	0.42	0.36	0.50	0.29	0.24	0.62	0.67	0.49	0.41	0.005	0.002

DATA B4: INCREASING NUMAVE -RECTANGLE

NUM- AVE	ORIENT. I.D. ANGLE (DEG.)	CORNER COORDINATES								LENGTH OF SIDES				ASPECT AVERAGE	
		1	1	2	2	3	3	4	4	1	2	3	4	RATIO (A.R.)	A.R. (A+C)/ (D+A) (B+D)
		ROW	COL.	ROW	COL.	ROW	COL.	ROW	COL.	A	B	C	D	(D/A)	(B+D)
1	RECT.	70.34	225.42	309.04	301.02	87.87	147.67	31.43	66.89	252.40	233.73	163.41	235.27	168.34	0.720 0.707
3	RECT.	70.545	225.59	308.45	301.02	87.87	147.49	31.86	66.89	252.40	233.12	163.43	234.81	168.31	0.722 0.709
5	RECT.	70.339	225.42	309.04	301.29	87.98	147.67	31.43	66.89	252.40	233.72	163.70	235.27	168.34	0.720 0.708
7	RECT.	70.108	225.16	308.93	301.57	87.02	147.19	31.22	66.74	251.6	234.70	164.15	234.61	168.47	0.718 0.709
9	RECT.	70.442	225.43	307.98	301.57	87.02	146.82	32.09	66.74	251.6	233.71	164.21	233.66	168.41	0.721 0.712
11	RECT.	70.83	225.43	307.98	301.58	87.96	146.82	32.09	66.29	252.66	232.83	164.54	234.81	168.48	0.724 0.712
13	RECT.	70.643	225.36	308.25	301.33	87.86	146.93	31.84	66.4	252.4	233.12	164.25	234.80	168.49	0.723 0.711
15	RECT.	70.637	224.69	308.06	301.94	86.7	146.66	31.14	66.02	252.29	234.45	164.92	235.39	168.19	0.717 0.709
17	RECT.	70.577	224.86	307.67	301.94	86.7	146.76	32.22	66.48	251.21	234.03	164.47	233.24	168.14	0.718 0.712
19	RECT.	70.377	224.86	307.67	302.13	87.04	146.76	32.22	66.48	251.21	233.77	164.76	233.24	168.14	0.719 0.713
21	RECT.	70.805	224.75	307.93	301.76	88.33	147.25	32.07	66.87	252.96	232.71	164.43	235.06	167.18	0.718 0.709
Ave.		70.528	225.11	308.17	301.68	87.40	146.98	31.81	66.55	252.04	233.67	164.38	234.45	168.20	0.7198 0.7104
SPREAD		1.02%	0.33%	0.44%	0.28%	1.86%	0.69%	3.39%	1.31%	0.69%	0.85%	0.74%	0.92%	0.78%	0.87% 0.68%
STD.		0.221	0.298	0.469	0.269	0.589	0.307	0.409	0.273	0.607	0.645	0.339	0.800	0.388	0.0021 0.0017
ACTUAL															0.7102 0.7074
% DIFFERENCE															1.36% 0.43%
MAXIMUM		70.83	225.43	309.04	302.13	88.33	147.67	32.22	66.89	252.96	234.70	164.92	235.39	168.49	0.724 0.713
MINIMUM		70.108	224.69	307.67	301.29	86.70	146.66	31.14	66.02	251.21	232.71	163.70	233.24	167.18	0.717 0.708
MAX-MIN		0.722	0.74	1.37	0.84	1.63	1.01	1.08	0.87	1.75	1.99	1.22	2.15	1.31	0.006 0.005

DATA C1: CONSISTENCY TEST - EQUILATERAL TRIANGLE

TEST #	I.D.	ORIENT. ANGLE (DEG.)	CORNER COORDINATES						LENGTH OF SIDES			HEIGHT (H)	ASPECT RATIO (A.R.) (H/B)	AVERAGE A.R.	
			1	1	2	2	3	3	1	2	3				
			ROW	COL.	ROW	COL.	ROW	COL.	1	2	3				
1	EQUIL.	61.330	163.70	254.25	390.00	378.00	389.06	115.40	257.93	262.60	264.70	222.00	0.845	0.860	
2	EQUIL.	61.439	163.29	254.59	390.00	378.00	389.08	115.54	258.12	262.46	265.17	222.27	0.847	0.862	
3	EQUIL.	61.439	163.29	254.59	390.00	378.00	389.00	115.57	258.12	262.43	265.09	222.28	0.847	0.862	
4	EQUIL.	61.439	162.67	254.24	390.15	378.07	389.00	115.57	259.00	262.50	265.43	223.28	0.851	0.865	
5	EQUIL.	61.619	162.65	254.23	390.15	378.07	389.00	115.57	259.02	262.50	265.44	223.31	0.851	0.865	
6	EQUIL.	61.443	163.29	254.59	390.15	378.07	389.00	115.57	258.29	262.50	265.09	222.45	0.847	0.862	
7	EQUIL.	61.439	162.67	254.24	390.15	378.07	389.00	115.57	259.00	262.50	265.43	223.28	0.851	0.865	
8	EQUIL.	61.324	162.67	254.24	390.24	378.71	389.00	114.43	259.39	264.28	266.03	223.20	0.845	0.859	
9	EQUIL.	61.358	162.65	254.23	390.22	378.52	389.00	115.57	259.30	262.95	265.44	223.50	0.850	0.863	
10	EQUIL.	61.358	162.65	254.23	390.22	378.52	389.00	114.43	259.30	264.09	266.04	223.16	0.845	0.860	
11	EQUIL.	61.358	162.65	254.23	390.22	378.52	389.00	114.43	259.30	264.09	266.04	223.16	0.845	0.860	
12	EQUIL.	61.358	162.65	254.23	390.22	378.52	389.00	115.57	259.30	262.95	265.44	223.50	0.850	0.863	
13	EQUIL.	61.330	162.65	254.23	389.00	378.00	389.00	114.43	257.98	263.57	266.04	221.78	0.841	0.859	
14	EQUIL.	61.330	162.65	254.23	389.00	378.00	389.00	114.43	257.98	263.57	266.04	221.78	0.841	0.859	
15	EQUIL.	61.330	162.65	254.23	389.00	378.00	389.00	114.43	257.98	263.57	266.04	221.78	0.841	0.859	
16	EQUIL.	61.330	162.65	254.23	389.00	378.00	389.00	114.43	257.98	263.57	266.04	221.78	0.841	0.859	
17	EQUIL.	61.330	162.65	254.23	389.00	378.00	389.00	114.43	257.98	263.57	266.04	221.78	0.841	0.859	
18	EQUIL.	61.330	162.65	254.23	389.00	378.00	388.06	115.39	257.98	262.61	264.74	222.06	0.846	0.861	
19	EQUIL.	61.330	162.65	254.23	389.00	378.00	388.06	115.39	257.98	262.61	264.74	222.06	0.846	0.861	
20	EQUIL.	61.330	162.65	254.23	389.00	378.00	388.06	115.39	257.98	262.61	264.74	222.06	0.846	0.861	
AVE.															
SPREAD			0.48%	0.64%	0.14%	0.32%	0.19%	0.26%	0.99%	0.56%	0.70%	0.51%	0.77%	1.09%	0.66%
STD.			0.0715	0.306	0.128	0.564	0.238	0.339	0.532	0.589	0.623	0.509	0.660	0.0032	0.0020
ACTUAL															
1 DIFFERENCE															
MAXIMUM			61.619	163.70	254.59	390.24	378.71	389.08	115.57	259.39	264.28	266.04	223.50	0.851	0.865
MINIMUM			61.324	162.65	254.23	389.00	378.00	388.06	114.43	257.93	262.43	264.70	221.78	0.841	0.859
MAX-MIN			0.295	1.05	0.36	1.24	0.71	1.02	1.14	1.46	1.85	1.34	1.72	0.009	0.006

DATA C3: INCREASING STLEN - EQUILATERAL TRIANGLES

STLEN	ORIENT. I.D. ANGLE (DEG.)	CORNER COORDINATES						LENGTH OF SIDES			HEIGHT (H)	ASPECT RATIO (A.R.) (H/B)	AVERAGE A.R.
		1 ROW	1 COL.	2 ROW	2 COL.	3 ROW	3 COL.	1 (B)	2	3			
1	EQUIL. 7.824	142.00	160.50	194.50	121.00	133.00	95.00	65.70	66.77	66.12	56.59	0.847	0.851
2	EQUIL. 61.248	163.00	254.00	389.00	378.00	389.00	116.00	257.78	262.00	264.80	222.02	0.847	0.863
3	EQUIL. 61.347	162.80	254.40	389.00	378.00	388.20	115.40	257.77	262.60	264.81	221.82	0.845	0.860
4	EQUIL. 61.308	162.80	254.20	389.00	378.00	388.09	115.27	257.86	262.73	264.68	221.89	0.845	0.860
5	EQUIL. 61.348	162.67	254.33	389.00	378.00	388.11	115.44	257.91	262.56	264.79	222.00	0.846	0.861
6	EQUIL. 61.323	162.70	254.28	389.00	378.00	388.07	115.36	257.91	262.64	264.74	221.97	0.845	0.860
8	EQUIL. 61.330	162.65	254.23	389.00	378.00	388.06	115.39	257.98	262.61	264.74	222.06	0.846	0.861
10	EQUIL. 61.334	162.66	254.26	389.00	378.00	388.05	115.42	257.96	262.58	264.72	222.05	0.846	0.861
15	EQUIL. 61.330	162.66	254.24	389.00	378.00	388.03	115.41	257.97	262.59	264.70	222.05	0.846	0.860
20	EQUIL. 61.328	162.66	254.23	389.00	378.00	388.02	115.41	257.97	262.59	264.68	222.06	0.846	0.860
Ave.	61.328	162.69	254.25	389.00	378.00	388.06	115.39	257.94	262.62	264.72	222.01	0.8453	0.8603
SPREAD	0.07%	0.09%	0.05%	0.00%	0.00%	0.02%	0.15%	0.05%	0.06%	0.04%	0.08%	0.13%	0.13%
STD.	0.0111	0.05	0.04	0.00	0.00	0.03	0.05	0.04	0.05	0.04	0.06	0.0003	0.0003
ACTUAL												0.8699	0.8768
% DIFFERENCE												2.82%	1.87%
MAXIMUM	61.348	162.80	254.33	389.00	378.00	388.11	115.44	257.98	262.73	264.79	222.06	0.846	0.861
MINIMUM	61.308	162.65	254.20	389.00	378.00	388.02	115.27	257.86	262.56	264.68	221.89	0.845	0.860
MAX-MIN	0.04	0.15	0.13	0.00	0.00	0.09	0.17	0.12	0.17	0.11	0.17	0.001	0.001
1	EQUIL. 8.062	105.50	135.50	147.00	188.50	170.50	127.00	67.31	65.84	65.55	58.72	0.892	0.876
2	-	-	-	-	-	-	-	-	-	-	-	-	-
3	EQUIL. 89.978	279.50	148.00	153.20	376.40	407.75	376.50	260.99	254.55	262.03	227.86	0.895	0.897
4	EQUIL. 89.955	279.43	147.71	153.10	376.30	407.50	376.50	261.18	254.40	262.20	228.11	0.897	0.899
5	EQUIL. 89.993	279.63	147.75	153.00	376.25	407.63	376.25	261.24	254.63	261.91	228.12	0.896	0.897
6	EQUIL. 89.991	279.57	147.57	152.96	376.21	407.50	376.25	261.35	254.54	262.03	228.27	0.897	0.898
8	EQUIL. 89.996	279.75	147.69	152.96	376.15	407.50	376.17	261.28	254.54	261.77	228.19	0.896	0.898
10	EQUIL. 89.998	279.75	147.65	152.96	376.12	407.50	376.13	261.29	254.54	261.77	228.20	0.897	0.898
15	EQUIL. 90.001	279.71	147.57	152.97	376.07	407.54	376.07	261.30	254.57	261.83	228.20	0.896	0.898
20	EQUIL. 90.001	279.74	147.60	152.97	376.05	407.55	376.05	261.27	254.58	261.77	228.16	0.896	0.897
Ave.	89.990	279.65	147.65	152.99	376.16	407.53	376.20	261.27	254.54	261.90	228.18	0.8964	0.8978
SPREAD	0.05%	0.11%	0.12%	0.09%	0.07%	0.03%	0.12%	0.07%	0.09%	0.16%	0.07%	0.10%	0.18%
STD.	0.0149	0.11	0.07	0.05	0.09	0.04	0.14	0.05	0.07	0.15	0.05	0.0002	0.0005
ACTUAL												0.8699	0.8768
% DIFFERENCE												3.05%	2.40%
MAXIMUM	90.001	279.75	147.75	153.10	376.30	407.63	376.50	261.35	254.63	262.20	228.27	0.897	0.899
MINIMUM	89.955	279.43	147.57	152.96	376.05	407.50	376.05	261.18	254.40	261.77	228.11	0.896	0.897
MAX-MIN	0.046	0.32	0.18	0.14	0.25	0.13	0.45	0.18	0.23	0.43	0.17	0.001	0.002
1	EQUIL. 40.236	137.00	130.50	181.00	182.50	201.00	121.00	68.12	64.67	64.70	59.95	0.927	0.897
2	-	-	-	-	-	-	-	-	-	-	-	-	-
3	EQUIL. 5.052	332.50	140.00	220.20	376.40	477.33	353.67	261.72	258.13	258.13	227.68	0.882	0.874
4	EQUIL. 5.047	332.50	140.25	220.00	376.33	477.40	353.60	261.51	258.40	257.90	227.37	0.880	0.872
5	EQUIL. 4.981	332.38	140.25	220.00	376.25	477.60	353.80	261.39	258.58	258.25	227.18	0.879	0.872
6	EQUIL. 4.990	332.38	140.38	219.96	376.21	477.57	353.71	261.25	258.59	258.05	227.02	0.878	0.871
8	EQUIL. 4.956	332.33	140.42	219.96	376.15	477.53	353.81	261.14	258.54	258.11	226.90	0.878	0.871
10	EQUIL. 4.938	332.34	140.39	219.96	376.12	477.57	353.86	261.15	258.57	258.19	226.90	0.878	0.871
15	EQUIL. 4.918	332.30	140.40	219.97	376.07	477.64	353.90	261.07	258.62	258.27	226.80	0.877	0.871
20	EQUIL. 4.915	332.28	140.40	219.98	376.05	477.61	353.90	261.04	258.58	258.27	226.77	0.877	0.871
Ave.	4.9635	332.36	140.36	219.98	376.17	477.56	353.80	261.22	258.55	258.15	226.99	0.8779	0.8710
SPREAD	2.66%	0.07%	0.12%	0.02%	0.07%	0.05%	0.08%	0.18%	0.09%	0.14%	0.26%	0.34%	0.13%
STD.	0.0432	0.07	0.07	0.02	0.09	0.07	0.10	0.16	0.07	0.13	0.20	0.0009	0.0004
ACTUAL												0.8449	0.8551
% DIFFERENCE												3.91%	1.87%
MAXIMUM	5.047	332.50	140.42	220.00	376.33	477.64	353.90	261.51	258.62	258.27	227.37	0.880	0.872
MINIMUM	4.915	332.28	140.25	219.96	376.05	477.40	353.60	261.04	258.40	257.90	226.77	0.877	0.871
MAX-MIN	0.132	0.22	0.17	0.04	0.28	0.24	0.30	0.47	0.22	0.37	0.60	0.003	0.001

DATA C4: INCREASING NUMAVE - EQUILATERAL TRIANGLES

NUM- AVE.	ORIENT. I.D. ANGLE (DEG.)	CORNER COORDINATES						LENGTH OF SIDES			ASPECT	
		1 ROW	1 COL.	2 ROW	2 COL.	3 ROW	3 COL.	1 (B)	2	3	HEIGHT (H)	AVERAGE RATIO (A.R.) (H/B)
1	EQUIL.61.330	162.65	254.23	389.00	378.00	388.06	115.40	257.98	262.60	264.73	222.07	0.846
3	EQUIL.61.330	162.65	254.23	389.00	378.00	388.06	115.40	257.98	262.60	264.73	222.07	0.846
5	EQUIL.61.330	162.65	254.23	389.00	378.00	388.06	115.40	257.98	262.60	264.73	222.07	0.846
7	EQUIL.61.653	163.66	255.43	389.00	377.00	388.06	115.40	256.04	261.60	264.51	220.11	0.841
9	EQUIL.61.653	163.66	255.43	389.00	377.00	387.81	115.54	256.04	261.46	264.22	220.15	0.842
11	EQUIL.61.850	163.66	255.43	389.00	376.00	387.81	115.54	255.57	260.46	264.22	219.90	0.844
13	EQUIL.61.710	165.19	255.54	389.00	376.00	386.81	115.54	254.17	260.47	262.14	218.27	0.838
15	EQUIL.61.910	165.19	255.54	389.00	375.00	386.23	116.30	253.70	258.71	261.24	218.24	0.844
17	EQUIL.61.910	165.19	255.54	389.00	375.00	386.23	116.30	253.70	258.71	261.24	218.24	0.844
19	EQUIL.62.200	166.00	256.43	389.00	374.00	386.23	116.30	252.09	257.71	261.03	216.67	0.841
21	EQUIL.62.200	166.00	256.43	389.00	374.00	385.62	117.31	252.09	256.71	259.98	216.97	0.845
AVE.	61.824	164.58	255.56	389.00	375.78	386.98	115.96	254.60	259.83	262.59	218.96	0.8427
SPREAD	1.41%	2.04%	0.86%	0.00%	1.06%	0.63%	1.65%	2.31%	2.27%	1.81%	2.46%	0.91%
STD.	0.2611	1.126	0.609	0.000	1.315	0.899	0.608	1.853	1.855	1.722	1.632	0.0023
ACTUAL												0.8699
% DIFFERENCE												3.13%
MAXIMUM	62.200	166.00	256.43	389.00	378.00	388.06	117.31	257.98	262.60	264.73	222.07	0.846
MINIMUM	61.330	162.65	254.23	389.00	374.00	385.62	115.40	252.09	256.71	259.98	216.67	0.838
MAX-MIN	0.870	3.35	2.20	0.00	4.00	2.44	1.91	5.88	5.89	4.76	5.39	0.008
1	EQUIL.89.849	280.79	146.63	151.29	376.24	408.53	376.93	263.61	257.24	263.35	230.10	0.895
3	EQUIL.89.826	280.79	146.63	152.46	376.15	408.53	376.93	262.96	256.07	263.35	229.68	0.897
5	EQUIL.89.996	279.75	147.69	152.46	376.15	407.50	376.17	261.53	255.04	261.77	228.33	0.895
7	EQUIL.89.996	279.75	147.69	152.46	376.15	407.50	376.17	261.53	255.04	261.77	228.33	0.895
9	EQUIL.89.996	279.75	147.69	152.46	376.15	407.50	376.17	261.53	255.04	261.77	228.33	0.895
11	EQUIL.89.995	279.43	149.26	154.00	376.14	407.50	376.17	259.24	253.50	260.56	226.15	0.892
13	EQUIL.90.326	279.43	149.26	154.00	376.14	407.40	374.70	259.24	253.40	259.23	226.17	0.893
15	EQUIL.90.068	279.43	149.26	154.80	375.00	407.40	374.70	257.86	252.60	259.23	224.81	0.890
17	EQUIL.90.068	278.63	149.91	154.80	375.00	407.40	374.70	256.90	252.60	259.06	223.71	0.886
19	EQUIL.90.068	278.63	149.91	154.80	375.00	407.40	374.70	256.90	252.60	259.06	223.71	0.886
21	EQUIL.90.118	278.63	149.91	156.00	375.00	406.64	374.49	256.33	250.64	258.50	223.60	0.892
AVE.	90.070	279.27	148.95	153.98	375.64	407.36	375.33	259.01	253.39	260.10	225.91	0.8915
SPREAD	0.37%	0.40%	1.49%	2.30%	0.31%	0.21%	0.45%	2.01%	1.74%	1.26%	2.09%	1.08%
STD.	0.0999	0.471	0.932	1.203	0.569	0.259	0.754	2.014	1.399	1.283	1.941	0.0036
ACTUAL												0.8699
% DIFFERENCE												2.49%
MAXIMUM	90.326	279.75	149.91	156.00	376.15	407.50	376.17	261.53	255.04	261.77	228.33	0.895
MINIMUM	89.995	278.63	147.69	152.46	375.00	406.64	374.49	256.33	250.64	258.50	223.60	0.886
MAX-MIN	0.331	1.12	2.22	3.54	1.15	0.86	1.68	5.20	4.40	3.27	4.73	0.010
1	EQUIL.4.711	332.67	138.43	219.96	376.15	477.53	354.93	263.09	258.44	260.49	229.16	0.887
3	EQUIL.4.956	332.33	140.42	219.96	376.15	477.58	353.81	261.14	258.59	258.13	226.89	0.877
5	EQUIL.4.956	332.33	140.42	219.96	376.15	477.58	353.81	261.14	258.59	258.13	226.89	0.877
7	EQUIL.4.956	332.33	140.42	219.96	376.15	477.58	353.81	261.14	258.59	258.13	226.89	0.877
9	EQUIL.4.889	331.57	140.68	221.00	375.00	476.80	353.12	259.10	256.73	257.34	225.06	0.877
11	EQUIL.4.722	331.57	140.68	221.00	375.00	477.58	353.81	259.10	257.45	258.35	224.86	0.873
13	EQUIL.4.908	331.57	140.68	222.00	375.00	476.80	353.12	258.67	255.74	257.34	224.86	0.879
15	EQUIL.4.908	331.50	142.13	222.00	375.00	476.80	353.12	257.33	255.74	256.18	223.31	0.873
17	EQUIL.4.908	331.50	142.13	222.00	375.00	476.80	353.12	257.33	255.74	256.18	223.31	0.873
19	EQUIL.4.927	331.50	142.13	223.00	375.00	476.80	353.12	256.91	254.74	256.18	223.11	0.876
21	EQUIL.5.296	330.36	143.16	223.00	375.00	476.37	351.51	255.49	254.46	254.42	221.56	0.871
AVE.	4.9411	331.58	141.38	221.55	375.26	477.01	353.17	258.47	256.42	256.92	224.43	0.8752
SPREAD	11.62%	0.59%	1.94%	1.37%	0.31%	0.25%	0.65%	2.19%	1.61%	1.53%	2.37%	0.97%
STD.	0.1416	0.541	0.953	1.079	0.478	0.422	0.666	1.796	1.439	1.212	1.678	0.0026
ACTUAL												0.8449
% DIFFERENCE												3.59%
MAXIMUM	5.296	332.33	143.16	223.00	376.15	477.58	353.81	261.14	258.59	258.35	226.89	0.879
MINIMUM	4.722	330.36	140.42	219.96	375.00	476.37	351.51	255.49	254.46	254.42	221.56	0.871
MAX-MIN	0.574	1.97	2.74	3.04	1.15	1.21	2.30	5.65	4.13	3.93	5.33	0.009

DATA 01: CONSISTENCY TEST - 150SCLES TRIANGLE									
TEST	I.D.	ANGLE	CORNER COORDINATES			LENGTH OF SIDES			ASPECT
			ROW	COL	ROW	COL	ROW	COL	
1	2	3	1	2	1	2	1	2	A.N.
1	1505.	63.916	116.48	278.35	143.43	286.80	261.03	290.72	255.39
2	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
3	1505.	63.916	117.46	277.69	143.43	286.22	261.03	289.55	254.73
4	1505.	63.916	116.48	278.35	143.43	286.08	261.57	290.72	254.44
5	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
6	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
7	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
8	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
9	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
10	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
11	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
12	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
13	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
14	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
15	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
16	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
17	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
18	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
19	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
20	1505.	63.916	116.48	278.35	143.43	286.96	261.09	290.72	255.55
AVERAGE									
AVE.	63.916	116.48	277.74	278.35	143.43	286.91	261.04	290.81	255.50
STDEV.	0.342	1.842	0.428	0.312	0.251	0.001	0.921	0.381	1.151
ACTUAL	0.0852	0.604	0.476	0.581	0.163	0.000	0.697	0.163	0.404
% DIFFERENCE	2.611	1.151	1.0050	0.9987	0.0024	0.0031	0.0024	0.0031	0.0024
MINIMUM	63.916	117.46	278.35	274.22	143.43	288.46	261.57	291.25	257.23
MAXIMUM	63.916	115.33	277.18	273.00	143.43	285.83	260.57	289.55	254.30
MIN-MAX	0.219	2.13	1.17	1.22	1.00	2.63	1.00	1.70	2.93

DATA D3: INCREASING STLEN - ISOSCELES TRIANGLES

STLEN #	I.D. #	ORIENT. ANGLE (DEG.)	CORNER COORDINATES						LENGTH OF SIDES			ASPECT		
			1		2		3							
			ROW	COL.	ROW	COL.	ROW	COL.	1	2	3	HEIGHT (H)	RATIO (A.R.) (H/B)	AVERAGE A.R.
1	EQUIL.	4.115	123.50	104.50	128.50	174.00	186.50	135.50	69.68	69.62	70.21	60.98	0.876	0.872
2	-	-	-	-	-	-	-	-	-	-	-	-	-	-
3	ISOS.	64.065	117.00	278.50	373.00	403.00	372.67	143.67	284.67	259.33	289.04	253.42	0.977	0.987
4	ISOS.	64.103	116.71	278.57	373.00	403.00	372.75	143.50	284.90	259.50	289.48	253.64	0.977	0.987
5	ISOS.	64.064	116.75	278.38	373.00	403.00	372.80	143.40	284.95	259.60	289.45	253.67	0.977	0.987
6	ISOS.	64.090	116.57	278.43	373.00	403.00	372.87	143.48	285.09	259.52	289.66	253.84	0.978	0.988
8	ISOS.	64.083	116.54	278.37	373.00	403.00	372.91	143.48	285.14	259.52	289.69	253.90	0.978	0.988
10	ISOS.	64.079	116.52	278.34	373.00	403.00	372.92	143.39	285.17	259.61	289.75	253.91	0.978	0.988
15	ISOS.	64.084	116.44	278.33	373.00	403.00	372.96	143.45	285.25	259.55	289.82	254.02	0.979	0.989
20	ISOS.	64.072	116.50	278.29	373.05	403.02	372.97	143.44	285.26	259.58	289.76	254.03	0.979	0.988
AVE.		64.082	116.57	278.38	373.00	403.00	372.88	143.44	285.10	259.55	289.65	253.85	0.9780	0.9878
SPREAD		0.06%	0.27%	0.10%	0.01%	.00%	0.06%	0.08%	0.13%	0.04%	0.13%	0.15%	0.16%	0.17%
STD.		0.0116	0.105	0.085	0.017	0.007	0.076	0.039	0.130	0.040	0.131	0.144	0.0005	0.0005
ACTUAL													1.0050	0.9987
± DIFF.													2.68%	1.08%
MAXIMUM		64.103	116.75	278.57	373.05	403.02	372.97	143.50	285.26	259.61	289.82	254.03	0.979	0.989
MINIMUM		64.064	116.44	278.29	373.00	403.00	372.75	143.39	284.90	259.50	289.45	253.64	0.977	0.987
MAX-MIN		0.039	0.31	0.28	0.05	0.02	0.22	0.11	0.36	0.11	0.37	0.39	0.002	0.002
1	EQUIL.	36.948	126.00	133.00	170.00	191.50	195.00	128.00	73.20	68.24	69.18	64.76	0.949	0.916
2	ISOS.	90.449	324.00	121.00	195.00	384.00	450.00	382.00	292.93	255.01	289.82	263.73	1.034	1.027
3	ISOS.	90.411	323.40	121.20	195.00	384.50	450.33	382.67	292.94	255.34	290.65	263.66	1.033	1.028
4	ISOS.	90.355	323.25	121.25	195.00	384.33	450.50	382.75	292.68	255.50	290.82	263.32	1.031	1.027
5	ISOS.	90.312	323.33	121.33	195.00	384.25	450.43	382.86	292.57	255.43	290.78	263.22	1.030	1.027
6	ISOS.	90.300	323.26	121.35	194.96	384.21	450.52	382.87	292.50	255.56	290.84	263.11	1.030	1.026
8	ISOS.	90.277	323.26	121.39	194.96	384.15	450.52	382.91	292.41	255.56	290.84	263.01	1.029	1.026
10	ISOS.	90.265	323.26	121.42	194.96	384.12	450.52	382.94	292.36	255.56	290.84	262.95	1.029	1.026
15	ISOS.	90.249	323.24	121.44	194.97	384.07	450.48	382.96	292.28	255.51	290.83	262.88	1.029	1.026
20	ISOS.	90.243	323.26	121.46	194.98	384.05	450.51	382.97	292.25	255.53	290.83	262.84	1.029	1.026
AVE.		90.285	323.26	121.37	194.97	384.16	450.49	382.89	292.43	255.52	290.82	263.04	1.0294	1.0259
SPREAD		0.12%	0.03%	0.17%	0.02%	0.07%	0.02%	0.06%	0.15%	0.05%	0.02%	0.18%	0.19%	0.11%
STD.		0.0365	0.027	0.068	0.017	0.093	0.031	0.071	0.144	0.044	0.020	0.165	0.0007	0.0004
ACTUAL													1.0050	0.9987
± DIFF.													2.43%	2.73%
MAXIMUM		90.355	323.33	121.46	195.00	384.33	450.52	382.97	292.68	255.56	290.84	263.32	1.031	1.027
MINIMUM		90.243	323.24	121.25	194.96	384.05	450.43	382.75	292.25	255.43	290.78	262.84	1.029	1.026
MAX-MIN		0.112	0.09	0.21	0.04	0.28	0.09	0.22	0.43	0.13	0.06	0.48	0.002	0.001
1	EQUIL.	68.565	161.00	78.50	107.00	123.00	174.00	149.50	69.97	72.05	72.18	59.99	0.833	0.850
2	-	-	-	-	-	-	-	-	-	-	-	-	-	-
3	ISOS.	6.177	198.00	88.00	228.80	372.60	448.00	234.50	286.26	259.08	289.76	255.28	0.985	0.993
4	ISOS.	6.196	198.00	88.00	228.91	372.73	447.82	234.55	286.40	258.87	289.63	255.49	0.987	0.994
5	ISOS.	6.177	198.00	88.00	228.80	372.60	447.73	234.55	286.26	258.82	289.55	255.34	0.987	0.994
6	ISOS.	6.189	198.00	88.00	228.87	372.68	447.69	234.41	286.35	258.85	289.45	255.43	0.987	0.994
8	ISOS.	6.195	198.00	88.00	228.90	372.63	447.63	234.45	286.30	258.72	289.42	255.41	0.987	0.994
10	ISOS.	6.221	197.88	87.96	228.91	372.60	447.60	234.40	286.33	258.70	289.49	255.44	0.987	0.994
15	ISOS.	6.214	197.93	87.97	228.92	372.62	447.57	234.36	286.33	258.70	289.40	255.45	0.987	0.994
20	ISOS.	6.225	197.89	87.94	228.94	372.61	447.55	234.35	286.36	258.66	289.42	255.49	0.988	0.994
AVE.		6.2024	197.95	87.981	228.89	372.63	447.65	234.43	286.33	258.75	289.48	255.43	0.9871	0.9939
SPREAD		0.77%	0.06%	0.07%	0.06%	0.03%	0.06%	0.09%	0.05%	0.08%	0.08%	0.06%	0.12%	0.08%
STD.		0.0165	0.051	0.023	0.043	0.045	0.089	0.077	0.041	0.078	0.079	0.047	0.0003	0.0002
ACTUAL													0.9791	0.9934
± DIFF.													0.82%	0.06%
MAXIMUM		6.225	198.00	88.00	228.94	372.73	447.82	234.55	286.40	258.87	289.63	255.49	0.988	0.994
MINIMUM		6.177	197.88	87.94	228.80	372.60	447.55	234.35	286.26	258.66	289.40	255.34	0.987	0.994
MAX-MIN		0.048	0.12	0.06	0.14	0.13	0.27	0.20	0.14	0.21	0.24	0.15	0.001	0.001

DATA D4: INCREASING NUMAVE - ISOSCELES TRIANGLES

NUM- AVE.	I.D. AVE.	ORIENT. (DEG.)	CORNER COORDINATES						LENGTH OF SIDES			HEIGHT (H)	ASPECT RATIO (A.R.) (H/B)	AVERAGE A.R.
			1 ROW	1 COL.	2 ROW	2 COL.	3 ROW	3 COL.	1 (B)	2 (B)	3 (B)			
1	ISOS.	63.774	115.57	277.18	373.00	404.00	374.00	143.43	286.97	260.57	290.99	255.69	0.981	0.990
3	ISOS.	63.902	116.54	278.37	373.00	404.00	372.91	143.48	285.58	260.52	289.69	254.14	0.976	0.984
5	ISOS.	64.083	116.54	278.37	373.00	403.00	372.91	143.48	285.14	259.52	289.69	253.90	0.978	0.988
7	ISOS.	64.083	116.54	278.37	373.00	403.00	372.91	143.48	285.14	259.52	289.69	253.90	0.978	0.988
9	ISOS.	63.968	118.30	278.60	373.00	403.00	372.91	143.48	283.46	259.52	288.24	252.01	0.971	0.981
11	ISOS.	64.150	118.30	278.60	373.00	402.00	372.14	144.35	283.02	257.65	287.15	252.00	0.978	0.987
13	ISOS.	64.150	118.30	278.60	373.00	402.00	372.14	144.35	283.02	257.65	287.15	252.00	0.978	0.987
15	ISOS.	64.267	118.87	279.52	373.00	402.00	372.14	144.35	282.11	257.65	287.08	250.97	0.974	0.985
17	ISOS.	64.450	118.87	279.52	373.00	401.00	372.14	144.35	281.67	256.65	287.08	250.74	0.977	0.989
19	ISOS.	64.450	118.87	279.52	373.00	401.00	370.58	144.58	281.67	256.43	285.60	250.80	0.978	0.987
21	ISOS.	64.517	119.87	280.84	373.00	400.00	370.58	144.58	279.77	255.43	285.35	248.92	0.975	0.987
AVE.		64.235	118.27	279.10	373.00	401.89	372.05	144.11	282.78	257.78	287.45	251.69	0.9763	0.9865
SPREAD		0.85%	2.82%	0.88%	0.00%	0.75%	0.63%	0.76%	1.90%	1.59%	1.51%	1.98%	0.75%	0.75%
STD.		0.1840	1.03	0.77	0.00	0.99	0.85	0.45	1.62	1.40	1.45	1.49	0.0024	0.0021
ACTUAL													1.0050	0.9987
% DIFFERENCE													2.85%	1.22%
MAXIMUM		64.517	119.87	280.84	373.00	403.00	372.91	144.58	285.14	259.52	289.69	253.90	0.978	0.989
MINIMUM		63.968	116.54	278.37	373.00	400.00	370.58	143.48	279.77	255.43	285.35	248.92	0.971	0.981
MAX-MIN		0.549	3.33	2.47	0.00	3.00	2.33	1.10	5.36	4.09	4.35	4.98	0.007	0.007
1	ISOS.	90.033	323.26	121.39	194.96	384.15	450.43	384.00	292.41	255.47	291.79	263.04	1.030	1.028
3	ISOS.	90.277	323.26	121.39	194.96	384.15	450.52	382.91	292.41	255.56	290.84	263.01	1.029	1.026
5	ISOS.	90.277	323.26	121.39	194.96	384.15	450.52	382.91	292.41	255.56	290.84	263.01	1.029	1.026
7	ISOS.	90.277	322.64	121.52	196.00	384.15	450.52	382.91	291.57	254.52	290.99	262.33	1.031	1.029
9	ISOS.	90.440	322.64	121.52	196.00	384.15	449.51	382.20	291.57	253.52	289.91	262.57	1.036	1.032
11	ISOS.	90.440	322.64	121.52	196.00	384.15	449.51	382.20	291.57	253.52	289.91	262.57	1.036	1.032
13	ISOS.	90.180	322.35	123.18	196.80	383.00	449.51	382.20	288.56	252.71	288.55	259.43	1.027	1.027
15	ISOS.	90.180	322.35	123.18	196.80	383.00	449.51	382.20	288.56	252.71	288.55	259.43	1.027	1.027
17	ISOS.	90.180	322.35	123.18	196.80	383.00	449.51	382.20	288.56	252.71	288.55	259.43	1.027	1.027
19	ISOS.	90.540	322.57	124.00	198.00	383.00	449.36	380.64	287.40	251.37	286.25	258.46	1.028	1.026
21	ISOS.	90.54	322.57	124.00	198.00	383.00	449.36	380.64	287.40	251.37	286.25	258.46	1.028	1.026
AVE.		90.339	322.60	122.61	196.60	383.51	449.70	382.01	289.73	253.11	288.87	260.63	1.0297	1.0278
SPREAD		0.40%	0.28%	2.13%	1.55%	0.30%	0.26%	0.59%	1.73%	1.66%	1.64%	1.75%	0.89%	0.63%
STD.		0.1432	0.26	1.05	0.93	0.57	0.44	0.79	1.89	1.29	1.65	1.82	0.0034	0.0025
ACTUAL													1.0050	0.9987
% DIFFERENCE													2.46%	2.92%
MAXIMUM		90.54	323.26	124.00	198.00	384.15	450.52	382.91	292.41	255.56	290.99	263.01	1.036	1.032
MINIMUM		90.18	322.35	121.39	194.96	383.00	449.36	380.64	287.40	251.37	286.25	258.46	1.027	1.026
MAX-MIN		0.36	0.91	2.61	3.04	1.15	1.16	2.27	5.01	4.19	4.74	4.55	0.009	0.006
1	ISOS.	6.173	198.00	87.00	228.90	372.63	447.63	234.45	287.30	258.72	289.93	256.53	0.992	0.997
3	ISOS.	6.173	198.00	87.00	228.90	372.63	448.71	234.41	287.30	259.66	290.84	256.29	0.987	0.995
5	ISOS.	6.195	198.00	88.00	228.90	372.63	447.63	234.45	286.30	258.72	289.42	255.41	0.987	0.994
7	ISOS.	6.500	198.00	89.00	230.30	372.50	447.63	234.45	285.33	257.47	288.91	254.64	0.989	0.997
9	ISOS.	6.500	198.00	89.00	230.30	372.50	447.63	234.45	285.33	257.47	288.91	254.64	0.989	0.997
11	ISOS.	6.500	198.00	89.00	230.30	372.50	447.19	233.68	285.33	257.51	288.15	254.63	0.989	0.995
13	ISOS.	6.551	197.86	90.04	230.30	372.50	447.19	233.68	284.32	257.51	287.75	253.49	0.984	0.992
15	ISOS.	6.655	197.86	90.04	230.73	371.73	447.19	233.68	283.60	256.73	287.75	252.89	0.985	0.994
17	ISOS.	6.655	197.86	90.04	230.73	371.73	447.19	233.68	283.60	256.73	287.75	252.89	0.985	0.994
19	ISOS.	6.678	197.88	91.13	230.73	371.73	445.88	233.63	282.52	255.66	286.02	251.94	0.985	0.993
21	ISOS.	6.867	197.88	91.13	231.56	370.82	445.88	233.63	281.71	254.47	286.02	251.34	0.988	0.997
AVE.		6.5667	197.93	89.71	230.43	372.07	447.05	233.93	284.23	256.92	287.85	253.54	0.9868	0.9947
SPREAD		10.23%	0.07%	3.49%	1.15%	0.49%	0.39%	0.35%	1.62%	1.66%	1.18%	1.61%	0.47%	0.54%
STD.		0.1731	0.07	0.99	0.66	0.57	0.65	0.37	1.42	1.17	1.13	1.31	0.0017	0.0017
ACTUAL													0.9791	0.9934
% DIFFERENCE													0.79%	0.14%
MAXIMUM		6.867	198.00	91.13	231.56	372.63	447.63	234.45	286.30	258.72	289.42	255.41	0.989	0.997
MINIMUM		6.195	197.86	88.00	228.90	370.82	445.88	233.63	281.71	254.47	286.02	251.34	0.984	0.992
MAX-MIN		0.672	0.14	3.13	2.66	1.81	1.75	0.82	4.59	4.25	3.39	4.07	0.005	0.005

DATA A2: CONSISTENCY OF ALTERNATE POSITIONS - SQUARES

TEST	I.D.	LENGTH OF SIDES				ASPECT AVERAGE	
		1	2	3	4	RATIO	A.R.
		A	B	C	D	(A.R.) (A+C)/ (A/B) (B/D)	
1	SQ.	255.01	261.02	257.01	257.00	0.977	0.988
2	SQ.	262.00	259.05	259.02	255.01	1.011	1.014
3	SQ.	257.82	259.03	254.05	262.23	0.995	0.982
4	SQ.	256.00	254.85	261.03	257.05	1.005	1.010
5	SQ.	255.99	258.30	257.99	253.52	0.991	1.004
6	SQ.	260.85	256.21	255.23	253.89	1.018	1.012
7	SQ.	257.86	256.14	256.37	258.82	1.007	0.999
8	SQ.	257.23	255.14	262.31	258.25	1.008	1.012
AVE.		257.84	257.46	257.87	256.97	1.0015	1.0025
SPREAD		2.71%	2.40%	3.20%	3.39%	4.11%	3.15%
STD.		2.279	2.058	2.636	2.692	0.0123	0.0111
ACTUAL						1.0000	1.0013
% DIFFERENCE						0.15%	0.13%
MAXIMUM		262	261.02	262.31	262.23	1.018	1.014
MINIMUM		255.01	254.85	254.05	253.52	0.977	0.982
MAX-MIN		6.99	6.17	8.26	8.71	0.041	0.032

DATA B2: CONSISTENCY OF ALTERNATE POSITIONS - RECTANGLES

TEST	I.D.	LENGTH OF SIDES				ASPECT AVERAGE	
		1	2	3	4	RATIO	A.R.
		A	B	C	D	(A.R.) (A+C)/ (A/B) (B/D)	
1	RECT.	91.45	143.30	93.13	144.56	0.638	0.641
2	RECT.	93.64	140.11	93.29	139.29	0.668	0.669
3	RECT.	93.01	140.03	95.01	140.09	0.664	0.671
4	RECT.	93.41	146.57	96.05	148.39	0.637	0.642
AVE.		92.877	142.50	94.37	143.08	0.6520	0.6559
SPREAD		2.36%	4.59%	3.09%	6.36%	4.76%	4.57%
STD.		0.8544	2.6934	1.2182	3.6637	0.0143	0.0142
ACTUAL						0.6636	0.6642
% DIFFERENCE						1.75%	1.24%
MAXIMUM		93.64	146.57	96.05	148.39	0.668	0.671
MINIMUM		91.45	140.03	93.13	139.29	0.637	0.641
MAX-MIN		2.19	6.54	2.92	9.1	0.031	0.030

DATA C2: CONSISTENCY OF ALTERNATE POSITIONS
EQUILATERAL TRIANGLES

TEST #	I.D.	LENGTH OF SIDES			HEIGHT H	ASPECT RATIO (H/B)	AVERAGE A.R.
		1	2	3			
		B					
1	EQ.	265.41	260.43	263.86	231.27	0.888	0.885
2	EQ.	261.61	261.28	261.92	226.65	0.867	0.868
3	EQ.	263.62	257.24	263.36	230.11	0.895	0.894
4	EQ.	257.98	262.61	264.73	222.06	0.846	0.861
5	EQ.	262.83	265.84	257.54	226.74	0.853	0.841
6	EQ.	258.41	264.03	264.02	222.14	0.841	0.854
7	EQ.	259.64	263.14	259.36	223.83	0.851	0.850
8	EQ.	260.49	258.69	263.10	226.10	0.874	0.880
AVE.		261.24	261.65	262.23	226.11	0.8643	0.8665
SPREAD		2.84%	3.29%	2.74%	4.07%	6.15%	6.07%
STD.		2.436	2.650	2.358	3.175	0.0186	0.0172
ACTUAL						0.8574	0.8660
% DIFFERENCE						0.81%	0.06%
MAXIMUM		265.41	265.84	264.73	231.27	0.8945	0.8939
MINIMUM		257.98	257.24	257.54	222.06	0.8413	0.8413
MAX-MIN		7.43	8.6	7.19	9.2069	0.0531	0.0525

DATA D2: CONSISTENCY OF ALTERNATE POSITIONS
ISOSCELES TRIANGLES

TEST #	I.D.	LENGTH OF SIDES			HEIGHT H	ASPECT RATIO (H/B)	AVERAGE A.R.
		1	2	3			
		B					
1	ISOS.	297.66	261.87	293.06	267.32	1.021	1.011
2	ISOS.	290.80	260.57	289.80	259.98	0.998	0.996
3	ISOS.	293.92	255.47	293.08	264.71	1.036	1.034
4	ISOS.	287.16	261.00	290.99	255.79	0.980	0.988
5	ISOS.	283.74	261.10	293.74	251.92	0.965	0.986
6	ISOS.	289.94	262.19	289.24	258.61	0.986	0.985
7	ISOS.	286.99	259.58	295.90	255.96	0.986	1.005
8	ISOS.	287.30	259.67	290.84	256.29	0.987	0.995
AVE.		289.68	260.18	292.08	258.82	0.9948	1.0000
SPREAD		4.81%	2.58%	2.28%	5.95%	7.17%	4.95%
STD.		4.146	1.980	2.101	4.738	0.0216	0.0155
ACTUAL						0.9921	0.9961
% DIFFERENCE						0.28%	0.40%
MAXIMUM		297.66	262.19	295.9	267.31	1.0361	1.0343
MINIMUM		283.74	255.47	289.24	251.92	0.9648	0.9848
MAX-MIN		13.92	6.72	6.66	15.392	0.0713	0.0495

APPENDIX B

TSTLOAD LISTING

```

D Line# 1      7      Microsoft FORTRAN77 V3.30 March 1985
1 $title:'Program_Get_Image'
2 $storage:2
3 $large
4      integer*2  window(64,512),bordr(2000),bordc(2000),ibuff(512)
5      integer*2  numedge,i,irow,j,icol,flag9,frow,fcoll,dct,nxtddct
6      integer*2  num8,ac,ar,toprow,botrow,nxttop,nxtbot
7      integer*2  flag10,icheck
8      character  fname*20
9      common /imdata/window
10
11 c ..... Initialize arrays.
12 c      Edge point coordinates will be stored in BORDR,BORDC (row,
13 c      col, resp.)
14 c      The image from the DT IRIS system is accessed in windows of
15 c      64 rows by 512 columns at a time, which is stored in
16 c      WINDOW(64,512).
17 c      IBUFF(512) is used to store one row of the image at a time.
18
19      do 5 i=1,2000
1 20      bordr(i)=0
1 21      bordc(i)=0
1 22      5 continue
23      do 6 i=1,512
1 24      ibuff(i)=120
1 25      6 continue
26      do 7 i=1,64
1 27      do 8 j=1,512
2 28      window(i,j)=120
2 29      8 continue
1 30      7 continue
31      flag10=0
32
33 c ..... Initialize DT IRIS system.
34 c      Set synchronicity to external (1): issync(1).
35 c      Set input and output buffers to buffer 0: isinfr(0),
36 c      isotfr(0).
37 c      Set input LUT to 6 (threshold at 127): isints(6).
38 c      Set output LUT to 0 (identity): isouts(0).
39 c      Clear buffer 0 for reception of image: isfclr(0)
40
41      call isinit()
42      call issync(1)
43      call isinfr(0)
44      call isotfr(0)
45      call isints(6)
46      call isouts(0)
47      call isfclr(0)
48
49 c ..... Obtain image using pass-thru mode:ispass().
50 c      Freeze the image and store in buffer 0: isfrez()
51
52      call ispass()
53      call isfrez()
54
55 c ..... Check image to see if obtained correctly.
56

```

```

D Line# 1      7
57      do 9 i=1,5
1 58          call isgetp(0,i,0,512,ibuff)
1 59          icheck=icheck+ibuff(1)
1 60      9 continue
61          if(icheck.ne.1275)then
62              flag10=4
63              goto 300
64          endif
65
66 c ..... Locate first edge point on object.
67
68          frow=520
69          fcol=520
70          call frstpt(ibuff,frow,fcol)
71          if(frow.gt.515)then
72 c              no object in scene
73              flag10=1
74              goto 300
75          endif
76          write(*,'(1x,a,i4,i4/)') 'frow,fcol=',frow,fcol
77
78 c ..... Find next points on edge by reading in a window(64,512) of the
79 c          global image(512,512) in DT IRIS buffer 0.
80 c          NUMEDGE: number of edge points found.
81 c          DCT: direction taken to find current edge point.
82 c          NXTDCT: direction to start looking for next edge point.
83 c          TOPROW: top row of window(64,512) in global coordinates.
84 c          BOTROW: bottom row of window in global coords.
85 c          NXTTOP: updated top row of window.
86 c          NXTBOT: updated bottom row of window.
87 c          AR,AC: row,col coords. of edge pt. in window coord. system.
88 c          IIRW,IICOL: row,col coords. of pt. in global coord. sys.
89
90          numedge=1
91          bordr(1)=frow
92          bordc(1)=fcol
93          flag9=0
94          dct=2
95          nxtdct=7
96          toprow=frow-1
97          botrow=toprow+63
98          ar=2
99          iirw=0
100         iicol=0
101         if(botrow.gt.511)then
102             botrow=511
103             toprow=448
104             ar=frow-toprow+1
105         endif
106         nxttop=0
107         nxtbot=0
108         ac=fcol+1
109         call getim(toprow)
110     200 num8=0
111
112 c ..... If row accessed is on top or bottom boundary of window,

```

```
D Line# 1      7
113 c          update accordingly.
114
115          if((irow.eq.1).or.(irow.eq.64))then
116              call upwind(irow,icol,toprow,nxttop,botrow,nxtbot,ar,ac)
117          endif
118
119 c ..... routine determines next direction to take to start
120 c ..... looking for next point on contour of object
121
122          call nxtdir(flag9,ar,ac,irow,icol,nxt dct,num8,dct)
123
124          if((num8.ge.1).and.(flag9.eq.0))then
125              if(num8.eq.1)then
126                  call dcti(flag9,ar,ac,irow,icol,nxt dct,num8)
127                  if(num8.gt.1)then
128                      flag10=2
129                      goto 300
130                  endif
131              endif
132          endif
133
134 c ..... If first point found again, end iteration.
135 c          Otherwise store coords. in BORDR,BORDC arrays, and
136 c          update variables.
137
138          if((irow+toprow-1.eq.frow).and.(icol-1.eq.fcol))then
139              nxtdct=0
140          else
141              numedge=numedge+1
142              bordr(numedge)=irow+toprow-1
143              bordc(numedge)=icol-1
144              dct=nxt dct
145              ar=irow
146              ac=icol
147              flag9=0
148          endif
149          if(nxt dct.ne.0)then
150              goto 200
151          endif
152
153 c ..... If this point in the program has been reached, the
154 c          edge of the object has been successfully located.
155
156          flag10=3
157
158          300 if(flag10.eq.1)then
159              write(*,'(1x,a/)') 'No object in scene.'
160          endif
161          if(flag10.eq.2)then
162              write(*,'(1x,a/)') 'CALL DCT8 done twice: no pt. found.'
163          endif
164          if(flag10.eq.3)then
165              write(*,'(1x,a/)') 'Successfully completed entire
166              & algorithm.'
167          endif
168          if(flag10.eq.4)then
```

Program_Get_Image

Page 4

01-23-89

09:06:13

```

D Line# 1      7      Microsoft FORTRAN77 V3.30 March 1985
169      write(*,'(1x,a/)' ) 'Image & window not loaded correctly.'
170      endif
171      call isend()
172
173 c ..... Have all border points, write to file on disk.
174
175      call savpts(bordr,bordc,numedg)
176
177      stop
178      end

```

Name	Type	Offset	P	Class
AC	INTEGER*2	50		
AR	INTEGER*2	40		
BORDC	INTEGER*2	0		LARGE
BORDR	INTEGER*2	0		LARGE
BOTROW	INTEGER*2	38		
DCT	INTEGER*2	32		
FCOL	INTEGER*2	26		
FLAG10	INTEGER*2	20		
FLAG9	INTEGER*2	30		
FNAME	CHAR*20	*****		
FROW	INTEGER*2	24		
I	INTEGER*2	16		
IBUFF	INTEGER*2	4000		LARGE
ICHECK	INTEGER*2	22		
IICOL	INTEGER*2	44		
IROW	INTEGER*2	42		
J	INTEGER*2	18		
NUM8	INTEGER*2	52		
NUMEDG	INTEGER*2	28		
NXTBOT	INTEGER*2	48		
NXTDCT	INTEGER*2	34		
NXTTOP	INTEGER*2	46		
TOPROW	INTEGER*2	36		
WINDOW	INTEGER*2	0		/IMDATA/

```

179
180      subroutine frstpt(ibuff,frow,fcoll)
181      integer*2 ibuff(512),frow,fcoll
182      integer*2 aa,bb,flag7,flag8
183
184      flag8=0
185      flag7=0
186      do 40 aa=2,512
1 187      if(flag7.eq.0) then
1 188      call isgetp(0,aa,0,512,ibuff)
1 189      do 30 bb=5,511
2 190      if(flag8.eq.0) then
2 191      if(ibuff(bb).eq.0) then
2 192      frow=aa
2 193      fcol=bb-1
2 194      flag7=1
2 195      flag8=1

```

```

D Line# 1      7
2 196          endif
2 197          endif
2 198      30    continue
1 199          endif
1 200      40    continue
201          return
202          end

```

Name	Type	Offset	P	Class
AA	INTEGER*2	58		
BB	INTEGER*2	60		
FCOL	INTEGER*2	8	*	
FLAG7	INTEGER*2	56		
FLAG8	INTEGER*2	54		
FROM	INTEGER*2	4	*	
IBUFF	INTEGER*2	0	*	LARGE

```

203
204      subroutine upwind(irow,icol,toprow,nxttop,botrow,nxtbot,ar,ac)
205          integer*2 window(64,512),irow,icol,toprow,botrow
206          integer*2 nxttop,nxtbot,ar,ac
207          common /imdata/window
208
209      c ..... Update window(64,512).
210
211          if(irow.eq.1)then
212              nxtbot=toprow+1
213              nxttop=nxtbot-63
214              if(nxttop.lt.0)then
215                  nxttop=0
216                  nxtbot=63
217                  ar=toprow+1
218              else
219                  ar=63
220              endif
221          else
222              if(irow.eq.64)then
223                  nxttop=botrow-1
224                  nxtbot=nxttop+63
225                  if(nxtbot.gt.511)then
226                      nxtbot=511
227                      nxttop=448
228                      ar=botrow-nxttop+1
229                  else
230                      ar=2
231                  endif
232              endif
233          endif
234          ac=icol
235          irow=ar
236          toprow=nxttop
237          botrow=nxtbot
238          call getim(toprow)
239      return

```

Program_Get_Image

Page 6

01-23-89

09:06:13

D Line# 1 7
240 end

Microsoft FORTRAN77 V3.30 March 1985

Name	Type	Offset	P	Class
AC	INTEGER*2	28	*	
AR	INTEGER*2	24	*	
BOTROW	INTEGER*2	16	*	
IICOL	INTEGER*2	4	*	
IROW	INTEGER*2	0	*	
NXTBOT	INTEGER*2	20	*	
NXTTOP	INTEGER*2	12	*	
TOPROW	INTEGER*2	8	*	
WINDOW	INTEGER*2	0		/IMDATA/

```

241
242      subroutine nxtdir(flag9,ar,ac,irow,icol,nxtdct,num8,dct)
243         integer*2 flag9,ar,ac,irow,icol,nxtdct,num8,dct
244         integer*2 window(64,512)
245         common /imdata/window
246
247         if(dct.eq.1) then
248             nxtdct=6
249             call dctvi(flag9,ar,ac,irow,icol,nxtdct,num8)
250         else
251             if(dct.eq.2) then
252                 nxtdct=7
253                 call dctvii(flag9,ar,ac,irow,icol,nxtdct,num8)
254             else
255                 if(dct.eq.3) then
256                     nxtdct=8
257                     call dct8(flag9,ar,ac,irow,icol,nxtdct,num8)
258                 else
259                     if(dct.eq.4) then
260                         nxtdct=1
261                         call dcti(flag9,ar,ac,irow,icol,nxtdct,num8)
262                     else
263                         if(dct.eq.5) then
264                             nxtdct=2
265                             call dctii(flag9,ar,ac,irow,icol,nxtdct,num8)
266                         else
267                             if(dct.eq.6) then
268                                 nxtdct=3
269                                 call dctiii(flag9,ar,ac,irow,icol,nxtdct,num8)
270                             else
271                                 if(dct.eq.7) then
272                                     nxtdct=4
273                                     call dctiv(flag9,ar,ac,irow,icol,nxtdct,num8)
274                                 else
275                                     if(dct.eq.8) then
276                                         nxtdct=5
277                                         call dctv(flag9,ar,ac,irow,icol,nxtdct,num8)
278                                     endif
279                                 endif
280                             endif
281                         endif

```

```

D Line# 1      7
      282      endif
      283      endif
      284      endif
      285      endif
      286
      287      return
      288      end

```

Name	Type	Offset	P	Class
AC	INTEGER*2	8	*	
AR	INTEGER*2	4	*	
DCT	INTEGER*2	28	*	
FLAG9	INTEGER*2	0	*	
IICOL	INTEGER*2	16	*	
IIROW	INTEGER*2	12	*	
NUM8	INTEGER*2	24	*	
NXTDCT	INTEGER*2	20	*	
WINDOW	INTEGER*2	0		/IMDATA/

```

289
290      subroutine dcti(flag9,ar,ac,irow,icol,nxt dct,num8)
291      integer*2 window(64,512),flag9,ar,ac,irow,icol,nxt dct,num8
292      common /imdata/window
293
294      if(window(ar-1,ac+1).eq.0)then
295          nxt dct=1
296          flag9=1
297          irow=ar-1
298          icol=ac+1
299      endif
300      if(flag9.eq.0)then
301          call dctii(flag9,ar,ac,irow,icol,nxt dct,num8)
302      endif
303      return
304      end

```

Name	Type	Offset	P	Class
AC	INTEGER*2	8	*	
AR	INTEGER*2	4	*	
FLAG9	INTEGER*2	0	*	
IICOL	INTEGER*2	16	*	
IIROW	INTEGER*2	12	*	
NUM8	INTEGER*2	24	*	
NXTDCT	INTEGER*2	20	*	
WINDOW	INTEGER*2	0		/IMDATA/

```

305
306      subroutine dotii(flag9,ar,ac,irow,icol,nxt dct,num8)
307      integer*2 window(64,512),flag9,ar,ac,irow,icol,nxt dct,num8
308      common /imdata/window
309
310      if(window(ar,ac+1).eq.0)then

```

D Line# 1 7

```

311      nxdct=2
312      flag9=1
313      irow=ar
314      icol=ac+1
315      endif
316      if(flag9.eq.0)then
317          call dctiii(flag9,ar,ac,irow,icol,nxdct,num8)
318      endif
319      return
320      end

```

Name	Type	Offset	P	Class
AC	INTEGER*2	8	*	
AR	INTEGER*2	4	*	
FLAG9	INTEGER*2	0	*	
IICOL	INTEGER*2	16	*	
IROW	INTEGER*2	12	*	
NUM8	INTEGER*2	24	*	
NXTDCT	INTEGER*2	20	*	
WINDOW	INTEGER*2	0		/IMDATA/

```

321
322      subroutine dctiii(flag9,ar,ac,irow,icol,nxdct,num8)
323          integer*2 window(64,512),flag9,ar,ac,irow,icol,nxdct,num8
324          common /imdata/window
325
326          if(window(ar+1,ac+1).eq.0)then
327              nxdct=3
328              flag9=1
329              irow=ar+1
330              icol=ac+1
331          endif
332          if(flag9.eq.0)then
333              call dctiv(flag9,ar,ac,irow,icol,nxdct,num8)
334          endif
335          return
336      end

```

Name	Type	Offset	P	Class
AC	INTEGER*2	8	*	
AR	INTEGER*2	4	*	
FLAG9	INTEGER*2	0	*	
IICOL	INTEGER*2	16	*	
IROW	INTEGER*2	12	*	
NUM8	INTEGER*2	24	*	
NXTDCT	INTEGER*2	20	*	
WINDOW	INTEGER*2	0		/IMDATA/

```

337
338      subroutine dctiv(flag9,ar,ac,irow,icol,nxdct,num8)
339          integer*2 window(64,512),flag9,ar,ac,irow,icol,nxdct,num8
340          common /imdata/window

```

```

D Line# 1      7
341
342      if(window(ar+1,ac).eq.0)then
343          nxdct=4
344          flag9=1
345          irow=ar+1
346          icol=ac
347      endif
348      if(flag9.eq.0)then
349          call dctv(flag9,ar,ac,irow,icol,nxdct,num8)
350      endif
351      return
352      end

```

Name	Type	Offset	P	Class
AC	INTEGER*2	8	*	
AR	INTEGER*2	4	*	
FLAG9	INTEGER*2	0	*	
IICOL	INTEGER*2	16	*	
IROW	INTEGER*2	12	*	
NUM8	INTEGER*2	24	*	
NXTDCT	INTEGER*2	20	*	
WINDOW	INTEGER*2	0		/IMDATA/

```

353
354      subroutine dctv(flag9,ar,ac,irow,icol,nxdct,num8)
355          integer*2 window(64,512),flag9,ar,ac,irow,icol,nxdct,num8
356          common /imdata/window
357
358          if(window(ar+1,ac-1).eq.0)then
359              nxdct=5
360              flag9=1
361              irow=ar+1
362              icol=ac-1
363          endif
364          if(flag9.eq.0)then
365              call dctvi(flag9,ar,ac,irow,icol,nxdct,num8)
366          endif
367          return
368      end

```

Name	Type	Offset	P	Class
AC	INTEGER*2	8	*	
AR	INTEGER*2	4	*	
FLAG9	INTEGER*2	0	*	
IICOL	INTEGER*2	16	*	
IROW	INTEGER*2	12	*	
NUM8	INTEGER*2	24	*	
NXTDCT	INTEGER*2	20	*	
WINDOW	INTEGER*2	0		/IMDATA/

```

369
370      subroutine dctvi(flag9,ar,ac,irow,icol,nxdct,num8)

```

D Line# 1 7 Microsoft FORTRAN77 V3.30 March 1985

```

371 integer*2 window(64,512),flag9,ar,ac,irow,icol,nxt dct,num8
372 common /imdata/window
373
374 if(window(ar,ac-1).eq.0)then
375     nxt dct=6
376     flag9=1
377     irow=ar
378     icol=ac-1
379 endif
380 if(flag9.eq.0)then
381     call dctvii(flag9,ar,ac,irow,icol,nxt dct,num8)
382 endif
383 return
384 end

```

Name	Type	Offset	P	Class
AC	INTEGER*2	8	*	
AR	INTEGER*2	4	*	
FLAG9	INTEGER*2	0	*	
IICOL	INTEGER*2	16	*	
IROW	INTEGER*2	12	*	
NUM8	INTEGER*2	24	*	
NXTDCT	INTEGER*2	20	*	
WINDOW	INTEGER*2	0		/IMDATA/

```

385
386 subroutine dctvii(flag9,ar,ac,irow,icol,nxt dct,num8)
387 integer*2 window(64,512),flag9,ar,ac,irow,icol,nxt dct,num8
388 common /imdata/window
389
390 if(window(ar-1,ac-1).eq.0)then
391     nxt dct=7
392     flag9=1
393     irow=ar-1
394     icol=ac-1
395 endif
396 if(flag9.eq.0)then
397     call dct8(flag9,ar,ac,irow,icol,nxt dct,num8)
398 endif
399 return
400 end

```

Name	Type	Offset	P	Class
AC	INTEGER*2	8	*	
AR	INTEGER*2	4	*	
FLAG9	INTEGER*2	0	*	
IICOL	INTEGER*2	16	*	
IROW	INTEGER*2	12	*	
NUM8	INTEGER*2	24	*	
NXTDCT	INTEGER*2	20	*	
WINDOW	INTEGER*2	0		/IMDATA/

```

401
402 subroutine dct8(flag9,ar,ac,irow,icol,nxt dct,num8)

```

Program_Get_Image

Page 11

01-23-89

09:06:13

D Line# 1 7 Microsoft FORTRAN77 V3.30 March 1985
 403 integer*2 window(64,512),flag9,irow,icol,ar,ac,nxt dct,num8
 404 common /imdata/window
 405
 406 if(window(ar-1,ac).eq.0)then
 407 nxdct=8
 408 flag9=1
 409 irow=ar-1
 410 icol=ac
 411 endif
 412 if(flag9.eq.0)then
 413 num8 = num8 + 1
 414 endif
 415 return
 416 end

Name	Type	Offset	P	Class
AC	INTEGER*2	8	*	
AR	INTEGER*2	4	*	
FLAG9	INTEGER*2	0	*	
ICOL	INTEGER*2	16	*	
IROW	INTEGER*2	12	*	
NUM8	INTEGER*2	24	*	
NXTDCT	INTEGER*2	20	*	
WINDOW	INTEGER*2	0		/IMDATA/

417
 418 subroutine getim(toprow)
 419 integer*2 window(64,512),aa,bb,toprow,r,ibuff(512)
 420 common /imdata/window
 421
 422 r=0
 423 do 500 aa=toprow,toprow+63
 1 424 r=r+1
 1 425 call isgetp(0,aa,0,512,ibuff)
 1 426 do 505 bb=1,512
 2 427 window(r,bb)=ibuff(bb)
 2 428 505 continue
 1 429 500 continue
 430 return
 431 end

Name	Type	Offset	P	Class
AA	INTEGER*2	64		
BB	INTEGER*2	68		
IBUFF	INTEGER*2	0		LARGE
R	INTEGER*2	62		
TOPROW	INTEGER*2	0	*	
WINDOW	INTEGER*2	0		/IMDATA/

432
 433 subroutine savpts(bordr,bordc,numedge)
 434 integer*2 bordr(2000),bordc(2000),numedge

```

D Line# 1      7
435          integer*2 aa
436
437 c ..... Write border points to disk.
438
439          open(6,file=' ',status='new')
440          write(6,'(i4/)') numedge
441          do 425 aa=1,numedge
1 442          write(6,426) bordr(aa),bordc(aa)
1 443          format(1x,i4,i4/)
1 444          continue
445          close(6)
446          write(*,'(1x,a/)') 'Border pts. written to disk.'
447          return
448          end

```

Name	Type	Offset	P	Class
AA	INTEGER*2	70		
BORDC	INTEGER*2	4	*	LARGE
BORDR	INTEGER*2	0	*	LARGE
NUMEDG	INTEGER*2	8	*	

Name	Type	Size	Class
DCT8			SUBROUTINE
DCTI			SUBROUTINE
DCTII			SUBROUTINE
DCTIII			SUBROUTINE
DCTIV			SUBROUTINE
DCTV			SUBROUTINE
DCTVI			SUBROUTINE
DCTVII			SUBROUTINE
FRSTPT			SUBROUTINE
GETIM			SUBROUTINE
IMDATA		0	COMMON
ISEND			SUBROUTINE
ISFCLR			SUBROUTINE
ISFREZ			SUBROUTINE
ISGETP			SUBROUTINE
ISINFR			SUBROUTINE
ISINIT			SUBROUTINE
ISINTS			SUBROUTINE
ISOTFR			SUBROUTINE
ISOUTS			SUBROUTINE
ISPASS			SUBROUTINE
ISSYNC			SUBROUTINE
MAIN			PROGRAM
NXTDIR			SUBROUTINE
SAVPTS			SUBROUTINE
UPWIND			SUBROUTINE

Pass One No Errors Detected
 448 Source Lines

APPENDIX C

TSTIMAGE LISTING

Program: test_image

Page 1

01-23-89

08:56:50

Microsoft FORTRAN77 V3.30 March 1985

```
D Line# 1      7
1 $title:'Program: test_image'
2 $storage:2
3   program testimage
4     real*4    dstnc(2000),dstmin,dstmax,xtranspt(25),ytranspt(25)
5     real*4    arx(100),ary(100),cornr(10),cornc(10),dist(4),diffgrad
6     real*4    minpos,maxside,minside,sidetol,sidedif,cz,diagdif
7     integer*2  bordr(2000),bordc(2000),trans(2000)
8     integer*2  trend(100),ttranspt(25),i,flag2,obj,stlen,trd
9     integer*2  numedge,numave,ftrd,isoside,bj,bk,cntzero
10    integer*2  tranpts,endtrd,begtrd,fff,nn
11    real*8     xg,yg,raditol,circl
12    character  fname*20
13
14    common /edgedata/bordr,bordc,numedge
15    common /radidata/dstnc
16    common /deltadat/trans
17    common /trenddat/arx,ary,trend
18    common /trddata/stlen,begtrd,endtrd,fff,trd
19    common /sdata/dist,minpos,sidedif,maxside,minside,obj,isoside,
20    &diagdif
21    common /maxima/cornr,cornc
22
23
24 c .....   Enter all parameter data .....
25
26 c .....   NUMAVE = # pts. to be averaged along edge to find ave pt.
27 c           to calculate radial distance to. (Always be an odd integer)
28 c           RADITOL = the %age 'tolerance' allowed for two radial
29 c           distances to be equal. (radial distance from geo. center to ave
30 c           edge pt.)
31 c           (within raditol% the trans() = 0, > raditol% the trans() = 1,
32 c           <raditol% the trans() = -1.)
33 c           CIRCL = if the %age of trans() entries equal to 0 is above circ
34 c           the program is routed to the circle test section.
35 c           STLEN = # of adjacent points that must have the same trans valu
36 c           (1 or -1 only) to define a trend.
37 c           SIDETOL = the tolerance allowed for two lengths (sides or
38 c           diagonals) to be equal (expressed as a percentage).
39 c           The values assigned to these parameters below are the optimum
40 c           values found in the research if this thesis and are
41 c           dependent on the shapes used.
42
43           numave=7
44           raditol=0.2
45           circl=65.0
46           stlen=8
47           sidetol=6.5
48
49 c ..... Next 11 lines can be used to input the parameters
50 c           write(*,333) 'Enter # points to be averaged ( NUMAVE ):'
51 c           read(*,'(bn,i3)') numave
52 c           write(*,333) 'Enter %age for TRANS=0 ( RADITOL ):'
53 c           read(*,'(bn,f12.8)') raditol
54 c           write(*,333) 'Enter %age border=0=circle ( CIRCL ):'
55 c           read(*,'(bn,f12.8)') circl
56 c           write(*,333) 'Enter string length to denote a trend (STLEN):'
```

```

D Line# 1      7      Microsoft FORTRAN77 V3.30 March 1985
57 c      read(*,'(bn,i3)') stlen
58 c      write(*,333) 'Enter % age tolerance for equal sides (SIDETOL):'
59 c      read(*,'(bn,f6.3)') sidetol
60 c 333 format(1x,a\
61
62
63 c ..... Read edge points from disk. These were extracted by program
64 c ..... TSTLOAD. Parameter numedge = # edge points extracted. The
65 c ..... row & column coordinates of edge points are stored in arrays
66 c ..... bordr & bordc respectively.
67
68      write(*,'(1x,a\')') 'Enter name of data file:'
69      read(*,'(a\')') fname
70      open(1,file=fname,status='old',access='sequential')
71      read(1,'(i4/)') numedge
72      do 435 i=1,numedge
1 73          read(1,'(1x,i4,i4/)') bordr(i),bordc(i)
1 74      435 continue
75          close(1,status='keep')
76
77 c ..... Array initialization .....
78
79      do 43 i=1,2000
1 80          dstnc(i)=0.0
1 81          trans(i)=0
1 82      43 continue
83
84 c ..... Calculate the area of the object and therefore the
85 c ..... geometric center (xg,yg).
86
87      call centroid(xg,yg)
88
89 c ..... Calculate the radius from (xg,yg) to each point on the
90 c ..... edge. The points on the edge are averaged to provide an
91 c ..... ave. row & col (xave,yave). This eliminates variances due
92 c ..... to digitization. The # points to be averaged for (xave,yave)
93 c ..... is = parameter NUMAVE. NUMAVE is always odd to associate
94 c ..... it with an actual edge point without bias. The radial
95 c ..... distances from (xg,yg) to (xave,yave) is stored in dstnc().
96
97      call radius(numave,dstmin,dstmax,xg,yg)
98
99 c ..... Calculate the trans array. The distances (dstnc) of two
100 c ..... adjacent edge points, n & n-1 are compared.
101 c ..... If their %age difference is > raditol%, trans(n) = 1;
102 c ..... If their %age difference is < raditol%, trans(n) = -1;
103 c ..... If their %age difference is within raditol%, trans = 0.
104 c ..... parameter cntzero = # points with trans = 0.
105
106      call gradient(numedge,cntzero,raditol)
107
108      obj=0
109      cz=cntzero
110      write(*,'(1x,a,i5/)') 'cntzero=',cntzero
111
112 c ..... If circl% of the points have trans()=0, the object is

```

```

D Line# 1      7      Microsoft FORTRAN77 V3.30 March 1985
113 c      circular: goto circle test @ 700
114
115      if((cz/numedge).ge.(circl/100))then
116          obj=4
117          goto 700
118      endif
119
120 c ..... If not circular, replace all 0's imbedded between
121 c          identical non zero trans nos. with the non-zero value.
122 c          (i.e. -1,0,0,0,-1 or 1,0,0,0,1 will have all the zeros
123 c          changed to -1 or 1 resp.; 1,0,0,0,-1 or -1,0,0,0,1 will
124 c          not change any.)
125
126      call repzero(numedge)
127
128 c ..... Replace all single imbedded 1's or -1's, as they are due to
129 c          digitization. (i.e. [1,-1,1] & [-1,1,-1] will have their
130 c          -1 & 1 replaced with 1 & -1 resp.)
131
132      call repones(numedge)
133
134 c ..... Have all raw data; now extract FEATURES .....
135
136 c ..... Locate trends (defined as stlen # of consecutive
137 c          points with same trans value [1 or -1] ).
138
139      call firsttd(numedge)
140
141 c ..... If a trend was not found, the data is incorrect, therefore
142 c          end program.
143
144      flag2=0
145      if(trd.eq.0)then
146          flag2=1
147          goto 600
148      endif
149      ftrd=trd
150
151 c ..... Using first trend (ftrd), find all trends. ....
152 c          Each change in trend is defined by 4 points.
153 c          The 1st 2 points define trend 1 which is the last set of
154 c          STLEN pts with same trans value (1 or -1) BEFORE the
155 c          transition pt. and whose coordinates are denoted by x1,y1 &
156 c          x2,y2. The 2nd 2 points define trend 2 which is the first set
157 c          of STLEN points with the same trans value (-1 or 1) AFTER the
158 c          transition pt. and whose coordinates are denoted by x3,y3 &
159 c          x4,y4. The trans values for trend 1 & 2 will be opposite,
160 c          (i.e. 1 & -1, or -1 & 1).
161 c          The transition point is either a maxima (corner) or minima (si
162 c          midpoint).
163 c          The coordinates of the 4 points for every transition area
164 c          (tranpts) are stored in arx(),ary() .
165 c          Their transition value is stored in trend() .
166
167      call alltrds(tranpts)
168

```

Program: test_image

Page 4

01-23-89

08:56:50

```
D Line# 1      7      Microsoft FORTRAN77 V3.30 March 1985
169 c ..... Save feature data to disk .....
170
171 c      call savedata(tranpts)
172
173 c ..... Calculate the x-y coordinate (avex,avey) of each transition
174 c      point. These are either a maxima or minima. This is
175 c      accomplished by a linear interpolation between
176 c      line a [defined by (arx(n),ary(n)) and (arx(n+1),ary(n+1))] an
177 c      line b [defined by (arx(n+2),ary(n+2)) and (arx(n+3),ary(n+3))]
178
179      call corners(tranpts,nn)
180
181 c ..... If nn (total # of maxima and minima) is not equal to 6 or 8
182 c      the object is neither triangle nor quadrilateral.
183
184      if((nn.ne.8).and.(nn.ne.6))then
185          write(*,'(lx,a,i4,a/)') 'nn=',nn,' UNKNOWN OBJECT.'
186          write(*,'(lx,a,f12.7,a,lx,f12.7,a/)') 'Geometric center (row,col)
187          & :(' ,xg,',',yg,')'
188          goto 800
189      endif
190
191 c ..... If nn=8 then object is a quadrilateral.
192 c      Determine type of quadrilateral; only ones interested in
193 c      this thesis are squares or rectangles. All others are
194 c      denoted as unknown quadrilaterals.
195
196 c ..... dist(1) - dist(4)=lengths of 4 sides;angl1 to angl4 = angles
197 c      formed between each side and the top of the screen.
198 c      sidedif = difference between the longest & shortest sides.
199
200      if(nn.eq.8)then
201          obj=1
202          call quad(sidetol,xg,yg)
203          write(*,'(lx,a,i4/a/)') 'obj=',obj
204          if(obj.eq.1)then
205              write(*,'(lx,a,f12.7,lx,a/)') 'Object= UNKNOWN QUADRILATERAL
206              &, angle=',minpos,' degrees.'
207          else
208              if(obj.eq.10)then
209                  write(*,'(lx,a,f12.7,lx,a/)') 'Object= SQUARE , angle= ',
210                  & minpos,' degrees.'
211              else
212                  if(obj.eq.12)then
213                      write(*,'(lx,a,f12.7,lx,a/)') 'Object= RECTANGLE, angle=',
214                      & minpos,' degrees.'
215                  endif
216              endif
217          endif
218          sidedif=sidedif*100
219          diagdif=diagdif*100
220          write(*,'(lx,a,f12.7,a,lx,f12.7,a/)') 'Geometric center (row,col)
221          & :(' ,xg,',',yg,')'
222          write(*,'(lx,a,f12.7,lx,a,f12.7,lx,a,f6.3,a/)') 'maxside=',
223          & maxside,' minside=',minside,' sidetol=',sidetol,'%'
224          write(*,'(lx,a,f12.7,a,lx,a,f12.7,a/)') 'sidedif=',sidedif,'%'
```

```

D Line# 1      7      Microsoft FORTRAN77 V3.30 March 1985
225      & 'diagdif =',diagdif,'% '
226      write(*,'(lx,a,f12.7,1x,a,f12.7/)') 'dist1=',dist(1),' dist2=',
227      &dist(2)
228      write(*,'(lx,a,f12.7,1x,a,f12.7/)') 'dist3=',dist(3),' dist4=',
229      &dist(4)
230      write(*,'(lx,a,f12.7,2x,f12.7/)') 'c1=',cornr(1),cornc(1)
231      write(*,'(5x,a,f12.7,2x,f12.7/)') 'c2=',cornr(2),cornc(2)
232      write(*,'(lx,a,f12.7,2x,f12.7/)') 'c3=',cornr(3),cornc(3)
233      write(*,'(5x,a,f12.7,2x,f12.7/)') 'c4=',cornr(4),cornc(4)
234      goto 800
235      endif
236
237 c ..... If nn (total # maxima & minima) = 6 then object is triangular
238 c Determine type of triangle equilateral or isosceles. All
239 c others are unknown triangles.
240
241 c ..... dist1 to dist3 are lengths of sides.
242 c ..... angl1 to angl3 are angles of sides.
243
244      if(nn.eq.6)then
245          call triangle(sidetol,xg,yg)
246          if(obj.eq.20)then
247              write(*,'(lx,a,f12.7,1x,a/)') 'Object= EQUILATERAL TRIANGLE,
248      & minpos=',minpos,' degrees.'
249          else
250              if(obj.eq.21)then
251                  write(*,'(lx,a,f12.7,1x,a/)') 'Object= ISOSCELES TRIANGLE,
252      & minpos=',minpos,' degrees.'
253                  bj=isoside
254                  bk=bj+1
255                  if(bk.eq.4)then
256                      bk=1
257                  endif
258                  write(*,'(lx,a,i3,a,i3,a/)') 'Sides',bj,' and',bk,' are equal.'
259              else
260                  if(obj.eq.2)then
261                      write(*,'(lx,a,f12.7,1x,a/)') 'Object=UNKNOWN TRIANGLE,
262      &angle=',minpos,' degrees.'
263                  endif
264              endif
265          endif
266          sidedif=sidedif*100
267          write(*,'(lx,a,f12.7,a,1x,f12.7,a/)') 'Geometric center (row,col)
268      & :(',xg,',',yg,')'
269          write(*,'(lx,a,f12.7,1x,a,f12.7,1x,a,f12.7/)') 'dist1=',
270      & dist(1),'dist2=',dist(2),'dist3=',dist(3)
271          write(*,'(lx,a,f12.7,2x,f12.7/)') 'c1=',cornr(1),cornc(1)
272          write(*,'(lx,a,f12.7,2x,f12.7/)') 'c2=',cornr(2),cornc(2)
273          write(*,'(lx,a,f12.7,2x,f12.7/)') 'c3=',cornr(3),cornc(3)
274          write(*,'(lx,a,f12.7,1x,a,f12.7,1x,a,f6.3,1x,a,f12.7,a/)')
275      &'maxside=',maxside,' minside=',minside,' sidetol=',sidetol,
276      &'% sidedif=',sidedif,'% '
277          goto 800
278      endif
279
280 c .... No trend is found, therefore no maxima/minima found.

```

D Line# 1 7

Microsoft FORTRAN77 V3.30 March 1985

```

281
282 600 if(flag2.eq.1)then
283     write(*,'(lx,a/)' ) 'No Trend Found'
284     write(*,'(lx,a,f12.7,a,lx,f12.7,a/)' ) 'Geometric center (row,col)
285     & :(' ,xg',' ,yg,')'
286     endif
287     goto 800
288
289 c ..... Circle test. If the difference between the longest and shortes
290 c     radii is within the tolerance sidetol, the object is a circle.
291 c     Otherwise the object is elliptical (unknown object).
292
293 700 if(obj.eq.4)then
294     diffmad=(dstmax-dstmin)/dstmin
295     if(diffmad.le.(sidetol/100))then
296         write(*,'(lx,a/)' ) 'Object is a CIRCLE'
297     else
298         write(*,'(lx,a/)' ) 'Object is ELLIPTICAL'
299     endif
300
301 c ..... To save circle data:
302 c     call savcir()
303
304     write(*,'(lx,a,f12.7,a,lx,f12.7,a/)' ) 'Geometric center (row,col)
305     & :(' ,xg',' ,yg,')'
306     write(*,'(lx,a,f12.7,a,f12.7,3x,a,f12.7,a/)' ) 'Radii min,max:',
307     & dstmin,' ,dstmax,':difference=',diffmad,'.'
308     endif
309
310 800 pause
311     stop
312     end

```

Name	Type	Offset	P	Class
ARX	REAL	0		/TRENDD/
ARY	REAL	400		/TRENDD/
BEGTRD	INTEGER*2	2		/TRDDAT/
BJ	INTEGER*2	354		
BK	INTEGER*2	356		
BORDC	INTEGER*2	4000		/EDGEDA/
BORDR	INTEGER*2	0		/EDGEDA/
CIRCL	REAL*8	280		
CNTZER	INTEGER*2	340		
CORNC	REAL	40		/MAXIMA/
CORNR	REAL	0		/MAXIMA/
CZ	REAL	342		
DIAGDI	REAL	36		/SDATA /
DIFFRA	REAL	358		
DIST	REAL	0		/SDATA /
DSTMAX	REAL	336		
DSTMIN	REAL	332		
DSTNC	REAL	0		/RADIDA/
ENDTRD	INTEGER*2	4		/TRDDAT/
FFF	INTEGER*2	6		/TRDDAT/
FLAG2	INTEGER*2	346		

Program: test_image

Page 7
01-23-89
08:56:50

Microsoft FORTRAN77 V3.30 March 1985

```
D Line# 1.      7
FNAME  CHAR*20      292
FTRD   INTEGER*2    348
I       INTEGER*2    312
ISOSID INTEGER*2      34 /SDATA /
MAXSID REAL          24 /SDATA /
MINPOS REAL          16 /SDATA /
MINSID REAL          28 /SDATA /
NN      INTEGER*2    352
NUMAVE  INTEGER*2    270
NUMEDG  INTEGER*2    8000 /EDGEDA/
OBJ      INTEGER*2     32 /SDATA /
RADITO  REAL*8       272
SIDEID  REAL         20 /SDATA /
SIDEIO  REAL        288
STLEN   INTEGER*2     0 /TRDDAT/
TRANPT  INTEGER*2    350
TRANS   INTEGER*2     0 /DELTAD/
TRD      INTEGER*2     8 /TRDDAT/
TREND   INTEGER*2    800 /TREND/
TTRANS  INTEGER*2    220
XG      REAL*8       316
XTRANS  REAL         20
YG      REAL*8       324
YTRANS  REAL        120

313
314
315      subroutine centroid(xg,yg)
316          integer*2  bordr(2000),bordc(2000),numedge,ff
317          real*8      xg,yg,area,sumarea
318          real*8      p1,p2,p3,p4,p5,p6,p7,p8
319          common /edgedata/bordr,bordc,numedge
320
321 c ..... Routine uses GREEN'S theoremin a plane to calculate the area
322 c          of the object (AREA), and the geometric center of the object
323 c          (XG,YG).
324 c          P1-P8 are used for number manipulation.
325
326          sumarea=0.0
327          xg=0.0
328          yg=0.0
329          area=0.0
330          do 50 ff=1,numedge-1
1 331              p1=bordr(ff)*bordc(ff+1)
1 332              p2=bordc(ff)*bordr(ff+1)
1 333              p3=bordr(ff)*bordr(ff)
1 334              p4=bordc(ff)*bordc(ff)
1 335              sumarea=sumarea+(p1-p2)
1 336              xg=xg+(p3*(bordc(ff+1)-bordc(ff)))
1 337              yg=yg+(p4*(bordr(ff+1)-bordr(ff)))
1 338          50 continue
339          p5=bordr(numedge)*bordc(1)
340          p6=bordc(numedge)*bordr(1)
341          p7=bordr(numedge)*bordr(numedge)
342          p8=bordc(numedge)*bordc(numedge)
```

```

343      sumarea=sumarea+(p5-p6)
344      xg=xg+(p7*(bordc(1)-bordc(numedge)))
345      yg=yg+(p8*(bordr(1)-bordr(numedge)))
346      area=0.5*sumarea
347      xg=0.5*xg/area
348      yg=-0.5*yg/area
349      return
350      end

```

Name	Type	Offset	P	Class
AREA	REAL*8	370		
BORDC	INTEGER*2	4000		/EDGEDA/
BORDR	INTEGER*2	0		/EDGEDA/
FF	INTEGER*2	378		
NUMEDG	INTEGER*2	8000		/EDGEDA/
P1	REAL*8	382		
P2	REAL*8	390		
P3	REAL*8	398		
P4	REAL*8	406		
P5	REAL*8	414		
P6	REAL*8	422		
P7	REAL*8	430		
P8	REAL*8	438		
SUMARE	REAL*8	362		
XG	REAL*8	0	*	
YG	REAL*8	4	*	

```

351
352      subroutine radius(numave,dstmin,dstmax,xg,yg)
353      integer*2 bordr(2000),bordc(2000),numedge,ff,dd,numave,mm,n
354      real*8      xg,yg
355      real*4      dstnc(2000),xaves,yaves,dstmin,dstmax
356      common /edgedata/bordr,bordc,numedge
357      common /radidata/dstnc
358
359 c ..... Routine calculates the DSTNC array.
360 c      XG,YG = geometric center of object.
361 c      NUMAVE = # pts. along edge to be averaged (always odd).
362 c      XAVES,YAVES = coordinates of the average of NUMAVE edge pts.
363 c      MM = actual pt. the DSTNC is to be associated with.
364 c      i.e.: average 5 pts. (numave=5) to obtain the average
365 c      coordinates xaves,yaves; take the distance from xg,yg to
366 c      xaves,yaves, storing the distance in DSTNC(3) (MM=3).
367 c      DSTMIN,DSTMAX = shortest and longest distances from xg,yg.
368
369      mm=0
370      do 60 ff=1,numedge
1 371      dd=ff
1 372      xaves=0.0
1 373      yaves=0.0
1 374      do 61 n=1,numave
2 375      xaves=xaves+bordr(dd)
2 376      yaves=yaves+bordc(dd)
2 377      dd=dd+1

```

```

D Line# 1      7      Microsoft FORTRAN77 V3.30 March 1985
2  378      if(dd.eq.(numedge+1)) then
2  379          dd=1
2  380      endif
2  381      61      continue
1  382          mm=ff+((numave-1)/2)
1  383          if(mm.gt.numedge) then
1  384              mm=mm-numedge
1  385          endif
1  386          xaves=xaves/numave
1  387          yaves=yaves/numave
1  388          dstnc(mm)=sqrt((xg-xaves)**2+(yg-yaves)**2)
1  389          if(ff.eq.1) then
1  390              dstmin=dstnc(mm)
1  391              dstmax=dstmin
1  392          endif
1  393          if(dstnc(mm).lt.dstmin) then
1  394              dstmin=dstnc(mm)
1  395          else
1  396              if(dstnc(mm).gt.dstmax) then
1  397                  dstmax=dstnc(mm)
1  398              endif
1  399          endif
1  400      60      continue
401          return
402      end

```

Name	Type	Offset	P	Class
BORDC	INTEGER*2	4000		/EDGEDA/
BORDR	INTEGER*2	0		/EDGEDA/
DD	INTEGER*2	452		
DSTMAX	REAL	8	*	
DSTMIN	REAL	4	*	
DSTNC	REAL	0		/RADIDA/
FF	INTEGER*2	448		
MM	INTEGER*2	446		
N	INTEGER*2	462		
NUMAVE	INTEGER*2	0	*	
NUMEDG	INTEGER*2	8000		/EDGEDA/
SQRT				INTRINSIC
XAVES	REAL	454		
XG	REAL*8	12	*	
YAVES	REAL	458		
YG	REAL*8	16	*	

```

403
404      subroutine gradient(numedge,cntzero,raditol)
405          integer*2  trans(2000),cntzero,numedge,ff
406          real*4      dstnc(2000)
407          real*8      raditol,delta
408          common /radidata/dstnc
409          common /deltadat/trans
410
411 c ..... DELTA = %difference between pt. FF and pt. FF-1, as a
412 c                percentage of FF.

```

```

D Line# 1      7      Microsoft FORTRAN77 V3.30 March 1985
413 c      RADITOL = % tolerance used to compare distances to FF &
414 c      FF-1. Within + or - RADITOL, TRANS()=0.
415 c      Above +RADITOL, TRANS()=+1.
416 c      Below -RADITOL, TRANS()=-1.
417 c      CNTZERO = # pts whose TRANS()=0.
418
419      cntzero=0
420      do 70 ff=1,numedge
1 421          if(ff.gt.1)then
1 422              delta=dstnc(ff)-dstnc(ff-1)
1 423          else
1 424              delta=dstnc(1)-dstnc(numedge)
1 425          endif
1 426          delta=delta/dstnc(ff)
1 427          if(delta.gt.(raditol/100))then
1 428              trans(ff)=1
1 429          else
1 430              if(delta.lt.(-1*raditol/100))then
1 431                  trans(ff)=-1
1 432              else
1 433                  trans(ff)=0
1 434                  cntzero=cntzero+1
1 435              endif
1 436          endif
1 437      70 continue
438      return
439      end

```

Name	Type	Offset	P	Class
CNTZER	INTEGER*2	4	*	
DELTA	REAL*8	470		
DSTNC	REAL	0		/RADIDA/
FF	INTEGER*2	466		
NUMEDG	INTEGER*2	0	*	
RADITO	REAL*8	8	*	
TRANS	INTEGER*2	0		/DELTAD/

```

440
441      subroutine repzero(numedge)
442          integer*2 trans(2000),numedge,last0,first0,flag3,ff,ddd,dd
443          integer*2 ag,fff,fz
444          common /deltadat/trans
445
446 c ..... find LAST non-zero trans value point (last0).
447
448      last0=0
449      flag3=0
450      do 80 ff=numedge,1,-1
1 451          if(flag3.ne.1)then
1 452              if(trans(ff).ne.0)then
1 453                  last0=ff
1 454                  flag3=1
1 455              endif
1 456          endif

```

```

D Line# 1      7
1 457      80  continue
458
459 c ..... find FIRST non-zero trans value point (first0).
460
461         first0=0
462         flag3=0
463         do 81 ff=1,numedge
1 464             if(flag3.ne.1)then
1 465                 if(trans(ff).ne.0)then
1 466                     first0=ff
1 467                     flag3=1
1 468                 endif
1 469             endif
1 470      81  continue
471
472 c ..... Compare trans(last0) & trans(first0); if same, convert all
473 c ..... points between to trans(their value) [1 or -1].
474
475         ddd=numedge-last0+first0
476         if(trans(last0).eq.trans(first0))then
477             do 82 ff=last0+1,last0+ddd
1 478                 dd=ff
1 479                 if(ff.ge.numedge+1)then
1 480                     dd=ff-numedge
1 481                 endif
1 482                 trans(dd)=trans(last0)
1 483      82  continue
484             endif
485
486 c ..... Continue until all points have been examined.
487
488         ag=first0+1
489         do 83 ff=ag,last0
1 490             if(trans(ff).eq.0)then
1 491                 flag3=0
1 492                 do 84 fff=ff+1,last0
2 493                     if(flag3.eq.0)then
2 494                         if(trans(fff).ne.0)then
2 495                             flag3=1
2 496                             fz=fff
2 497                         endif
2 498                     endif
2 499      84  continue
1 500                 if(trans(first0).eq.trans(fz))then
1 501                     do 85 fff=first0+1,fz-1
2 502                         trans(fff)=trans(first0)
2 503                     continue
1 504                 endif
1 505                 first0=fz
1 506             else
1 507                 first0=ff
1 508             endif
1 509      83  continue
510         return
511         end

```

Program: test_image

Page 12

01-23-89

08:56:50

D Line# 1 7

Microsoft FORTRAN77 V3.30 March 1985

Name	Type	Offset	P	Class
AG	INTEGER*2	494		
DD	INTEGER*2	492		
DDD	INTEGER*2	488		
FF	INTEGER*2	482		
FFF	INTEGER*2	498		
FIRST0	INTEGER*2	484		
FLAG3	INTEGER*2	480		
FZ	INTEGER*2	502		
LAST0	INTEGER*2	478		
NUMEDG	INTEGER*2	0	*	
TRANS	INTEGER*2	0		/DELTAD/

```
512
513      subroutine repones(numedge)
514          integer*2 trans(2000),numedge,ff
515          common /deltadat/trans
516
517          if((trans(1).ne.trans(numedge)).and.(trans(numedge).eq.
518      &      trans(2)))then
519              trans(1) = trans(numedge)
520          endif
521          do 87 ff=2,numedge-1
1 522              if((trans(ff).ne.trans(ff-1)).and.
1 523      &      (trans(ff-1).eq.trans(ff+1)))then
1 524                  trans(ff)=trans(ff-1)
1 525              endif
1 526      87 continue
527              if((trans(numedge).ne.trans(numedge-1)).and.
528      &      (trans(numedge-1).eq.trans(1)))then
529                  trans(numedge)=trans(numedge-1)
530              endif
531          return
532      end
```

Name	Type	Offset	P	Class
FF	INTEGER*2	506		
NUMEDG	INTEGER*2	0	*	
TRANS	INTEGER*2	0		/DELTAD/

```
533
534      subroutine firsttd(numedge)
535          integer*2 trans(2000),numedge,trd,fff,begtrd,endtrd,trdsum
536          integer*2 flag6,dd,ff,stlen,n
537          common /deltadat/trans
538          common /trddata/stlen,begtrd,endtrd,fff,trd
539
540 c ..... Routine : take STLEN adjacent pts; add their TRANS() values;
541 c If all are +1's, their sum (TRDSUM) is equal to +STLEN.
542 c Thus, the beginning of an increasing trend has been found.
543 c If all are -1's, their sum (TRDSUM) is equal to -STLEN.
```

Program: test_image

Page 13

01-23-89

08:56:50

D Line# 1 7 Microsoft FORTRAN77 V3.30 March 1985

```

544 c      Thus, the beginning of a decreasing trend has been found.
545 c      BEGTRD = the first point of STLEN pts. of the trend found.
546 c      ENDTRD = the last point of STLEN pts. of the trend found.
547 c      TRD = the type of trend found (1 or -1).
548 c      Once the first trend has been located, the routine ends.
549
550      flag6=0
551      do 90 ff=1,(numedge+30)
1 552          if(flag6.eq.0)then
1 553              dd=ff
1 554              if(dd.gt.numedge)then
1 555                  dd=ff-numedge
1 556              endif
1 557              begtrd=0
1 558              endtrd=0
1 559              trdsum=0
1 560              trd=0
1 561              do 91 n=1,stlen
2 562                  if(dd.eq.(numedge+1))then
2 563                      dd=1
2 564                  endif
2 565                  if(n.eq.1)then
2 566                      begtrd=dd
2 567                  else
2 568                      if(n.eq.stlen)then
2 569                          endtrd=dd
2 570                      endif
2 571                  endif
2 572                  trdsum=trdsum+trans(dd)
2 573                  dd=dd+1
2 574          91      continue
1 575          if(trdsum.eq.stlen)then
1 576              flag6=1
1 577              fff=ff
1 578              trd=1
1 579          else
1 580              if(trdsum.eq.((-1)*stlen))then
1 581                  flag6=1
1 582                  fff=ff
1 583                  trd=-1
1 584              else
1 585                  trd=0
1 586                  flag6=0
1 587              endif
1 588          endif
1 589      endif
1 590      90      continue
1 591      return
592      end

```

Name	Type	Offset	P	Class
BEGTRD	INTEGER*2	2		/TRDDAT/
DD	INTEGER*2	516		
ENDTRD	INTEGER*2	4		/TRDDAT/
FF	INTEGER*2	512		

Program: test_image

Page 14

01-23-89

08:56:50

Microsoft FORTRAN77 V3.30 March 1985

```
D Line# 1      7
FFF  INTEGER*2      6  /TRDDAT/
FLAG6 INTEGER*2    510
N    INTEGER*2    520
NUMEDG INTEGER*2    0 *
STLEN INTEGER*2    0  /TRDDAT/
TRANS INTEGER*2    0  /DELTAD/
TRD  INTEGER*2    8  /TRDDAT/
TRDSUM INTEGER*2   518
```

```
593
594      subroutine alltrds(tranpts)
595
596          integer*2  bordr(2000),bordc(2000),trans(2000),trend(100)
597          integer*2  numedge,tranpts,begtrd,endtrd,stlen,x1,x2,x3,x4
598          integer*2  y1,y2,y3,y4,t1,t2,t3,t4,ff,fff,dd,trdsum,n
599          integer*2  nexttrd,flag4,trd,i,flag5
600          real*4     arx(100),ary(100)
601          common /edgedata/bordr,bordc,numedge
602          common /deltadat/trans
603          common /trenddat/arx,ary,trend
604          common /trddata/stlen,begtrd,endtrd,fff,trd
605
606 c ..... Routine: looks for the last of the first trend and the start
607 c          of the next (opposite) trend. The last of the first trend
608 c          consists of STLEN pts., the first point = BEGTRD, the STLEnth
609 c          point = ENDTRD. The coordinates of BEGTRD are stored in X1,Y
610 c          The coordinates of ENDTRD are stored in X2,Y2. The start of
611 c          the next trend consists of STLEN pts, the first and STLEnth p
612 c          are BEGTRD and ENDTRD again. Their coordinates are stored in
613 c          X3,Y3 & X4,Y4, respectively. When all four points are locate
614 c          they are stored in arrays ARX(),ARY(), and their trend types
615 c          (T1 to T4) are stored in TREND(). The routine continues until
616 c          all transition points are located.
617
618          do 88 i=1,100
1 619              arx(i)=0.0
1 620              ary(i)=0.0
1 621              trend(i)=0
1 622      88      continue
623
624 c ..... determine end of initial trend (ftrd) .....
625          t1=0
626          t2=0
627          t3=0
628          t4=0
629          x1=0
630          x2=0
631          x3=0
632          x4=0
633          y1=0
634          y2=0
635          y3=0
636          y4=0
637          tranpts=1
638          x1=bordr(begtrd)
```

```

D Line# 1      7      Microsoft FORTRAN77 V3.30 March 1985
639      x2=bordr(endtrd)
640      y1=bordc(begtrd)
641      y2=bordc(endtrd)
642      t1=trans(begtrd)
643      t2=trans(endtrd)
644      flag5=0
645      do 100 ff=fff,(numedge+fff-1+30)
1 646          dd=ff
1 647          if(dd.gt.numedge)then
1 648              dd=ff-numedge
1 649          endif
1 650          trdsum=0
1 651          do 102 n=1,stlen
2 652              if(dd.eq.(numedge+1))then
2 653                  dd=1
2 654              endif
2 655              if(n.eq.1)then
2 656                  begtrd=dd
2 657              else
2 658                  if(n.eq.stlen)then
2 659                      endtrd=dd
2 660                  endif
2 661              endif
2 662              trdsum=trdsum+trans(dd)
2 663              dd=dd+1
2 664      102      continue
1 665          if(trdsum.eq.(trd*stlen))then
1 666
1 667 c ..... end of initial trend found .....
1 668
1 669          x1=bordr(begtrd)
1 670          y1=bordc(begtrd)
1 671          x2=bordr(endtrd)
1 672          y2=bordc(endtrd)
1 673          t1=trans(begtrd)
1 674          t2=trans(endtrd)
1 675          flag5=1
1 676      else
1 677          nexttrd=(-1)*stlen)*trd
1 678
1 679 c ..... beginning of opposite trend found .....
1 680
1 681          if(trdsum.eq.nexttrd)then
1 682              x3=bordr(begtrd)
1 683              y3=bordc(begtrd)
1 684              x4=bordr(endtrd)
1 685              y4=bordc(endtrd)
1 686              t3=trans(begtrd)
1 687              t4=trans(endtrd)
1 688              flag5=2
1 689          endif
1 690      endif
1 691      flag4=0
1 692      do 93 i=1,(tranpts-2)
2 693
2 694 c ..... If first points are relocated, end loop .....

```

Program: test_image

Page 16

01-23-89

08:56:50

Microsoft FORTRAN77 V3.30 March 1985

```

D Line# 1      7
2 695
2 696          if((arx(i).eq.x1).and.(ary(i).eq.y1))then
2 697              if((arx(i+1).eq.x2).and.(ary(i+1).eq.y2))then
2 698                  flag4=1
2 699              endif
2 700          endif
2 701      93      continue
1 702          if(flag4.eq.0)then
1 703              if(flag5.eq.2)then
1 704                  arx(tranpts)=x1
1 705                  ary(tranpts)=y1
1 706                  arx(tranpts+1)=x2
1 707                  ary(tranpts+1)=y2
1 708                  arx(tranpts+2)=x3
1 709                  ary(tranpts+2)=y3
1 710                  arx(tranpts+3)=x4
1 711                  ary(tranpts+3)=y4
1 712                  trend(tranpts)=t1
1 713                  trend(tranpts+1)=t2
1 714                  trend(tranpts+2)=t3
1 715                  trend(tranpts+3)=t4
1 716                  x1=x3
1 717                  x2=x4
1 718                  y1=y3
1 719                  y2=y4
1 720                  tranpts=tranpts+4
1 721                  trd=(-1)*trd
1 722                  flag5=0
1 723                  t1=0
1 724                  t2=0
1 725                  t3=0
1 726                  t4=0
1 727              endif
1 728          endif
1 729      100      continue
730          return
731      end

```

Name	Type	Offset	P	Class
ARX	REAL	0		/TRENDD/
ARY	REAL	400		/TRENDD/
BEGTRD	INTEGER*2	2		/TRDDAT/
BORDC	INTEGER*2	4000		/EDGEDA/
BORDR	INTEGER*2	0		/EDGEDA/
DD	INTEGER*2	556		
ENDTRD	INTEGER*2	4		/TRDDAT/
FF	INTEGER*2	552		
FFF	INTEGER*2	6		/TRDDAT/
FLAG4	INTEGER*2	566		
FLAG5	INTEGER*2	550		
I	INTEGER*2	524		
N	INTEGER*2	560		
NEXTTR	INTEGER*2	564		
NUMEDG	INTEGER*2	8000		/EDGEDA/
STLEN	INTEGER*2	0		/TRDDAT/

```

D Line# 1      7
T1      INTEGER*2      526
T2      INTEGER*2      528
T3      INTEGER*2      530
T4      INTEGER*2      532
TRANPT  INTEGER*2      0 *
TRANS   INTEGER*2      0 /DELTAD/
TRD     INTEGER*2      8 /TRDDAT/
TRDSUM  INTEGER*2      558
TREND   INTEGER*2      800 /TREND/
X1      INTEGER*2      534
X2      INTEGER*2      536
X3      INTEGER*2      538
X4      INTEGER*2      540
Y1      INTEGER*2      542
Y2      INTEGER*2      544
Y3      INTEGER*2      546
Y4      INTEGER*2      548

```

```

732
733      subroutine savedata(tranpts)
734      integer*2  bordr(2000),bordc(2000),trans(2000),numedge,
735      &          trend(100),tranpts
736      real*4     dstnc(2000),arx(100),ary(100)
737      common /edgedata/bordr,bordc,numedge
738      common /radidata/dstnc
739      common /deltadat/trans
740      common /trenddat/arx,ary,trend
741
742 c ..... Routine saves data on disk.
743
744      open(6,file=' ',status='new')
745      write(*, '(1x,a/)' ) 'Writing data to disk .....'
746      write(*, '(1x,a,t7,a,t14,a,t27,a,t39,a/)' ) 'numedge#','bordr',
747      & 'bordc','dstnc','trans'
748      do 200 n=1,numedge
1 749      write(6, '(i4,1x,i3,1x,i3,1x,f15.11,1x,i3/)' ) n,bordr(n),
1 750      & bordc(n),dstnc(n),trans(n)
1 751      200 continue
752      write(*, '(3x,a,t12,a,t22,a,t30,a/)' ) 'tranpts#','arx','ary',
753      & 'trend'
754      do 205 n=1,tranpts-1
1 755      write(6, '(i4,1x,f7.3,1x,f7.3,1x,i4/)' ) n,arx(n),ary(n),
1 756      & trend(n)
1 757      205 continue
758      close(6)
759      write(*, '(1x,a/)' ) '..... finished writing.'
760
761      return
762      end

```

Name	Type	Offset	P	Class
ARX	REAL	0		/TREND/
ARY	REAL	400		/TREND/
BORDC	INTEGER*2	4000		/EDGEDA/

```

D Line# 1      7
BORDR  INTEGER*2      0  /EDGEDA/
DSTNC  REAL           0  /RADIDA/
N      INTEGER*2     570
NUMEDG INTEGER*2     8000 /EDGEDA/
TRANPT INTEGER*2      0 *
TRANS  INTEGER*2      0  /DELTAD/
TREND  INTEGER*2     800  /TREND/

```

```

763
764      subroutine savcir()
765          integer*2 bordr(2000),bordc(2000),trans(2000),numedge,n
766          real*4   dstnc(2000)
767          common /edgedata/bordr,bordc,numedge
768          common /radidata/dstnc
769          common /deltadat/trans
770
771          open(6,file=' ',status='new')
772          write(*,'(lx,a/)' ) 'Writing data to disk .....'
773          do 217 n=1,numedge
1 774              write(6,'(i4,lx,i3,lx,i3,lx,f15.11,lx,i3/)' ) n,bordr(n),
1 775              & bordc(n),dstnc(n),trans(n)
1 776          217 continue
777          close(6)
778          write(*,'(lx,a/)' ) '..... finished writing.'
779          return
780      end

```

Name	Type	Offset	P	Class
BORDC	INTEGER*2	4000		/EDGEDA/
BORDR	INTEGER*2	0		/EDGEDA/
DSTNC	REAL	0		/RADIDA/
N	INTEGER*2	576		
NUMEDG	INTEGER*2	8000		/EDGEDA/
TRANS	INTEGER*2	0		/DELTAD/

```

781
782      subroutine corners(tranpts,nn)
783          integer*2 trend(100),tranpts,nn,i,ttranspt(25),inittrd,numcorn
784          real*4   arx(100),ary(100),cornr(10),cornc(10),avex
785          real*4   avey,xtranspt(25),ytranspt(25)
786          real*8   p1,p2,p3,p4,p5,p6
787          common /trenddat/arx,ary,trend
788          common /maxima/cornr,cornc
789
790 c ..... This routine takes 4 pts at a time from the arrays ARX(),ARY()
791 c (Each group of 4 represents a transition point area.), and
792 c performs a linear interpolation from the first 2 pts. to the
793 c second 2 pts. to find their linear intersection. This
794 c intersection is a transition point whose coordinates are
795 c stored in XTRANSPT(),YTRANSPT(). The type of trend associated
796 c with this point is the one the point is changing from (i.e. if
797 c transition is from a TRANS()=1 to a TRANS()=-1, the trend of
798 c the transition pt. is TTRANSPT()=1.)

```

```

D Line# 1      7      Microsoft FORTRAN77 V3.30 March 1985
799 c      Once all transition pts. are found (NN=# transition pts.),
800 c      those indicating corners (TTRANSPT()==1) have their coordinates
801 c      stored in CORNR(), CORNC().
802
803      avex=0.0
804      avey=0.0
805      p1=0.0
806      p2=0.0
807      p3=0.0
808      p4=0.0
809      p5=0.0
810      p6=0.0
811      nn=0
812
813 c ..... Storage space for up to 25 transition points.
814
815      do 210 i=1,25
1 816          xtranspt(i)=0.0
1 817          ytranspt(i)=0.0
1 818          ttranspt(i)=0
1 819      210  continue
820
821      inittrd=0
822      do 300 i=1,tranpts-4,4
1 823          if(i.eq.1)then
1 824              if(trend(2).eq.1)then
1 825                  inittrd=1
1 826              else
1 827                  inittrd=2
1 828              endif
1 829          endif
1 830          if(trend(i+1).eq.1)then
1 831              p1=(arx(i+3)*ary(i+2)-arx(i+2)*ary(i+3))*(ary(i)-ary(i+1))
1 832              p2=(arx(i+1)*ary(i)-arx(i)*ary(i+1))*(ary(i+2)-ary(i+3))
1 833              p3=(arx(i)-arx(i+1))*(ary(i+2)-ary(i+3))
1 834              p4=(arx(i+2)-arx(i+3))*(ary(i)-ary(i+1))
1 835              avey=(p1-p2)/(p3-p4)
1 836              p5=(arx(i)-arx(i+1))*(arx(i+3)*ary(i+2)-arx(i+2)*ary(i+3))
1 837              p6=(arx(i+2)-arx(i+3))*(arx(i+1)*ary(i)-arx(i)*ary(i+1))
1 838              avex=(p5-p6)/(p3-p4)
1 839          else
1 840              avex=(arx(i+1)+arx(i+2))/2
1 841              avey=(ary(i+1)+ary(i+2))/2
1 842          endif
1 843          nn=nn+1
1 844          xtranspt(nn)=avex
1 845          ytranspt(nn)=avey
1 846          ttranspt(nn)=trend(i+1)
1 847      300  continue
848
849 c ..... Separate maxima(corners) from minima (side midpoints).
850 c      Store row and column coordinates of each maxima in
851 c      CORNR() and CORNC(), resp.
852
853      do 307 i = 1,10
1 854          cornr(i)=0.0

```

```

D Line# 1      7
1 855          cornc(i)=0.0
1 856 307      continue
1 857          do 308 i=inittrd,nn,2
1 858              numcorn=numcorn+1
1 859              cornr(numcorn)=xtranspt(i)
1 860              cornc(numcorn)=ytranspt(i)
1 861 308      continue
1 862          return
1 863          end

```

Name	Type	Offset	P	Class
ARX	REAL	0		/TRENDD/
ARY	REAL	400		/TRENDD/
AVEX	REAL	830		
AVEY	REAL	834		
CORNC	REAL	40		/MAXIMA/
CORNR	REAL	0		/MAXIMA/
I	INTEGER*2	886		
INITTR	INTEGER*2	888		
NN	INTEGER*2	4	*	
NUMCOR	INTEGER*2	894		
P1	REAL*8	838		
P2	REAL*8	846		
P3	REAL*8	854		
P4	REAL*8	862		
P5	REAL*8	870		
P6	REAL*8	878		
TRANPT	INTEGER*2	0	*	
TREND	INTEGER*2	800		/TRENDD/
TTRANS	INTEGER*2	580		
XTRANS	REAL	630		
YTRANS	REAL	730		

```

864
865
866      subroutine quad(sidetol,xg,yg)
867          integer*2  obj,flag1,isoside
868          real*8     xg,yg
869          real*4     cornr(10),cornc(10),dist(4),minpos,pi
870          real*4     topl,top2,top3,top4,bot1,bot2,bot3,bot4
871          real*4     diag1,diag2,diag3,diag4,minddiag,maxdiag,diagdif
872          real*4     diff1,diff2,angl1,angl2,angl3,angl4
873          real*4     sidetol,minside,sidedif,maxside
874          common /maxima/cornr,cornc
875          common /sdata/dist,minpos,sidedif,maxside,minside,obj,isoside,
876          &diagdif
877
878 c ..... Routine calculates the length of each side (DIST(1)-DIST(4))
879 c and the length of each diagonal distance from the geometric
880 c center XG,YG to each corner (DIAG1-DIAG4).
881 c MINSIDE & MAXSIDE = shortest & longest sides.
882 c MINDIAG & MAXDIAG = shortest & longest diagonals.
883 c SIDEDIF = difference between MINSIDE & MAXSIDE.
884 c DIAGDIF = difference between MINDIAG & MAXDIAG.

```

Program: test_image

Page 21

01-23-89

08:56:50

```
D Line# 1      7      Microsoft FORTRAN77 V3.30 March 1985
885 c      SIDETOL = % tolerance allowed for two lengths to be equal.
886 c      If the sides are equal (within SIDETOL), and their diagonals
887 c      area also equal (within SIDETOL), the object is a square
888 c      (OBJ=10). IF the sides are not equal (outside SIDETOL), the
889 c      opposite sides are checked for equality. If the pairs of
890 c      sides are equal (within SIDETOL), and the pairs of diagonals
891 c      are also equal, the object is a rectangle (OBJ=12). If
892 c      the object is neither of the above, it is unknown (OBJ=1).
893 c      The orientation of the object is calculated and the smallest
894 c      positive angle formed between the top of the screen and any
895 c      side is stored in MINPOS.
896
897      pi=3.14159265
898      angl1=0.0
899      angl2=0.0
900      angl3=0.0
901      angl4=0.0
902      sidedif=0.0
903      flag1=0
904      top1=cornr(2)-cornr(1)
905      top2=cornr(3)-cornr(2)
906      top3=cornr(4)-cornr(3)
907      top4=cornr(1)-cornr(4)
908      bot1=cornc(2)-cornc(1)
909      bot2=cornc(3)-cornc(2)
910      bot3=cornc(4)-cornc(3)
911      bot4=cornc(1)-cornc(4)
912      dist(1)=sqrt(top1**2+bot1**2)
913      dist(2)=sqrt(top2**2+bot2**2)
914      dist(3)=sqrt(top3**2+bot3**2)
915      dist(4)=sqrt(top4**2+bot4**2)
916      maxside=amax1(dist(1),dist(2),dist(3),dist(4))
917      minside=amin1(dist(1),dist(2),dist(3),dist(4))
918      sidedif=(maxside-minside)/minside
919
920 c ..... diag1 to diag4 = distance from (xg,yg) to each corner.
921
922      diag1=sqrt((cornr(1)-xg)**2+(cornc(1)-yg)**2)
923      diag2=sqrt((cornr(2)-xg)**2+(cornc(2)-yg)**2)
924      diag3=sqrt((cornr(3)-xg)**2+(cornc(3)-yg)**2)
925      diag4=sqrt((cornr(4)-xg)**2+(cornc(4)-yg)**2)
926      mindiag=amin1(diag1,diag2,diag3,diag4)
927      maxdiag=amax1(diag1,diag2,diag3,diag4)
928      diagdif=(maxdiag-mindiag)/mindiag
929
930 c ..... If the difference between longest & shortest sides (sidedif)
931 c      is within the prescribed tolerance sidetol, the object
932 c      could be a square if the length of each diagonal (diagdif)
933 c      are also equal.
934
935      obj=1
936      if(sidedif.le.(sidetol/100))then
937      if(diagdif.le.(sidetol/100))then
938
939 c ..... Sides and diagonals are both equal, therefore object is a
940 c      square (10)
```

D Line# 1 7

Microsoft FORTRAN77 V3.30 March 1985

```
941
942      obj=10
943    endif
944  else
945
946 c ..... Object is not square. Check opposite sides for equality.
947 c      If opposite sides are within tolerance h, they are equal.
948
949      diff1=(abs(dist(3)-dist(1)))/dist(1)
950      diff2=(abs(dist(4)-dist(2)))/dist(2)
951      if((diff1.le.sidetol).and.(diff2.le.sidetol))then
952        if(diagdif.le.(sidetol/100))then
953
954 c ..... Opposite sides are equal, and diagonals are equal.
955 c      Therefore object is a rectangle (12).
956
957      obj=12
958    endif
959  endif
960 endif
961
962 c ..... Calculate angle of orientation: defined as the minimum
963 c      positive angle between top of screen and any side.
964
965      if((top1.eq.0.0).or.(bot2.eq.0.0))then
966        angl1=0.0
967        angl2=90.0
968        angl3=180.0
969        angl4=270.0
970        flag1=1
971      else
972        if((bot1.eq.0.0).or.(top2.eq.0.0))then
973          angl1=90.0
974          angl2=180.0
975          angl3=270.0
976          angl4=360.0
977          flag1=1
978        endif
979      endif
980      if(flag1.ne.1)then
981        angl1=atan2(top1,bot1)
982        angl2=atan2(top2,bot2)
983        angl3=atan2(top3,bot3)
984        angl4=atan2(top4,bot4)
985        angl1=angl1*180/pi
986        angl2=angl2*180/pi
987        angl3=angl3*180/pi
988        angl4=angl4*180/pi
989      endif
990      if(angl1.ge.360.0)then
991        angl1=angl1-360.0
992      endif
993      if(angl2.ge.360.0)then
994        angl2=angl2-360.0
995      endif
996      if(angl3.ge.360.0)then
```

Program: test_image

Page 23

01-23-89

08:56:50

Microsoft FORTRAN77 V3.30 March 1985

```

D Line# 1      7
997      angl3=angl3-360.0
998      endif
999      if(angl4.ge.360.0)then
1000      angl4=angl4-360.0
1001      endif
1002      minpos=amin1(angl1,angl2,angl3,angl4)
1003      if(minpos.lt.0.0)then
1004      minpos=amax1(angl1,angl2,angl3,angl4)
1005      if((angl1.ge.0.0).and.(angl1.lt.minpos))then
1006      minpos=angl1
1007      endif
1008      if((angl2.ge.0.0).and.(angl2.lt.minpos))then
1009      minpos=angl2
1010      endif
1011      if((angl3.ge.0.0).and.(angl3.lt.minpos))then
1012      minpos=angl3
1013      endif
1014      if((angl4.ge.0.0).and.(angl4.lt.minpos))then
1015      minpos=angl4
1016      endif
1017      endif
1018      return
1019      end

```

Name	Type	Offset	P	Class
ABS				INTRINSIC
AMAX1				INTRINSIC
AMIN1				INTRINSIC
ANGL1	REAL	900		
ANGL2	REAL	904		
ANGL3	REAL	908		
ANGL4	REAL	912		
ATAN2				INTRINSIC
BOT1	REAL	934		
BOT2	REAL	938		
BOT3	REAL	942		
BOT4	REAL	946		
CORNC	REAL	40		/MAXIMA/
CORNR	REAL	0		/MAXIMA/
DIAG1	REAL	950		
DIAG2	REAL	954		
DIAG3	REAL	958		
DIAG4	REAL	962		
DIAGDI	REAL	36		/SDATA /
DIFF1	REAL	974		
DIFF2	REAL	978		
DIST	REAL	0		/SDATA /
FLAG1	INTEGER*2	916		
ISOSID	INTEGER*2	34		/SDATA /
MAXDIA	REAL	970		
MAXSID	REAL	24		/SDATA /
MINDIA	REAL	966		
MINPOS	REAL	16		/SDATA /
MINSID	REAL	28		/SDATA /
OBJ	INTEGER*2	32		/SDATA /

Program: test_image

Page 24

01-23-89

08:56:50

Microsoft FORTRAN77 V3.30 March 1985

```
D Line# 1      7
PI      REAL      896
SIDEI    REAL      20  /SDATA /
SIDEI    REAL      0  *
SQRT
TOP1     REAL      918  INTRINSIC
TOP2     REAL      922
TOP3     REAL      926
TOP4     REAL      930
XG       REAL*8      4  *
YG       REAL*8      8  *

1020
1021      subroutine triangle(sidetol,xg,yg)
1022          integer*2  obj, isoside
1023          real*8      xg,yg
1024          real*4      cornr(10),cornc(10),dist(4),minpos,pi
1025          real*4      mdif1,mdif2,mdif3,top1,top2,top3
1026          real*4      bot1,bot2,bot3,angl1,angl2,angl3,s1,s2
1027          real*4      sidetol,maxside,minside,sidedif,cornup,diagdif
1028          common /maxima/cornr,cornc
1029          common /sdata/dist,minpos,sidedif,maxside,minside,obj,isoside,
1030              &diagdif
1031
1032 c ..... Routine is similar to that in subroutine QUAD above.
1033
1034          pi=3.14159265
1035          mdif1=0.0
1036          mdif2=0.0
1037          mdif3=0.0
1038          isoside=0
1039          sidedif=0.0
1040          top1=cornr(2)-cornr(1)
1041          top2=cornr(3)-cornr(2)
1042          top3=cornr(1)-cornr(3)
1043          bot1=cornc(2)-cornc(1)
1044          bot2=cornc(3)-cornc(2)
1045          bot3=cornc(1)-cornc(3)
1046          dist(1)=sqrt(top1**2+bot1**2)
1047          dist(2)=sqrt(top2**2+bot2**2)
1048          dist(3)=sqrt(top3**2+bot3**2)
1049          maxside=amax1(dist(1),dist(2),dist(3))
1050          minside=amin1(dist(1),dist(2),dist(3))
1051
1052 c ..... sidedif is difference between longest and shortest sides.
1053
1054          sidedif=(maxside-minside)/minside
1055          obj=2
1056          if(sidedif.le.(sidetol/100))then
1057 c ..... If all sides are equal (within tolerance sidetol) object
1058 c ..... is an equilateral triangle.
1059
1060              obj=20
1061          else
1062 c ..... Check for two sides within tolerance. If two sides are
1063 c ..... equal the object is an isosceles triangle.
```

```

1064      s1=(abs(dist(1)-dist(2)))/dist(1)
1065      s2=(abs(dist(1)-dist(2)))/dist(2)
1066      sidedif=amax1(s1,s2)
1067      if(sidedif.le.(sidetol/100))then
1068          isoside=1
1069          obj=21
1070      else
1071          s1=(abs(dist(2)-dist(3)))/dist(2)
1072          s2=(abs(dist(2)-dist(3)))/dist(3)
1073          sidedif=amax1(s1,s2)
1074          if(sidedif.le.(sidetol/100))then
1075              isoside=2
1076              obj=21
1077          else
1078              s1=(abs(dist(3)-dist(1)))/dist(3)
1079              s2=(abs(dist(3)-dist(1)))/dist(1)
1080              sidedif=amax1(s1,s2)
1081              if(sidedif.le.(sidetol/100))then
1082                  isoside=3
1083                  obj=21
1084              else
1085                  obj=2
1086              endif
1087          endif
1088      endif
1089      cornup=amax1(cornr(1),cornr(2),cornr(3))
1090
1091 c ..... Calculate the angle of orientation. This is defined as
1092 c the minimum positive angle between the top of the screen
1093 c and a side.
1094
1095      if(cornup.eq.cornr(1))then
1096          if(bot1.eq.0.0)then
1097              angl1=90.0
1098              angl2=210.0
1099              angl3=330.0
1100              flag1=1
1101          else
1102              if(bot3.eq.0.0)then
1103                  angl1=30.0
1104                  angl2=150.0
1105                  angl3=270.0
1106                  flag1=1
1107              endif
1108          endif
1109      endif
1110      if(cornup.eq.cornr(2))then
1111          if(bot1.eq.0.0)then
1112              angl1=270.0
1113              angl2=30.00
1114              angl3=150.0
1115              flag1=1
1116          else
1117              if(bot2.eq.0.0)then
1118                  angl1=330.0
1119                  angl2=90.00

```

Program: test_image

Page 26

01-23-89

08:56:50

D Line# 1 7

Microsoft FORTRAN77 V3.30 March 1985

```
1120      angl3=210.0
1121      flag1=1
1122      endif
1123    endif
1124  endif
1125  if(cornup.eq.cornr(3)) then
1126    if(bot2.eq.0.0) then
1127      angl1=150.0
1128      angl2=270.0
1129      angl3=30.0
1130      flag1=1
1131    else
1132      if(bot3.eq.0.0) then
1133        angl1=210.0
1134        angl2=330.0
1135        angl3=90.0
1136        flag1=1
1137      endif
1138    endif
1139  endif
1140  if(flag1.eq.0) then
1141    angl1=atan2(top1,bot1)
1142    angl2=atan2(top2,bot2)
1143    angl3=atan2(top3,bot3)
1144    angl1=angl1*180/pi
1145    angl2=angl2*180/pi
1146    angl3=angl3*180/pi
1147  endif
1148  if(angl1.ge.360.0) then
1149    angl1=angl1-360.0
1150  endif
1151  if(angl2.ge.360.0) then
1152    angl2=angl2-360.0
1153  endif
1154  if(angl3.ge.360.0) then
1155    angl3=angl3-360.0
1156  endif
1157  minpos=amin1(angl1,angl2,angl3)
1158  if(minpos.lt.0.0) then
1159    minpos=amax1(angl1,angl2,angl3)
1160    if((angl1.ge.0.0).and.(angl1.lt.minpos)) then
1161      minpos=angl1
1162    endif
1163    if((angl2.ge.0.0).and.(angl2.lt.minpos)) then
1164      minpos=angl2
1165    endif
1166    if((angl3.ge.0.0).and.(angl3.lt.minpos)) then
1167      minpos=angl3
1168    endif
1169  endif
1170  return
1171  end
```

Name Type Offset P Class

ABS INTRINSIC

Program: test_image

Page 27

01-23-89

08:56:50

Microsoft FORTRAN77 V3.30 March 1985

D Line# 1	7		
AMAX1			INTRINSIC
AMIN1			INTRINSIC
ANGL1	REAL	1034	
ANGL2	REAL	1038	
ANGL3	REAL	1042	
ATAN2			INTRINSIC
BOT1	REAL	1010	
BOT2	REAL	1014	
BOT3	REAL	1018	
CORNC	REAL	40	/MAXIMA/
CORNR	REAL	0	/MAXIMA/
CORNUP	REAL	1030	
DIAGDI	REAL	36	/SDATA /
DIST	REAL	0	/SDATA /
FLAG1	REAL	1046	
ISOSID	INTEGER*2	34	/SDATA /
MAXSID	REAL	24	/SDATA /
MDIF1	REAL	986	
MDIF2	REAL	990	
MDIF3	REAL	994	
MINPOS	REAL	16	/SDATA /
MINSID	REAL	28	/SDATA /
OBJ	INTEGER*2	32	/SDATA /
PI	REAL	982	
S1	REAL	1022	
S2	REAL	1026	
SIDEDI	REAL	20	/SDATA /
SIDETO	REAL	0 *	
SQRT			INTRINSIC
TOP1	REAL	998	
TOP2	REAL	1002	
TOP3	REAL	1006	
XG	REAL*8	4 *	
YG	REAL*8	8 *	

1172

Name	Type	Size	Class
ALLTRD			SUBROUTINE
CENTRO			SUBROUTINE
CORNER			SUBROUTINE
DELTAD		4000	COMMON
EDGEDA		8002	COMMON
FIRSTT			SUBROUTINE
GRADIE			SUBROUTINE
MAXIMA		80	COMMON
QUAD			SUBROUTINE
RADIDA		8000	COMMON
RADIUS			SUBROUTINE
REPONE			SUBROUTINE
REPZER			SUBROUTINE
SAVCIR			SUBROUTINE
SAVEDA			SUBROUTINE
SDATA		40	COMMON

Program: test_image

Page 28

01-23-89

08:56:50

Microsoft FORTRAN77 V3.30 March 1985

D Line# 1 7

TESTLM

TRDDAT

TRENDI

TRIANG

10
1000

PROGRAM
COMMON
COMMON
SUBROUTINE

Pass One No Errors Detected
1172 Source Lines