

To the Graduate Council:

I am submitting herewith a thesis written by Mongi Abidi entitled "Shape and Area Correlation in Time-Varying Imagery." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.



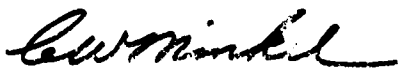
R. C. Gonzalez, Major Professor

We have read this thesis
and recommend its acceptance:





Accepted for the Council:



The Graduate School

SHAPE AND AREA CORRELATION
IN
TIME-VARYING IMAGERY

A Thesis
Presented for the
Master of Science
Degree
The University of Tennessee, Knoxville

Mongi Abidi
March 1985

ACKNOWLEDGEMENT

This technical work summarizes the findings of 3 years of research in radar motion analysis done by the author at the Image and Pattern Analysis Laboratory at the University of Tennessee under the guidance of Professor R. C. Gonzalez. This work was partially supported by NOAA under Contract NA-82-SAC-00072.

The author wishes to express his deep gratitude to his advisor, Professor R. C. Gonzalez, for suggesting the topic of this work and for his help and encouragement. Thanks are also due to NOAA and especially to Mr. Ken Shreeve and Mr. Bob Elvander for their helpful discussions with me. Special thanks are due to Ms. Roberta Campbell and Ms. Denise Smiddy for their highly professional typing.

ABSTRACT

In this work we present a global scheme for motion detection and characterization in radar images. It involves 2 steps: (1) each image is rapidly scanned using shape extraction and recognition to extract important clues regarding areas of major changes; (2) using the latter clues, a local analysis based on correlation then is applied to selected areas of the image. This technique was applied to a sequence of real radar cloud images where circulation was suspected; the results showed clear automatic detection of this circulation. The algorithms used in this study and their formulation are fast enough to envision real time tracking of clouds.

TABLE OF CONTENTS

CHAPTER	PAGE
1. INTRODUCTION	1
1.1 Background	1
1.2 General Thunderstorm Elements	7
1.3 Recognition Approach	13
2. SHAPE DESCRIPTION AND MATCHING	25
2.1 Fourier Descriptors of a Curve	25
2.2 Discrete Fourier Descriptors from a Polygonal Curve	30
2.3 Relationship Between Fourier Descriptors and Geometric Shapes	39
2.4 Computational Considerations	44
2.5 Elliptic Description of Contours	62
2.6 Approximation of Curves by a Finite Number of Harmonics	67
2.7 Elliptic Properties of the Fourier Features	74
2.8 Normalization of the Elliptic Descriptors	79
2.9 Illustrative Example	82
3. MOTION DETECTION VIA AREA CORRELATION	89
3.1 Tracking by Correlation	89
3.2 Cross-Correlation Coefficient in Discrete Images	92
3.3 Detection of Simulated Motion in Random Texture Images	103
3.4 Detection of Simulated Motion in Real Radar Images	109
4. SHAPE ANALYSIS AND CORRELATION IN CIRCULATION DETECTION	113
4.1 Shape Extraction and Matching	113
4.2 Air Flow by Correlation	118
4.3 Measurement of Circulation in Cloud Images	121
5. CONCLUSIONS	123
LIST OF REFERENCES	124

TABLE OF CONTENTS (continued)

CHAPTER	PAGE
APPENDICES	129
APPENDIX A: Correlation	129
APPENDIX B: Shape	139
APPENDIX C: Input/Output	169
VITA	184

LIST OF FIGURES

FIGURE	PAGE
1.2-1 Low-level plan view of mature supercell.	12
1.2-2 Scheme of data collection.	12
1.3-1 Tornado detection and location.	14
1.3-2 Search area reduction by intensity thresholding and characteristic shape detection.	17
1.3-3 Sample image quantized at different levels of intensities.	19
1.3-4 Image from Fig. 1.3-3 quantized in a square grid.	19
1.3-5 Map of the boundary lines of equal intensities areas from Fig. 1.3-4.	20
1.3-6 Chain forms generated from the contours in Fig. 1.3-5.	20
1.3-7 Search area reduction by the Minimum-Distance Classifier for template (a",b",c",d") in initial area (a',b'c',d').	21
1.3-8 Tornado location: Search-area reduction by Dismissal Probability Matrix	22
1.3-9 Tornado location: TREC Vector generation for template cell.	23
2.1-1 Rectangular and polar representation of planar curves.	26
2.1-2 Parametric representation angle versus arc length of planar curves.	26
2.2-1 Approximation of curves by polygon series.	32
2.4-1 Configuration of the neighborhood of a multiple pixel for 8-connected contours.	47

FIGURE	PAGE
2.4-2 Configuration of the neighborhood of a multiple pixel for 4-connected contours.	48
2.4-3 Chain-code link labels for (a) 8-connected chains, (b) 4-connected chains.	50
2.4-4 Example of (a) 8-neighbor, (b) 4-neighbor coded chains from the contour of a circular object.	50
2.4-5 Example of 8-neighbor code chain from Fig. 1.3-6.	52
2.9-1 Decomposition of a square in harmonics.	84
2.9-2 Projection $x(t)$.	84
2.9-3 Projection $y(t)$.	84
2.9-4 Phasor decomposition	88
3.2-1 Initial image P_1 .	93
3.2-2 Search image P_2 .	94
3.2-3 Representation of a storm scanned by a radar located at the origin.	95
3.3-1 (a) Initial noise image, (b) Manual rotation simulation from (a), Difference-image from (a) and (b).	104
3.3-2 Circulation detection in noise image using (a) 7 by 7 Operator, (b) 5 by 5 Operator, (c) 3 by 3 Operator and (d) 2 by 2 Operator.	105
3.3-3 Initial noise image, (b) Computer simulated rotation from (a), (c) Difference-image from (a) and (b).	107
3.3-4 Circulation detection of computerized rotation (a) 7 by 7 Operator, (b) 5 by 5 Operator, (c) 3 by 3 Operator, and 2 by 2 Operator.	108
3.4-1 (a) State of a storm shortly before the occurrence of 3 tornadoes, (b) A computerized rotation simulation from (a), (c) Difference-image from (a) and (b).	110

FIGURE	PAGE
3.4-2 Circulation detection in radar images (a) 7 by 7 Operator, (b) 5 by 5 Operator, (c) 3 by 3 Operator, and 2 by 2 Operator.	111
4.1-1 Sequence of coded Radar scans.	115
4.1-2 Monochrome reflectivity display from (1).	115
4.1-3 Hook-like shapes in mature stage.	115
4.1-4 Four-image sequence from (3).	115
4.1-5 Coded display showing mature hook-like echo.	115
4.1-6 Display of the sequence shown in (1) for $VIP \geq 4$.	115
4.1-7 Hook contours extracted from Fig. 4.1-5 for 3 distinct VIP values.	116
4.1-8 A single contour extracted from Fig. 4.1-5.	116
4.1-9 Examples of library prototype contours.	117
4.2-1 TREC Vectors of image #1 and #2 from Fig. 4.1-1.	119
4.2-2 TREC Vectors of image #2 and #3 from Fig. 4.1-1.	119
4.2-3 Air Flow in a Tornado model.	120
4.2-4 Typical Example of Air Flow Map derived from the case of April 10, 1979.	120
4.2-5 Air Flow Map superimposed to dominant Tornado features.	122
4.3-1 Circulation mapping from the data in Fig. 4.2-1.	122

CHAPTER 1

INTRODUCTION

1.1. Background

Time-varying imagery is one of the most recent disciplines in the newly emerging area of computer vision. This area is by itself a component of the relatively new aspects of computer applications. More and more, imaging sensors and processors are becoming an increasingly important segment of closed-loop controllers in automatic tracking, missile guidance, robotics, and many other intelligent civil and military applications. Motion measurement and interpretation always implies the processing of an image sequence in order to retrieve some properties of the imaged scenes which change in time. In this work, we concentrate on techniques which deal with motion measurement and interpretation using an illustrative example from weather forecasting. General procedures for detecting and characterizing certain types of motions are proposed. These procedures are versatile enough to be applied to various problems involving moving scenes. These techniques were evaluated with real radar images taken for weather forecasting purposes. In order to come up with adequate models and mathematical descriptions for this problem, one should be familiar with the physical phenomena we intend to describe. These phenomena include cloud formation and dynamics, wind movement, and cloud-precipitation relationships. One should be familiar also with sensors used in cloud tracking as well as display

techniques and methods of interpretation of sensed data, and finally terminology and definitions used in this field.

Tornado detection and location may be defined as the process through which an image sequence, generated by radar tracking a tornadic thunderstorm, is gradually modified by pattern-recognition and image-processing techniques to derive an air-flow map yielding the identification of areas of moderate or severe circulation. The use of radars to gain quantitative and qualitative information about storms and other weather phenomena dates back to the 1940's [1,2]. Early approaches to this problem consisted of manual computations and graphical techniques for tracking and forecasting storm behavior. During the 1960's and early 1970's, the combined availability of general-purpose digital computers and radars equipped with digital-signal processing capabilities introduced a fundamental change in the acquisition and analysis of weather-related data. These radars increased both the quantity and resolution of radar data, thus making possible the development of techniques for gaining more detailed information about the structure of weather phenomena. The extraction of this detailed information is often practical only through the use of digital computers due to the volume of data that needs to be processed, as well as the complexity of the algorithms used in the information extraction process.

Increased data volumes and more powerful digital processing capabilities lead in a natural way to the application of extracting information from radar signatures. Pattern recognition and image processing are the most prominent fields of work in autonomous or semi-

autonomous (i.e., human-guided) information extraction. In addition to data interpretation functions, pattern recognition and image processing techniques offer the potential for overall radar system simplification in the sense that information derived from these techniques can often reduce the number of independent radar signals required to compute parameters such as horizontal wind fields [12].

The first application of pattern recognition methods to weather-radar data appears to be a cross-correlation technique used by Hilst and Russo [3] which yielded a single vector giving storm direction and speed. This approach subsequently was refined by the introduction of storm statistical parameters such as storm length, width, and intensity [4]. Other techniques which are related to pattern recognition methods were reported in the early and mid 1970's. These included the work of Blackmer et al. [5] and Austin and Bellon [6], who used correlation coefficients, the optical correlation method of Zawadski [7], the Fourier-contour approach of Ostlund [8], and the bivariate-normal-distribution fitting method of Wiggert et al. [9]. Some of these early approaches are compared by Alaka et al. [10].

The most comprehensive studies related to the use of pattern recognition for tracking storm motions appear to be the work reported by Rinehart [11] and Symthe [12]. Rinehart used the correlation coefficient of reflectivity as the basis for extracting indications of internal storm motions from a single non-Doppler weather radar. Smythe used a similar approach with reflectivity and velocity data obtained from a single Doppler radar for retrieving the transverse wind. These reports

are particularly important because of their thoroughness and meaningful results. Their recentness is also a very useful indicator of the state of the art in the application of pattern recognition concepts to the quantification of storm motion.

The techniques presented can be viewed as an application in a broader framework of time-varying imaging where the data collected are presented as different frames of the same scene. Each frame is processed either individually or in conjunction with one or more of the previous frames to extract meaningful features. The features extracted vary from simple intensity descriptions to more advanced characterization of the scene, involving boundaries, texture analysis, etc. The time-varying imaging differs in its techniques from the traditional way of single-image processing by its use of results from previous frames to guide the feature extraction process in the rest of the frames [13]. The techniques used reduce to a matching problem, either to 2-D (area) or 1-D (segment) comparison. The first approach concerns area correlation. A rectangular area selected in an initial frame is searched in the following frames. A correlation model suitable for Pattern Recognition and Image Processing applications is introduced by Bogner [14]. This model provides us with a solid mathematical basis for the generating process of samples being tested. It also gives a strong theoretical background for the comparison of the covariance matrices using the Mahalanobis distance as a similarity measure. Sclove [15] completed and refined the approach in an elegant way by the consideration of an ultimate property of image data: interpixel correlation. The two dimensional Markov model takes

care of the information redundancy and straightens out the classification scheme. Similar ideas were put into practice via experimental work. Chen and Pavlidis [16] used a correlation-coefficient technique to analyze textures for segmentation purposes. Abidi and Gonzalez [17] and Rinehart [11] applied a similar technique to radar data. In general, area correlation always has been a dependable matching tool. Its easy implementation however, does not always compensate for a major drawback, namely, the processing time required.

The second approach taken to solve matching problems in image data concerns the use of single or one-dimensional features. This approach goes beyond the interpixel intensity comparison to more exhaustive operations such as shape characterization. This includes edge detection, boundary finding algorithms, shape segmentation, shape analysis, etc. Sobel, Robert and Hueckel operators, to name only a few, were used in several frame-to-frame analyses [17,18,19]. Boundary finding algorithms always were problem-oriented; nevertheless, their success was of great help. Elliott et al. [20] present a suitable model for the problems which fall within this scope. Shape analysis attacked the problem of image recognition on several fronts using shape descriptors. Pavlidis [21] has published an overall review of the algorithms developed in this manner. These techniques use global or local features. The most popular global shape descriptors are the Fourier descriptors [22,23,24] and the moments [25,26]. Rosenfeld [27]; Fukunaga, et al. [28,29] gave a general idea of the use of local shape descriptors in scene recognition and target tracking.

Among all these techniques only those which can break the trade-offs between the computational feasibility and theoretical optimality would be operational. Further, of those operational techniques only those which take a processing time on the order of the interframe time would be attractive methods to apply to time-varying-imagery type problems.

A more recent study in radar images by Abidi and Gonzalez [30] was based on the use of structural features for characterizing and detecting motion in weather radar images. The weather approaches discussed above have been applied only to the detection of large scale motions. The present work focuses on tornado detection and location in radar images using only reflectivity data.

We now outline the following sections. In the remainder of this chapter we introduce the major elements of thunderstorms to establish an understanding of the underlying physical processes. The last section of this chapter will be devoted to a global description of the recognition approach proposed in this work. As will be detailed in that section, the recognition approach can be subdivided into two major tasks, namely shape analysis and correlation. The theoretical aspects of these two tasks will be dealt with in the following two chapters. The techniques used for curve definition and coding, shape-feature extraction, and contour matching and recognition will be covered in Chapter 2. The technique of area correlation and its applications in cloud tracking will be the topic of Chapter 3. The last two sections of Chapter 3 present some experimental results intended to quantify the performance of the

techniques presented earlier in that chapter. In Chapter 4 we apply the techniques presented in Chapter 2 and 3 to a real-case tornado detection. First, characteristic shapes are extracted and recognized, and then the air-flow map of that case is generated using area matching by correlation.

1.2. General Thunderstorm Elements

Weather phenomena usually are classified according to the physical process immediately involved in their formation and, in some cases, by the flash flood potential of these phenomena. In this work, weather conditions will be related directly to their radar echoes. The properties of general thunderstorms and the tracking capabilities of weather radars have led to models for both severe and non-severe weather conditions. Ninety-nine percent of the total number of thunderstorms are benign [31]. Of those classified as severe (i.e., producing tornadoes, damaging surface winds, or large hail), our focus is on thunderstorms capable of generating tornadoes or any conditions favorable to the creation of a tornado.

For clarity, we relate the severity of a storm to a scale which translates the significance of the reflectivity of the radar to a degree of severity. The radar Video Integrator Processor (VIP) output ranges from 1 to 6 when the reflectivity varies in the regions separated by 0, 30, 41, 46, 50, 57, and 63, for an intensity range from 0 to 63 [31]. This could be translated in a simpler qualitative scale ranging through weak, moderate, strong, very strong, intense, or extreme categories of

storms. Although an ordinary thunderstorm is characterized by elements such as lightning, flash flooding, hail and waterspouts, these elements very seldom create favorable conditions for tornadoes. A typical VIP value of 4 is associated with all these conditions, and in the worst case, generates heavy rain and hail of a diameter less than or equal to 1 cm. A moderate level of turbulence excludes tornadoes, while an extreme level of turbulence gives a high probability of their occurrence. The least damaging of the severe thunderstorms is the pulse severe thunderstorm. This phenomenon can generate extreme turbulence and, hence, possible tornadoes. Their relatively short lifetime, however, makes tracking and recognition almost impossible.

The multicell severe thunderstorm lasts much longer than a pulse severe thunderstorm. The multiplicity of cells in this storm prevents its growth to a large size, resulting in poor radar reflections. This "blindness" of the radar is serious in terms of automated pictorial analysis of small scale circulations. As will be shown below, however, schemes based on large scale descriptions are feasible.

The last case of severe weather conditions treated in this work is the supercell severe thunderstorm. This type of storm is the most likely to generate funnel clouds or tornadoes. For example, 60% of the supercell thunderstorms that have occurred in the state of Oklahoma (an example used later in Chapters 3 and 4) produced funnel clouds or tornadoes [31]. The important features of this type of storm in terms of automated analyses are a lifetime of 6 hours or more and a movement slower than the environmental winds. The low relative speed tends to raise the

degree of success for a type-matching algorithm during the tracking operation. Furthermore, the relative position of the storm elements and their occurrence in position and time yield a reasonable forecast of the position and time of possible tornadoes. The straightforward relation between the VIP type of precipitation and radar reflectivity allows a determination of the zone where tornadoes are most likely to occur. The relative position of the different precipitations is fairly constant for a storm moving east or northeast. More steady is the relative position of the storm cells leading to the formation of a tornado.

1.2.1. The Tornado

A tornado can be defined as a violently rotating narrow column of air in contact with the ground and extending from a thunderstorm base. It is implied that this phenomenon is mixed with other different storm elements which are carried along by the violent spinning motion. Although a funnel cloud is different from a tornado by the fact that this phenomenon occurs in the tower or the base of the storm without extension to the ground, a radar model would be suitable for either one of them. Their appearance is basically the same down to the lowest detection elevation of a radar.

Tornadoes often are located beneath the right or rear periphery of the Wall Cloud (WC) generally separating the heavy precipitation zone in the northeast of the storm from the precipitation-free zone in the southwest. This observation is valid in the case of a storm whose overall motion is guided by a southwest/northeast axis. In fact, these

characteristics represent the typical storm behavior pattern over most of the United States. From previous studies and observations [31], it appears that a tornado cyclone often is detected by a radar as an annular section of a cylinder, a dot, or small spiral. This appearance certainly is affected by the scale of the tornado itself, its state of evolution, the distance between the tornado location and the radar station, the tilt, and the Plane Position Indicator (PPI).

In most radar images, a tornado is seen as a circulation which might be accompanied by other types of movements having their source in the general storm motion and not in the tornado phenomenon itself. Since the radar tracking procedure detects cloud movements at the base of the storm and gives no indication of their future behavior in a downward motion to ground, conditions leading to tornadoes are seen by a radar as large wind circulations (called cyclones). The detection and location of tornadoes is affected by a number of factors such as scale, severity, and lifetime. In addition, several special cases are important because they violate the commonly known relationships between these factors. For example, recent studies [31] indicate that tornadoes which caused death and property damage were associated with small scale cyclones of about 2-10 km (1-6) miles in radius. Some tornado cases are still mysterious despite the surveillance and data collection which accompanied their occurrence. Tornadoes have occurred in the early stages of a storm with and without severe weather conditions. Unlike most tornadoes, which occur in the left rear quadrant beneath

the Bounded Weak Echo Region [Fig 1.2-1], Hurricane tornadoes often occur in the right front quadrant relative to the movement of the storm [31].

1.2.2. Data Collection

The scheme used for data collection is fairly simple and is directly related to the radar beam scanning motion. The radar beam integrates through a gate length for a fixed elevation and azimuth. The integration result is associated with the corresponding gate. The latter operations are repeated for different gates at fixed azimuths. When the operation is completed for all the azimuths at a fixed elevation, all the preceding steps are repeated for another elevation. The first and last azimuths depend on the position of the storm with respect to the radar station. After a complete period, a G by A by E volumetric picture is obtained, where G is the number of gates along one azimuth, A is the number of azimuths scanned at a fixed elevation, and E is the total number of elevations.

Figure 1.2-2 illustrates the scanning geometry used to obtain the data utilized throughout this work. The gate length scanning along one radial was fixed to 0.998 km, and data were collected at a 2 degree interval for azimuths for the data used in Chapter 3 whereas in Chapter 4 these parameters were 0.600 km and 1 degree, respectively.

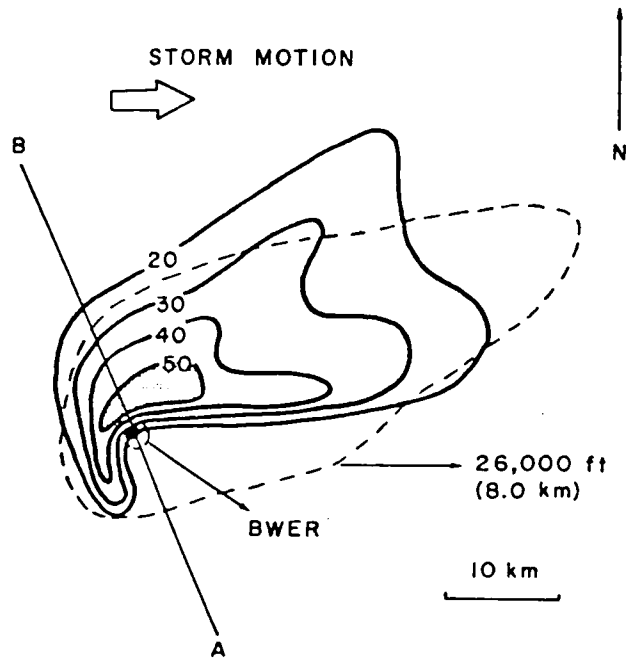


Figure 1.2-1 Low-level plan view of mature supercell.

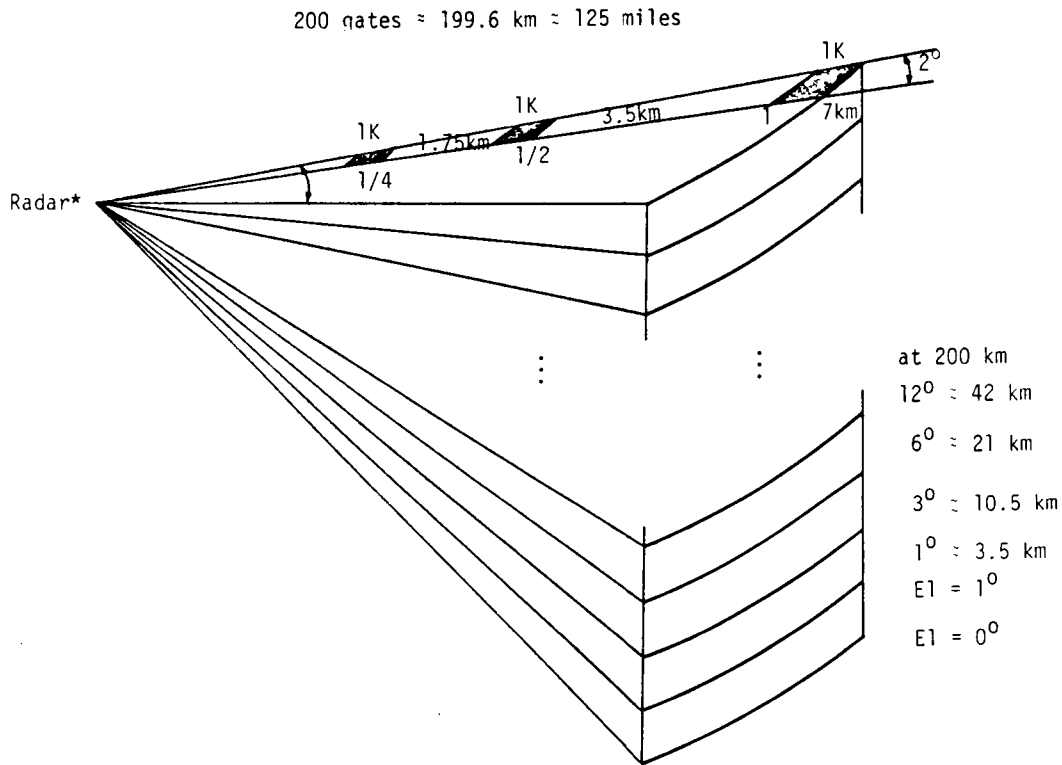
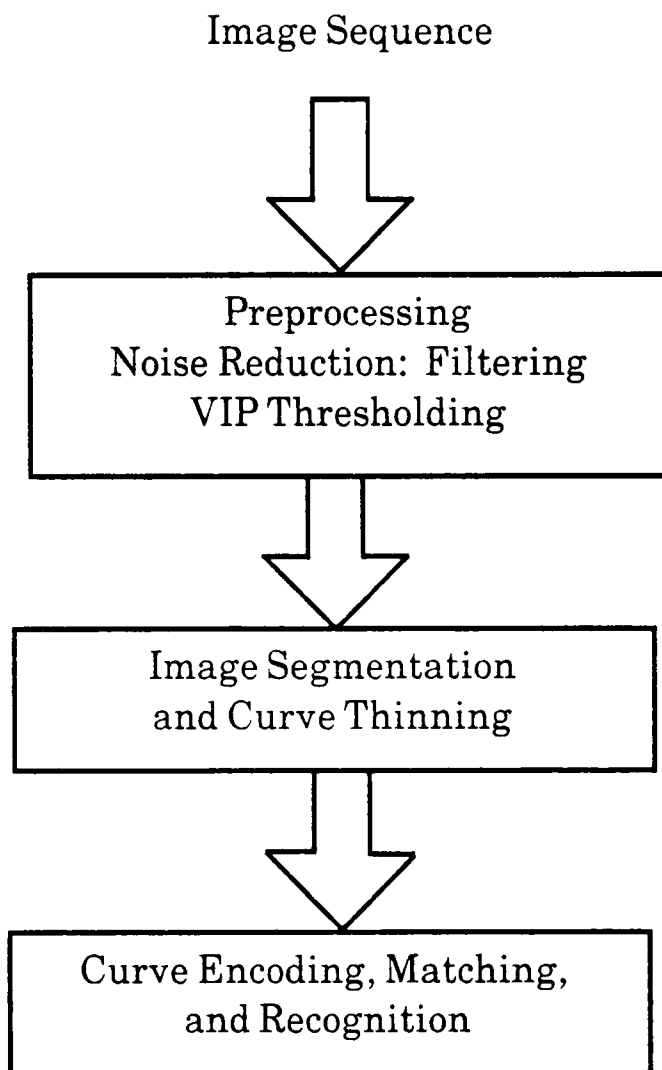


Figure 1.2-2 Scheme of data collection.

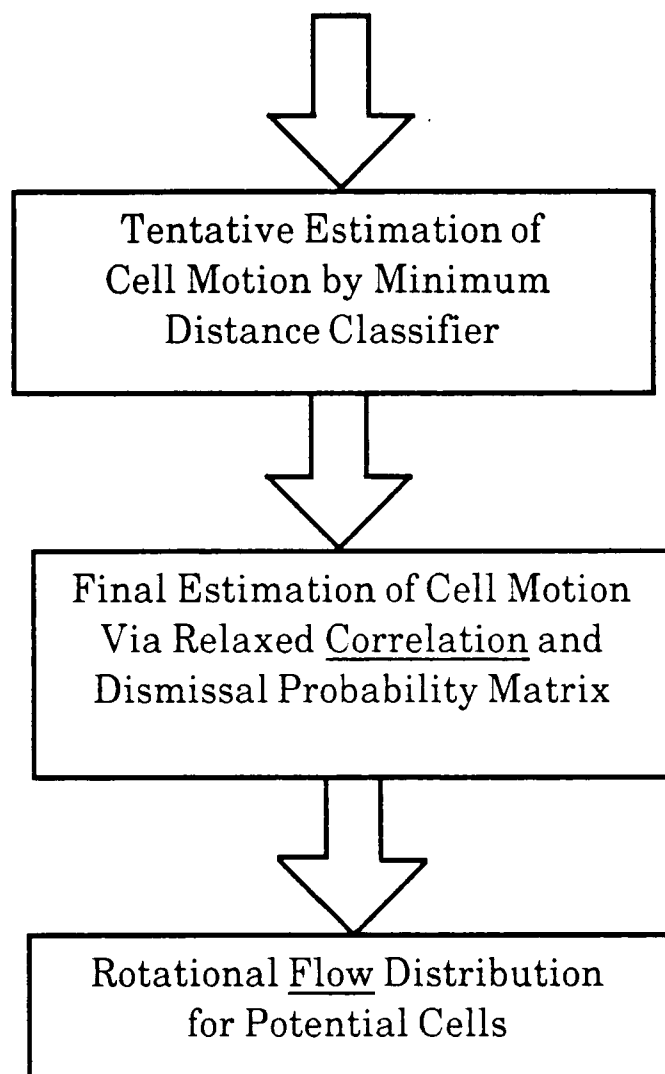
1.3. Recognition Approach

The recognition approach described in this work is summarized in Fig. 1.3-1. Each of the steps mentioned above may be subdivided into three subtasks, as shown in Fig. 1-3-1. With reference to Fig. 1.3-1(a), an image sequence is filtered by neighborhood averaging and thresholded at a fixed VIP (Video Integrator Processor) value. In the following experiments, intensities of $VIP \leq 2$ are considered too weak to generate severe weather conditions; consequently the image areas of such intensities are dropped in later analyses. Figure 1.3-2 illustrates the thresholding procedure and its consequences on later steps. Several studies [30,32] suggested that, in most cases, a parent thunderstorm takes the shape of a pendant or a hook at the time of tornado formation. The recognition of characteristic shapes can be used to guide the search process and improve its overall efficiency. In order to extract these characteristic shapes, the images in the processed sequence need to be segmented [18,19]. The resulting binary images are not always suitable for recognition because of noise and inconsistent features. Frequently, the segmentation process generates a few meaningless shapes of certain shades, but in most cases, cloud radar images are reasonably regular and allow the extraction of well-defined contours common to storm data. Depending on the steps used for image segmentation, the extracted contours are several pixels wide, making immediate description very difficult and thus requiring a thinning phase [33,34]. The thinning



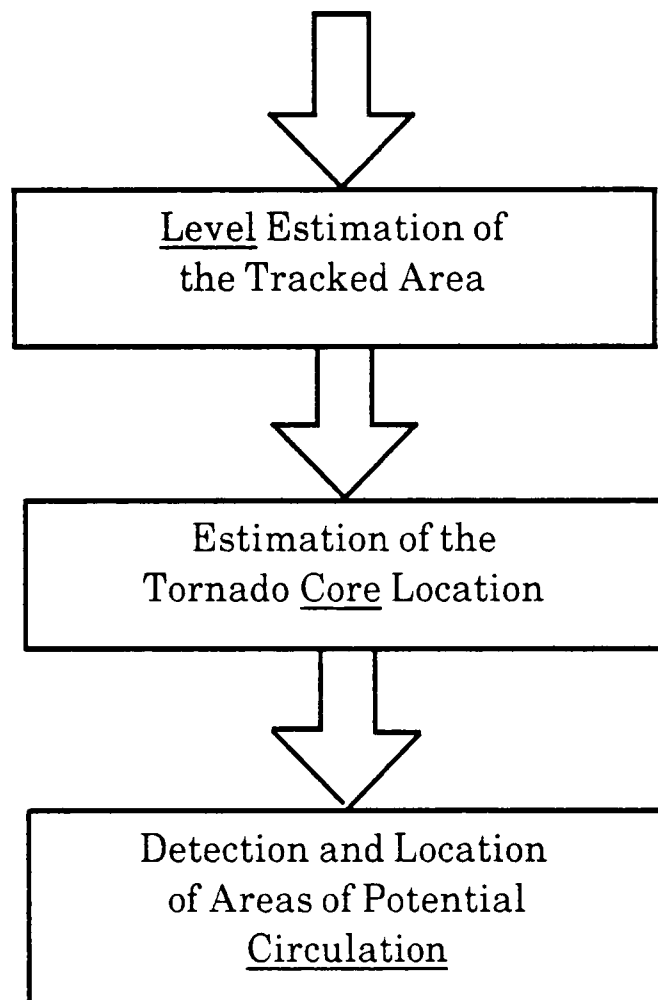
(a) Recognition of the limits of high tornadic activities.

Figure 1.3-1 Tornado detection and location.



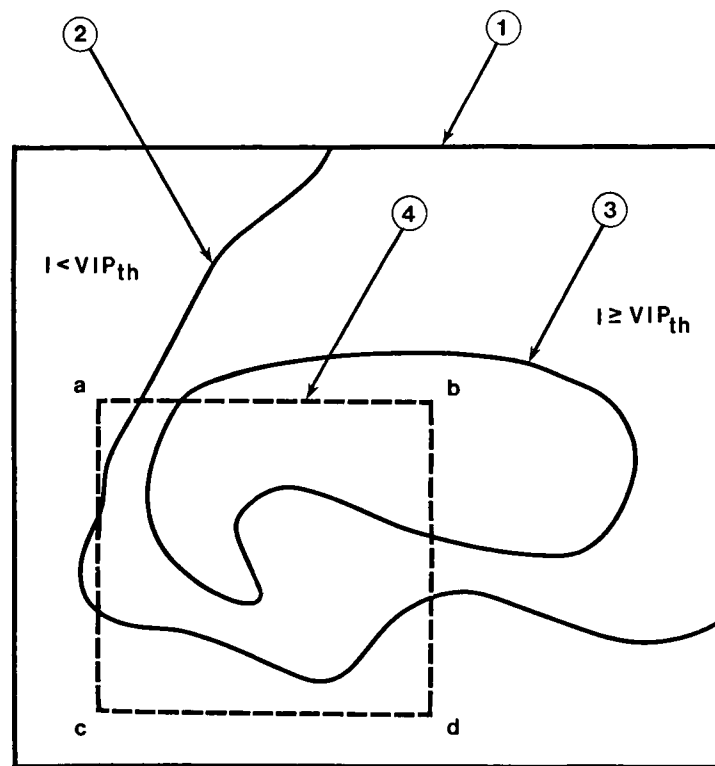
(b) Motion characterization for each template cell.

Figure 1.3-1 (continued).



(c) Results interpretation and decision matching point.

Figure 1.3-1 (continued).



1. Boundary of preprocessed image sequence.
2. Boundary from VIP thresholding.
3. Characteristic shape boundary.
4. Area of potential tornadic activities.

Figure 1.3-2. Search area reduction by intensity thresholding and characteristic shape detection.

algorithm developed for this purpose generates unique pathway contours of constant one-pixel width. This phase is illustrated with a synthetic image [35] and its possible variations of segmentation and generated paths [Fig. 1.3-3 through 1.3-6].

Beyond this step no more pictorial information is handled, but rather contours are stored as coded chains. Each contour is then matched against all the prototypes stored in a library provided for this purpose [36]. The subject of shape description, matching, and recognition will be covered in detail in Chapter 2. When one or several characteristic shapes are recognized, a square area around each pendant or hook shape is declared potentially tornadic. For a tornadic mother-supercell this area covers approximately 32 by 32 pixel squares or an area of about 12 miles on a side. Within this notably reduced area, search for circulation is undertaken to confirm or negate the assumptions resulting from shape analysis.

To characterize the motion of each cell in the area of interest, matching of a template cell and its neighbors is performed through the Minimum-Distance Classifier [22]. This is a pre-scan which eliminates poor matches. Afterwards, potential candidates are matched against the template cell using a relaxed, extensively overlapped correlation [30,38] that was developed by Abidi and Gonzalez [17]. When the correlation procedure is performed on each cell, an air flow map is obtained for the area suspected to contain tornadic activities [Fig. 3.1-1(b) (pp. 15) and Figs. 1.3-7 through 1.3-9]. After the air-flow map is generated, a visual or automatic interpretation is required to judge the potential for

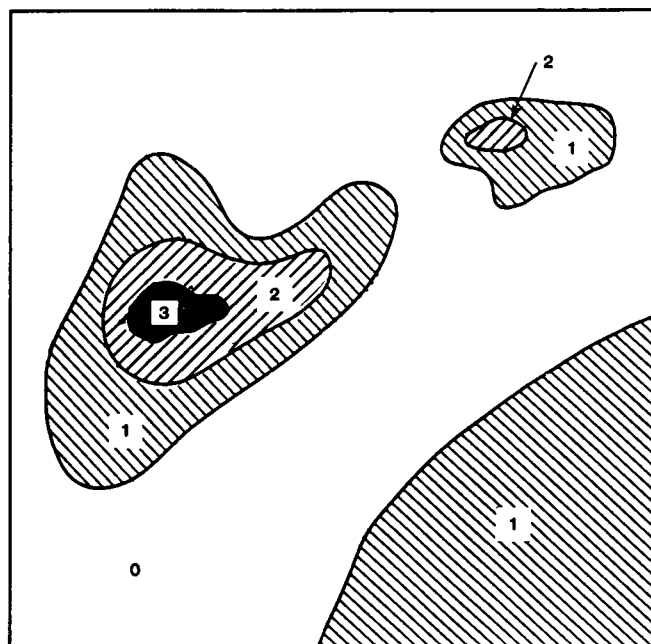


Figure 1.3-3. Sample image quantized at different levels of intensities.

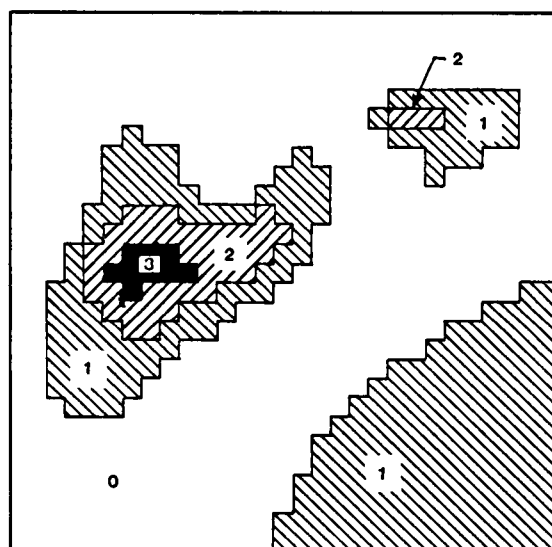


Figure 1.3-4. Image from Fig. 1.3-3 quantized in a square grid.

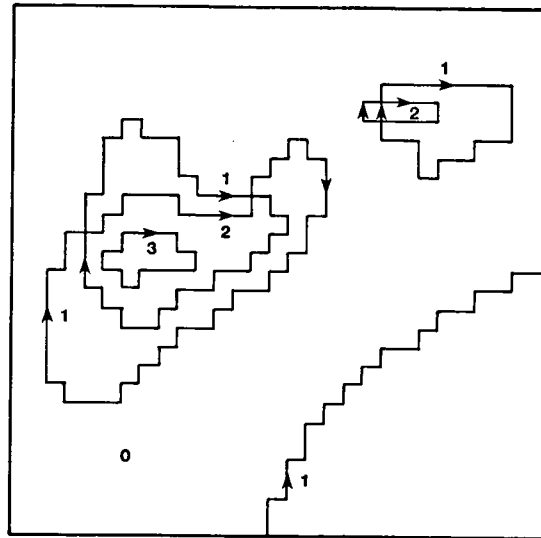


Figure 1.3-5. Map of the boundary lines of equal intensities areas from Fig. 1.3-4.

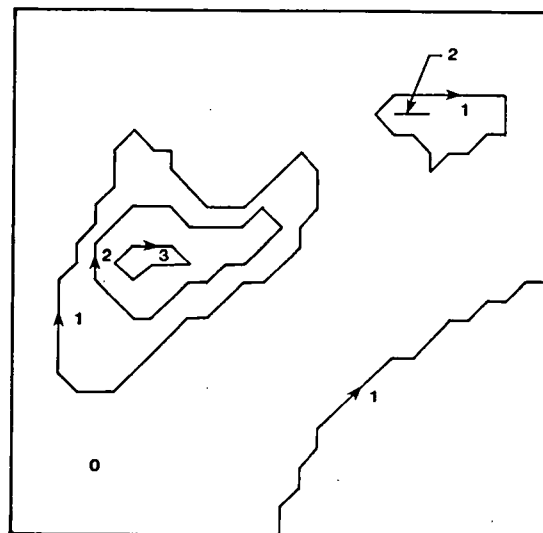
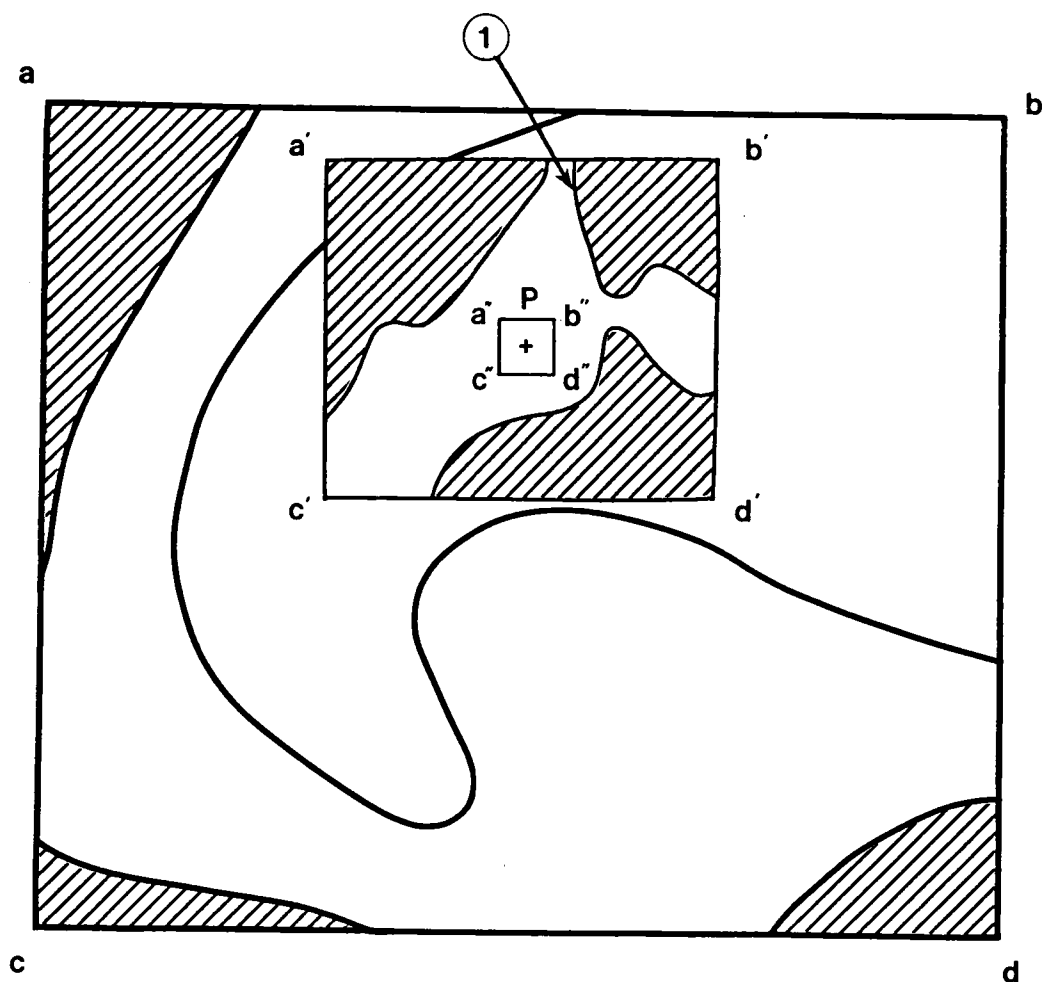
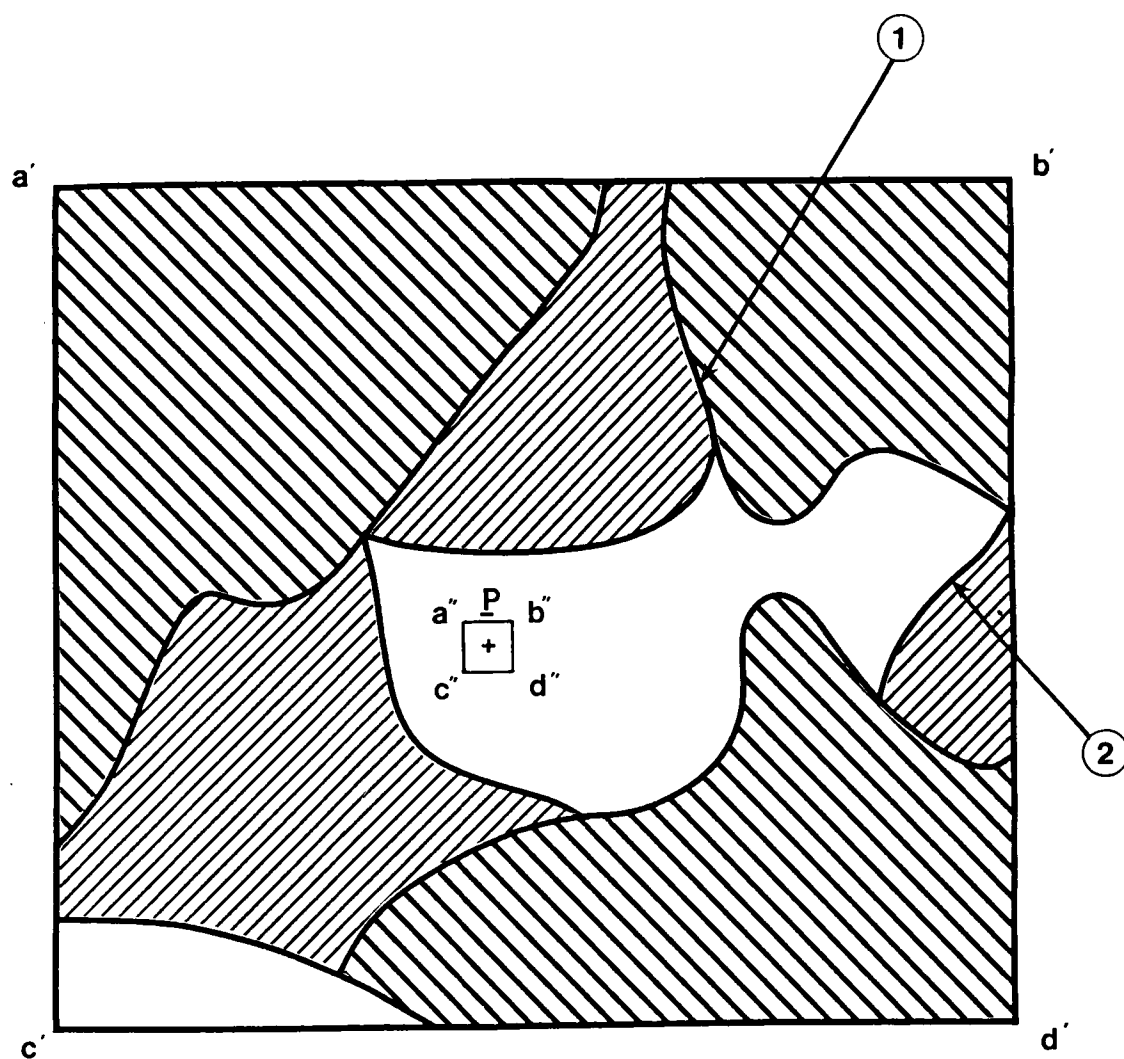


Fig. 1.3-6. Chain forms generated from the contours in Fig. 1.3-5.



1. Boundary from Minimum-Distance Classifier.

Figure 1.3-7. Search area reduction by the Minimum-Distance Classifier for template (a'' , b'' , c'' , d'') in initial area (a' , b' , c' , d').



1. Boundary from Minimum-Distance Classifier.
2. Boundary from Dismissal Probability Matrix.

Figure 1.3-8. Tornado location: Search area reduction by Dismissal Probability Matrix.

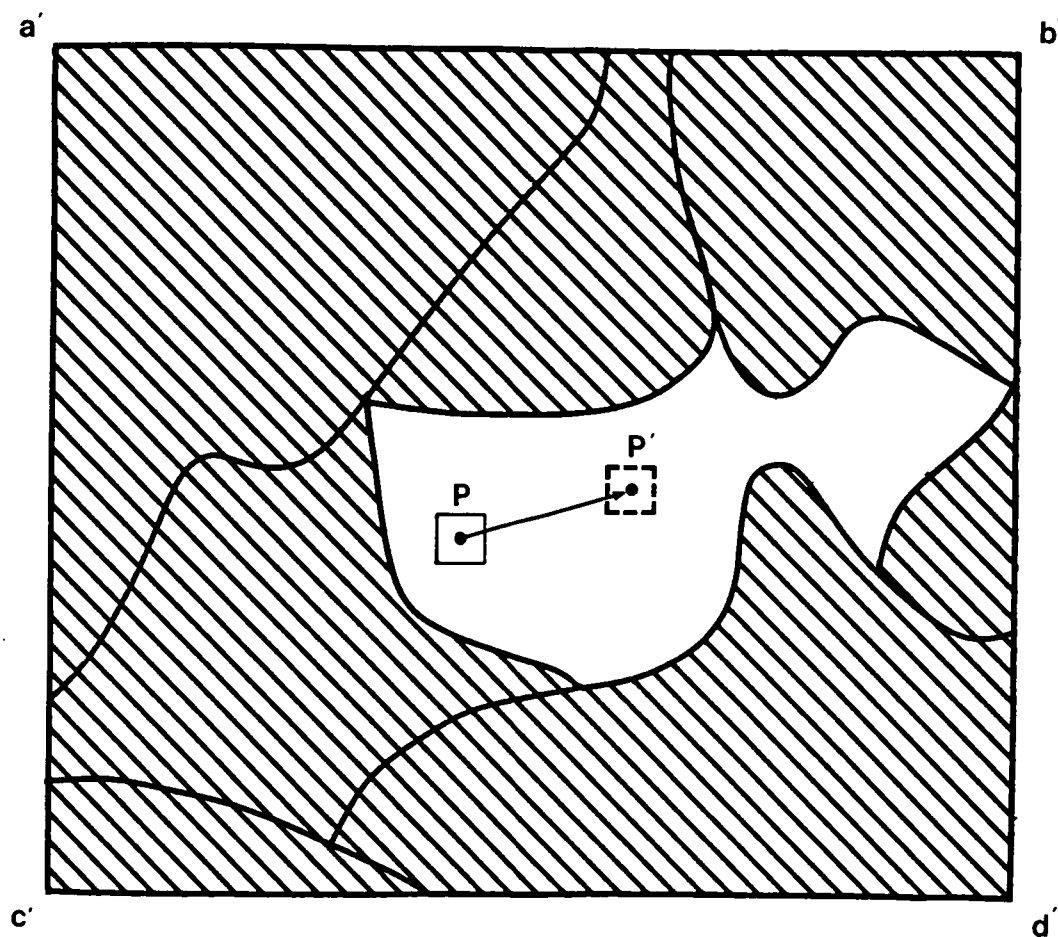


Figure 1.3-9. Tornado location: TREC Vector generation for template cell.

circulations in the storm. This is accomplished by the 3 tasks shown in Fig. 1.3-1(c) (pp. 16). They cover the translation of characteristic patterns which might appear in the air flow map. In fact, depending on the scan level, the definition in the tracked area, radar resolution, and the distribution of the TREC (Tracking Ecoes by Correlation) vectors indicates whether the radar beam is hitting where the tornado is being formed at a higher or a lower level.

The applications of the methods described all showed great potential when applied to several data sets supplied by NOAA. These results are discussed in detail in Chapter 4. The process of curve definition, coding, and matching is discussed in the next section. Detailed descriptions of the other aspects of Fig. 1.3-1 (pp. 14) have been fully detailed by Abidi and Gonzalez [17,30].

CHAPTER 2

SHAPE DESCRIPTION AND MATCHING

2.1. Fourier Descriptors of a Curve

For a curve defined in the complex plane the Fourier Descriptors can be defined in 3 major ways relative to the parametric representation of the curve in question. With reference to Fig. 2.1-1, a point M which belongs to a curve Γ can be located in space by rectangular coordinates x and y or by polar coordinates r and θ . In a complex representation, these parameters are related as follows:

$$Z(t) = x(t) + j y(t) \quad (2.1-1)$$

$$Z(t) = r(t) \exp \{j\theta(t)\} \quad (2.1-2)$$

$$r(t) = [x^2(t) + y^2(t)]^{\frac{1}{2}} \quad (2.1-3)$$

$$\theta(t) = \tan^{-1}[y(t)/x(t)]. \quad (2.1-4)$$

A third representation, which is the one most used for curves, is possible through the parametrization of a curve by the angle formed by that curve and a fixed or relative position versus the length of the curve [Fig.2.1-2].

We will start by generating the Fourier Descriptors using the angle versus length presentation. We assume that Γ is a clockwise-orientated closed curve with parametric presentation

$$Z(\ell) = (x(\ell), y(\ell)) \quad (2.1-5)$$

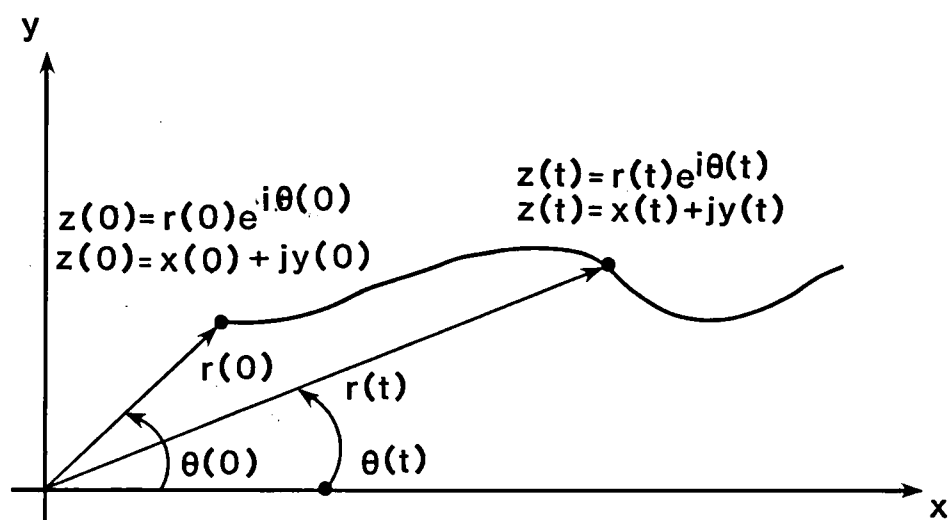


Figure 2.1-1. Rectangular and polar representation of planar curves.

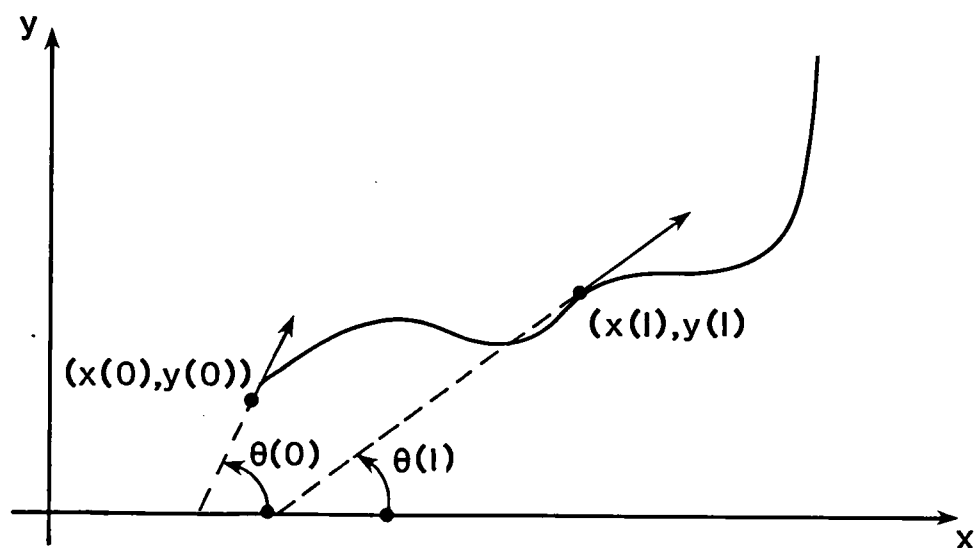


Figure 2.1-2. Parametric representation angle versus arc length of planar curves.

where ℓ is the arc length and

$$0 \leq \ell \leq L \quad (2.1-6)$$

denotes the angular direction of Γ at point ℓ by $\theta(\ell)$. The cumulative angular function $\phi(\ell)$ then will denote the net amount of angular bend between the starting point $\ell = 0$ and the point ℓ . So

$$\Phi(\ell) = \theta(\ell) - \theta(0) \quad \text{mod } 2\pi \quad (2.1-7)$$

with the boundary conditions $\Phi(0) = \delta_0$ and $\Phi(L) = -2\pi + \delta_0$. The magnitude of $\Phi(\ell)$ achieves values higher than 2π when the curve winds in a spiral. Figure 2.1-2 shows the measure of $\Phi(\ell)$ as a function of ℓ . Let us normalize the ϕ function by defining

$$\phi^*(x) = \phi\left(\frac{Lx}{2\pi}\right) + x \quad (2.1-8)$$

where

$$0 \leq \frac{Lx}{2\pi} \leq L \quad (2.1-9)$$

or

$$0 \leq x \leq 2\pi$$

As defined, ϕ^* is a periodic function over the x space with 2π period if the scanning of the curve is performed continuously in a clockwise sense. The function ϕ^* is periodic; its Fourier series expansion is given by

$$\phi^*(x) = m_0 + \sum_{i=1}^{\infty} A_i \cos(ix - a_i) \quad (2.1-10)$$

The set of $\{A_i, \alpha_i; i = 1, \dots, \infty\}$ are the Fourier Descriptors of curve Γ and the general terms represent the i^{th} harmonic "amplitude" and phase "angle." The set $\{A_i\}$ is called the amplitude spectrum and $\{\alpha_i\}$ the phase spectrum. Generally the FD's are given by an equivalent definition

$$\phi^*(x) = \sum_{k=-\infty}^{+\infty} C_k \exp\{jkx\} \quad (2.1-11)$$

where

$$C_k = \frac{1}{2\pi} \int_0^{2\pi} \phi^*(x) \exp\{-jkx\} dx \quad (2.1-12)$$

With the series $\{A_i, \alpha_i\}^\infty$ or $\{C_k\}^\infty$ representing the FD's of a certain curve, three questions arise. How well do the series reconstruct a curve from its FD's if calculated by the latter method? How well do we construct a curve from an arbitrary series $\{A_i, \beta_i\}^\infty$? How easily can we construct a curve from a truncated series $\{A_i, \alpha_i\}^N$ generated by low-pass filtering a series $\{A_i, \alpha_i\}^\infty$ obtained either from Eq. (2.1-8) or arbitrarily created? The first question reduces to one of recovering Γ easily from its angle-versus-length function $\theta(\ell)$. If the starting point of Γ is $Z(0)$, the position of $Z(\ell)$ is given by the expression

$$Z(\ell) = Z(0) + \int_0^\ell e^{i\theta(t)} dt \quad (2.1-13)$$

or equivalently

$$x(\ell) = x(0) + \int_0^\ell \cos \theta(t) \, dt \quad (2.1-14)$$

$$y(\ell) = y(0) + \int_0^\ell \sin \theta(t) \, dt \quad (2.1-15)$$

In fact, by definition $\theta(\ell)$ measures the direction of the vector

$$\sigma'(\ell) = (x'(\ell), y'(\ell))$$

and is tangent to σ at ℓ . The parametrization of a curve by its own arc length makes the speed $|\sigma'(\ell)|$ always equal to 1.

$$|\sigma'(\ell)| = 1, \quad (2.1-16)$$

The only solutions for x' and y' are

$$x'(\ell) = \cos \theta(\ell) \quad (2.1-17)$$

$$y'(\ell) = \sin \theta(\ell) \quad (2.1-18)$$

The fundamental theorem of integral calculus yields

$$x(\ell) - x(0) = \int_0^\ell x'(t) \, dt \quad (2.1-19)$$

$$y(\ell) = y(0) + \int_0^\ell y'(t) \, dt \quad (2.1-20)$$

It is worth noting that because of the periodicity of $x(t)$ and $y(t)$ any $\theta^*(\ell) \equiv \theta(\ell) \pmod{2\pi}$ will lead to the above construction. For completeness, in the remainder of this development $\theta^*(\ell)$ will be used instead of $\theta(\ell)$. Following the previous assumption Eqs. (2.1-7) and (2.1-13) write

$$\theta^*(\ell) = \phi(\ell) + \theta(0) \quad (2.1-21)$$

$$\begin{aligned} Z(\ell) = Z(0) + \frac{L}{2\pi} \int_0^{\frac{2\pi\ell}{L}} \exp\{i[-x + \theta(0) + C_0 \\ + \sum_{k=1}^N A_k \cos(kx - \alpha_k)]\} dx \end{aligned} \quad (2.1-22)$$

Obviously, if we desire a curve Γ' with similar shape but starting at a point Z' , having an initial direction δ'_0 , and of arc length L' , the exact expression is used where the parameters Z , δ_0 , and L now are substituted by Z' , δ'_0 , and L' [39,40].

2.2. Discrete Fourier Descriptors from a Polygonal Curve

In this section we shall derive the Fourier coefficients in rectangular $\{a_k, b_k\}$ or in polar form $\{A_k, \alpha_k\}$. We assume that a polygonal curve has m vertices labeled $P_0, P_1, \dots, P_{m-1}$. The edge (P_{i-1}, P_i) has length ℓ_i defined by

$$\ell_i = D(P_{i-1}, P_i) \quad i = 1, 2, \dots, m \quad (2.2-1)$$

where D is the Euclidean distance and $P_m = P_0$. We also assume that the change in angular direction at Vertex P_i is $\Delta\ell_i$. The total curve length L is then

$$L = \sum_{i=1}^m \Delta\ell_i \quad (2.2-2)$$

and

$$\phi(\ell) = \sum_{i=1}^k \Delta\phi_i \quad \text{for} \quad \sum_{i=1}^k \Delta\ell_i \leq \ell < \sum_{i=1}^{k+1} \Delta\ell_i$$

if we assume that $\phi(\ell) = 0$ for $0 \leq \ell \leq \Delta\ell_1$. Figure 2.2-1 shows the description of a simple closed polygonal curve in terms of edge lengths and vertex bends. Expanding ϕ^* , which is defined by

$$\Phi^*(x) = \phi\left(\frac{Lx}{2\pi}\right) + x \quad (2.2-3)$$

we obtain

$$\Phi^*(x) = m_0 + \sum_{k=1}^{\infty} (a_k \cos kx + b_k \sin kx) \quad (2.2-4)$$

where

$$m_0 = \frac{1}{2\pi} \int_0^{2\pi} \Phi^*(x) dx \quad (2.2-5)$$

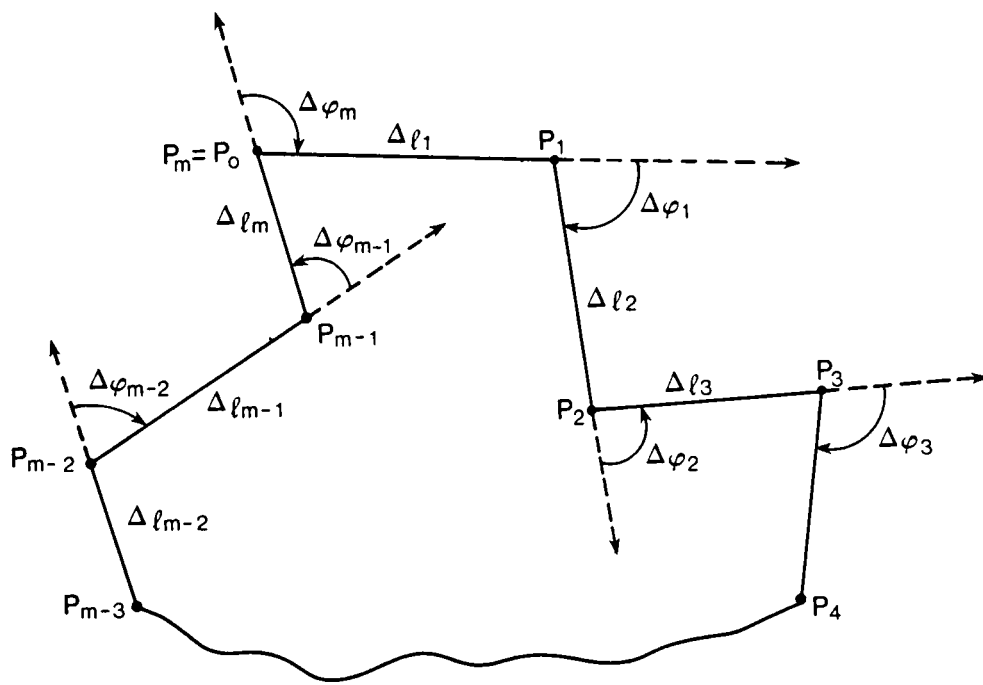


Figure 2.2-1. Approximation of curves by polygon series.

$$a_k = \frac{1}{\pi} \int_0^{2\pi} \phi^*(x) \cos kx \, dx \quad (2.2-6)$$

$$b_k = \frac{1}{\pi} \int_0^{2\pi} \phi^*(x) \sin kx \, dx \quad (2.2-7)$$

Using Eq. (2.2-3) and substituting variable $u = Lx/2\pi$, m_0 , a_k , and b_k may be expressed as

$$m_0 = \frac{1}{L} \int_0^L \phi(u) \, du + \pi \quad (2.2-8)$$

$$a_k = \frac{2}{L} \int_0^L \left(\phi(u) + \frac{2\pi u}{L} \right) \cos \frac{2\pi k u}{L} \, du \quad (2.2-9)$$

$$b_k = \frac{2}{L} \int_0^L \left(\phi(u) + \frac{2\pi u}{L} \right) \sin \frac{2\pi k u}{L} \, du \quad (2.2-10)$$

In this case $\phi(x)$ is a sum of step functions; this reduces the expressions to

$$m_0 = -\pi \frac{1}{L} \sum_{i=1}^m \ell_i \Delta\phi_i$$

$$a_k = \frac{1}{k\pi} \sum_{i=1}^m \Delta\phi_i \sin \frac{2\pi k \ell_i}{L} \quad (2.2-11)$$

$$b_k = \frac{1}{k\pi} \sum_{i=1}^m \Delta\phi_i \cos \frac{2\pi k \ell_i}{L} \quad (2.2-12)$$

where

$$\ell_i = \sum_{k=1}^i \Delta \ell_k \quad (2.2-13)$$

The final forms of a_k , b_k , are appealing because of their similarity and also because the parameters involved, namely the angular bend and the arc length, are very natural in representing polygons [24,28,29]. Before using this contour description in a shape matching algorithm, let's investigate the dependency of this description on starting conditions and the relative position of the curve described with respect to the coordinate system.

The Fourier Descriptors of a specific contour are not unique; as one can see, they are dependent upon the starting point of the contour tracking. This means that there is a set of Fourier coefficients for each delay τ with respect to an arbitrary position. To carry out the following steps, let us consider a contracted form of the Fourier Descriptors as given in Eqs. (2.2-11) and (2.2-12):

$$a_n = \frac{1}{2\pi} \int_0^{2\pi} u(t) \exp[-jnt] dt \quad (2.2-14)$$

$$u(t) = \sum_{n=-\infty}^{\infty} a_n \exp[jnt] \quad (2.2-15)$$

Suppose that an initial series $\{a_n^0\}$ has been calculated for an arbitrary function $u^0(t)$ describing a contour. For a function $u(t) = u^0(t + \tau)$, where τ is an arbitrary delay, the resulting Fourier coefficients become

$$a_n = \frac{1}{2\pi} \int_0^{2\pi} u^0(t + \tau) \exp[-jnt] dt \quad (2.2-16)$$

$$= \exp[jn\tau] \frac{1}{2\pi} \int_0^{2\pi} u^0(t) \exp[-jnt] dt \quad (2.2-17)$$

$$= \exp[jn\tau] a_n^{(0)}$$

It is now of interest to consider the changes in Fourier Descriptors when the curve is shifted, rotated, and dilated in the complex plane.

Translation

If Z_0 is a constant characterizing the translation of the curve

$$u(t) = u^0(t) + Z_0 \quad (2.2-18)$$

then since Z_0 is constant, only a_0 is affected:

$$\begin{aligned} u(t) &= \sum_{n=-\infty}^{\infty} a_n^0 \exp[jnt] + Z_0 \\ &= (a_0^0 + Z_0) + \sum_{n=-\infty}^{\infty} a_n^0 \exp[jnt] \end{aligned} \quad (2.2-19)$$

hence,

$$a_0 = a_0^0 + Z_0 \quad (2.2-20)$$

$$a_n = a_n^0 \quad \text{for } n \neq 0$$

All coefficients except a_0 are invariant to translation. For any description, the coefficient a_0 indicates the position of the curve center of gravity.

Rotation

Suppose that a contour is rotated around its center of gravity by an angle ϕ . The function $u^0(t)$ is multiplied by $\exp[j\phi]$:

$$u(t) = \exp[j\phi] u^0(t) \quad (2.2-21)$$

The resulting Fourier Descriptors are all multiplied by $\exp[j\phi]$:

$$a_n = \exp[j\phi] a_n^0 \quad (2.2-22)$$

Scaling

If the original curve is scaled by a factor R , this results in a fictitious scaling of the spatial coordinates by a factor $1/R$. Using the scaling property of the Fourier Transform, we can show that the Fourier Descriptors are multiplied by R :

$$a_n = R a_n^0 \quad (2.2-23)$$

In general, the Fourier Descriptors are affected by a change of starting point, a rotation and dilation in the following manner:

$$a_n = \exp[jn\tau] R \exp[j\phi] a_n^0 \quad (2.2-24)$$

It is worth noting that, while the effect of scaling is visible in the amplitude spectrum, the effect of the rotation and change of starting point is reflected in the phase spectrum. (The dependence of the phase on the order of the coefficient is very inconvenient if the curve search process does not ensure a controllable start [41].)

From Eq. (2.2-23) it can be shown that these Fourier coefficients in their common form are not particularly useful because of the insignificance of some information they carry with regard to shape.

If we normalize these coefficients in the following manner

$$b_n = \frac{a_{n+1} a_{-n+1}}{a_1^2}, \quad (2.2-25)$$

the effect of starting point, rotation, and dilation are neutralized. In fact

$$\begin{aligned} b_n &= \frac{a_{n+1} a_{-n+1}}{a_1^2} \\ &= \frac{a_{n+1}^0 \exp[j(n+1)\tau] R \exp[j\phi] a_{-n+1}^0 \exp[j(-n+1)\tau] R \exp[j\phi]}{[a_1^0 \exp[j\tau] R \exp[j\phi] R \exp[j\phi]]^2} \\ &= \frac{a_{n+1}^0 a_{-n+1}^0}{[a_1^0]^2} = b_n^0. \end{aligned}$$

For some problems, the orientation of the contours studied is necessary in following matching and identification processes. For this reason, the series $\{b_n\}$ is completed by defining b_1 as

$$\begin{aligned}
 b_1 &= \frac{a_2 |a_1|}{a_1^2} = \frac{a_2^0 \exp[j2\tau] R \exp[j\phi] a_1^0 R}{[a_1^0 \exp[j\tau] R \exp[j\phi]]^2} \\
 &= \frac{a_2^0 |a_1^{(0)}|}{[a_1^0]^2} \exp[-j\phi] \\
 &= b_1^{(0)} \exp[-j\phi]
 \end{aligned} \tag{2.2-26}$$

In more sophisticated cases where the difference between the unknown sample and the nearest sample in the training set is not very visible a "joint moment" is considered instead of the series $\{b_n\}$:

$$d_{mp} = \frac{a_{n+1}^m a_{-n+1}^p}{a_1^{m+p}} \tag{2.2-27}$$

Certain combinations of m and p allow d_{mp} to describe certain symmetrical properties between 2 shapes. In practical problems, normalized Fourier Descriptors are very practical in shape matching, but the position, orientation, and size of compared contours are sometimes necessary for recognition. For this reason, if retrieving position, orientation, and size information from (b_n) is impossible, then the Fourier Descriptors are not normalized, and the parameters

characterizing this information are estimated in the sense of the similarity measure defined for the problem at hand. Detailed solutions of this problem using the Euclidean distance as a similarity measure are presented in the following section.

2.3. Relationship Between Fourier Descriptors and Geometric Shapes

A number of useful relationships link functions of the Fourier Descriptors of a simple closed curve with the geometric properties of the shape which is bounded by that curve. The major justification for this is that the Fourier Descriptors can be made independent of position, rotation, size, and starting point. This property already has been shown in Section 2.2. It has been shown also in earlier work [40,42] that the pure shape information resides in the normalized amplitude spectrum rather than in the phase spectrum. Furthermore, when an algorithm computes the Fourier Descriptors, the information which is related to position, orientation, and starting point is constant with respect to a coordinate system and generally factored out in parameters such as $Z(0)$, $\theta(0)$, and L , which were all introduced in Section 2.1. Following this thinking, Fourier Descriptors are used for matching and identification of polygonal closed curves representing the shapes encountered in the problem at hand. In shape-analysis problems it is assumed that there is a finite set of known equivalent classes of contours called prototypes and one unknown contour to be classified into one of these classes. Sometimes the orientation and size of the unknown contour need to be

recovered also. The matching process can be performed in the Fourier domain using the Minimum-Distance Classifier (MDC).

Let Z_i be the normalized prototype for classes S_i , $i = 1, 2, \dots, P$. These prototypes are generated with some arbitrary, but fixed, starting point and angular orientation. Let $C_i = \{C_{ki}\}^*$ be the set of Fourier coefficients for prototype Z_i . For practical applications the series $\{C_{ki}\}^*$ is low-pass filtered, that is, only a truncated $\{C_{ki}\}^N$ is stored for each prototype Z_i . An unknown contour Z with a series $\{C_i\}^N$ is classified in class S_j if

$$d(Z, Z_j) < d(Z, Z_i) \quad \text{for } i = 1, 2, \dots, P, i \neq j \quad (2.3-1)$$

where d is a distance measure defined as

$$d(\alpha, \beta) = \left[\sum_{n=-N}^N \left| a_n - b_n \right|^2 \right]^{1/2} \quad (2.3-2)$$

where $\{a_n\}^N$ and $\{b_n\}^N$ represent truncated series Fourier Descriptors of curves α and β . To eliminate the effect in position, a_0 and b_0 are set equal, then the summation is

$$d^*(\alpha, \beta) = \left[\sum_{\substack{n=-N \\ n \neq 0}}^N \left| a_n - b_n \right|^2 \right]^{1/2} \quad (2.3-3)$$

In practice, curve α is not compared with β but with a rotated (ϕ), a scaled (R), and an adjusted starting point (τ); that is

$$d^*(\alpha, \beta) = \text{Min} \left[\sum_{n=-N}^{+N} \left| a_n - R \exp(j\phi) \exp(jn\tau) b_n \right|^2 \right]^{1/2} \quad (2.3-4)$$

where $d(\alpha, \beta)$ is minimized with respect to R , ϕ , and τ . The two contours will match perfectly if they differ only by a scaling factor, a rotation about their centroids, or by the starting point. The two curves are then said to be fitted in the Least-Mean-Square sense. Equation (2.3-4) can be written

$$d^2(\alpha, \beta) = \sum \left| a_n - R e^{j(n\tau + \phi)} b_n \right|^2 \quad (2.3-5)$$

The summation and maximization are assumed for the rest of the section. Expression (2.3-5) equals

$$\begin{aligned} & \sum \left[a_n - R e^{j(n\tau + \phi)} b_n \right] \left[a_n^* - R e^{-j(n\tau + \phi)} b_n^* \right] \\ &= \sum a_n a_n^* + R^2 \sum b_n b_n^* - 2R \sum \text{Real} \left[a_n^* b_n e^{j(n\tau + \phi)} \right] \end{aligned}$$

If we set

$$a_n^* b_n = \xi_n e^{j\psi_n}$$

the summation becomes

$$\sum a_n a_n^* + R^2 \sum b_n b_n^* - 2 R \sum \xi_n \cos(\Psi_n + n\tau + \phi) \equiv X$$

To maximize X with respect to R , ϕ , and τ we take its partial derivatives and set them equal to zero.

$$\frac{\partial X}{\partial R} = 2R \sum b_n b_n^* - 2 \sum \xi_n \cos(\Psi_n + n\tau + \phi) \quad (2.3-6)$$

$$\frac{\partial X}{\partial \phi} = 2R \sum \xi_n \sin(\Psi_n + n\tau + \phi) \quad (2.3-7)$$

$$\frac{\partial X}{\partial \tau} = 2R \sum \xi_n n \sin(\Psi_n + n\tau + \phi) \quad (2.3-8)$$

By setting Eq. (2.3-6) equal to zero

$$R = \frac{\sum \xi_n \cos(\Psi_n + n\tau + \phi)}{\sum b_n b_n^*} \quad (2.3-9)$$

where R is given as a function of τ , ϕ , and known quantities which can be computed from the numerical series $\{a_n\}^N$ and $\{b_n\}^N$. Equations (2.3-7) can be modified as follows:

$$\sum \xi_n \sin(\Psi_n + n\tau + \phi) = 0$$

$$\sum \xi_n [\sin(\Psi_n + n\tau) \cos \phi + \cos(\Psi_n + n\tau) \sin \phi] = 0$$

$$\cos \phi \sum \xi_n \sin(\Psi_n + n\tau) = -\sin \phi \sum \xi_n \cos(\Psi_n + n\tau)$$

$$\phi = -\tan^{-1} \left[\frac{\sum \xi_n \sin(\Psi_n + n\tau)}{\sum \xi_n \cos(\Psi_n + n\tau)} \right] \quad (2.3-10)$$

Similarly Eq. (2.3-8) can be written as

$$\phi = -\tan^{-1} \left[\frac{\sum \xi_n n \sin(\Psi_n + n\tau)}{\sum \xi_n n \cos(\Psi_n + n\tau)} \right] \quad (2.3-11)$$

where ϕ is given as a function of τ and known quantities. By equating the expressions in Eqs. (2.3-10) and (2.3-11) we get an equation in τ .

$$\begin{aligned} f(\tau) &= \sum \xi_n \sin(\Psi_n + n\tau) \sum n \xi_n \cos(\Psi_n + n\tau) \\ &- \sum \xi_n \cos(\Psi_n + n\tau) \sum n \xi_n \sin(\Psi_n + n\tau) = 0 \end{aligned}$$

It can be shown [17] that this equation has a solution in the interval $[0, 2\pi]$ if the summation is finite. All possible solutions also are recoverable by numerical techniques. For each root of $f(\tau) = 0$ the R and ϕ solutions are given by Eqs. (2.3-9) and (2.3-10). The multiplicity of the solutions is certainly related to symmetry, reflection, and rotation of a contour with respect to principal axes [35,39].

The optimal values R , ϕ , and τ are then entered in Eq. (2.3-4) to evaluate the distance between curves α and β . Using this similarity measure, a curve Z is preclassified using Eq. (2.3-1). For a final

classification, the distance space is partitioned into disjoint subsets which are generally convex and centered around a prototype center. In other words, besides verifying Eq. (2.3-1), the minimum distance between the pattern to be classified and the nearest prototype has to be less than a certain threshold, d_{th} , beyond which the pattern is rejected. It has been shown, then, that in both cases, whether the Fourier coefficients are normalized or not, the distance defined in Eq. (2.3-2) is optimal. In the case where the Fourier coefficients are normalized, the information concerning rotation, starting point, and scaling is lost, but the comparison is straightforward. On the other hand, if the Fourier Descriptors are not normalized, the information concerning rotation, starting point, and scaling is recovered by the minimization process presented earlier in this section. The matching hence is prolonged because this minimization process has to be done for each pattern to be classified, whether or not this information is needed.

2.4. Computational Considerations

Heavy computation generally is involved in using Fourier Descriptors for contour matching and identification. To estimate the efficiency of this method for digital contours (our actual purpose), we shall first introduce the general form of the Fourier coefficients to compare it against the new developments.

For a one-dimensional array $Z_N(z_1, z_2, \dots, z_N)'$, the Fourier coefficients are given as a series $\{c_k\}^N$ or one-dimensional array $C_N = (c_1, c_2, \dots, c_N)'$ from:

$$c_k = \frac{1}{N} \sum_{i=1}^N Z_i \exp\{-j 2\pi ki/N\} \quad (2.4-1)$$

If we define the vector W by

$$W = W_N = (w^1, w^2, w^3, \dots, w^N)' \quad (2.4-2)$$

where

$$w = w_N = \exp\{-j \frac{2\pi}{N}\} \quad (2.4-3)$$

the vectors Z and C are then related by

$$C = WW'Z \quad (2.4-4)$$

Because of the periodicity of the function defining w_N the matrix $W_N' W_N$ has only N distinct values. In general, this matrix is filled only once and will not be considered a major computation task. To compute all the elements of C , N^2 multiplications are required. It can be shown that the number of multiplications needed can be cut down to $2 N \log_2 N$ if N is an integer power of 2 [18]. The algorithm which performs this reduction is called the Fast Fourier Transform (FFT). If N is not an exact power of 2, the efficiency of the algorithm is altered dramatically.

The computation of Fourier coefficients is radically different when used for contour description. To show how it differs from the general case, we shall first present how the parameters ℓ_i and $\Delta\phi$ presented in Eqs. (2.2-11) through (2.2-13) are computed from image data.

In general, coded chains describe closed contours extracted from binary images resulting from an original image scene after preprocessing segmentation. Depending on scene content, regions in this binary image are split or merged to form significant features. After this step the image then is processed through gradient techniques to generate initial contours. For most contour generators the resulting patterns are only a few pixels wide, requiring, then, a thinning procedure for most of the contours in the picture [See Figs. 1.3-5 through 1.3-9 (pp. 20 and 22 through 24)]. Obviously, the outcome depends significantly on the thinning algorithm used. Because of the shape regularity encountered in the experiments to which this theoretical development is intended to be applied, the thinning algorithm used generates a unique path of either 8- or 4-connected contours. Figure 2.4-1 shows all possible connections between a central point P and its two neighbors for the 8- connecting scheme. The total number of possible patterns is 32 for oriented contours and 16 for non-oriented contours. The patterns are much more simple for the 4- connectivity scheme. Figure 2.4-2 shows all possible connections between a central point P and its two neighbors for this connectivity.

These thinning approaches are presented to evaluate $\Delta\ell_i$ and $\Delta\phi_i$ in order to compute the Fourier Descriptors for a contour coded with these

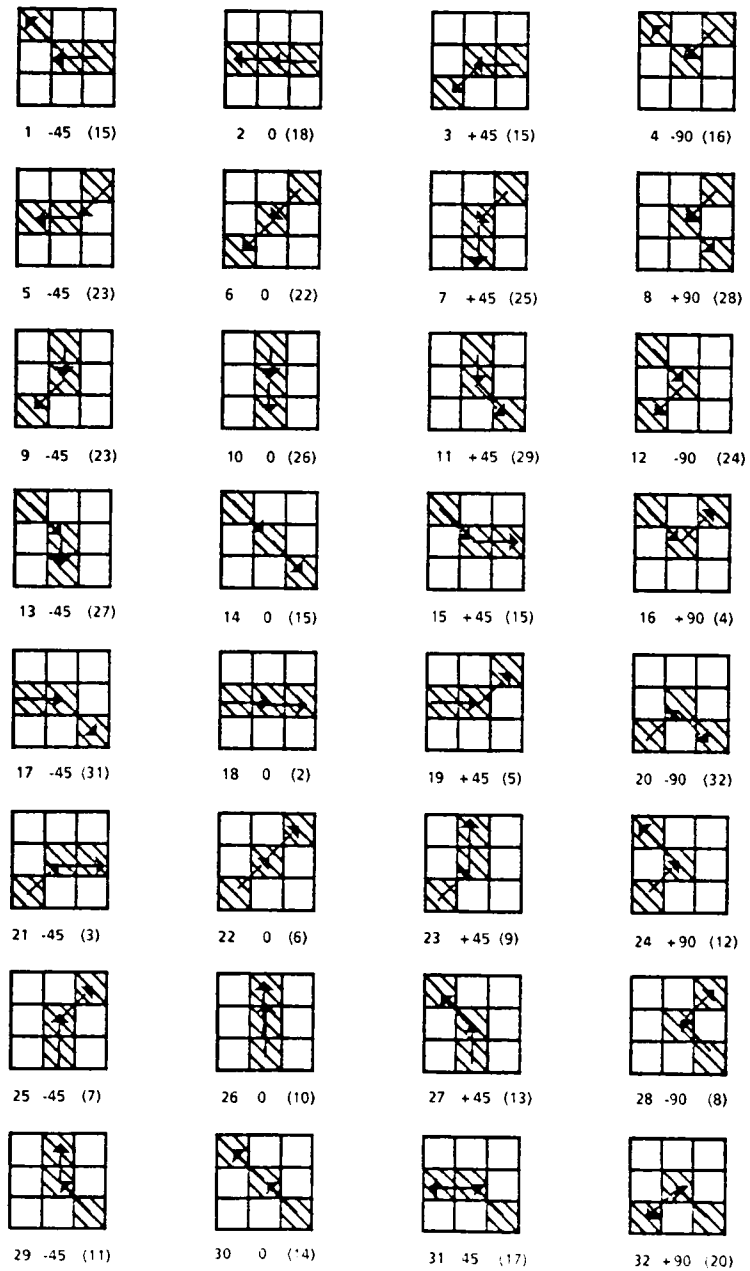


Figure 2.4-1. Configuration of the neighborhood of a multiple pixel for 8-connected contours.

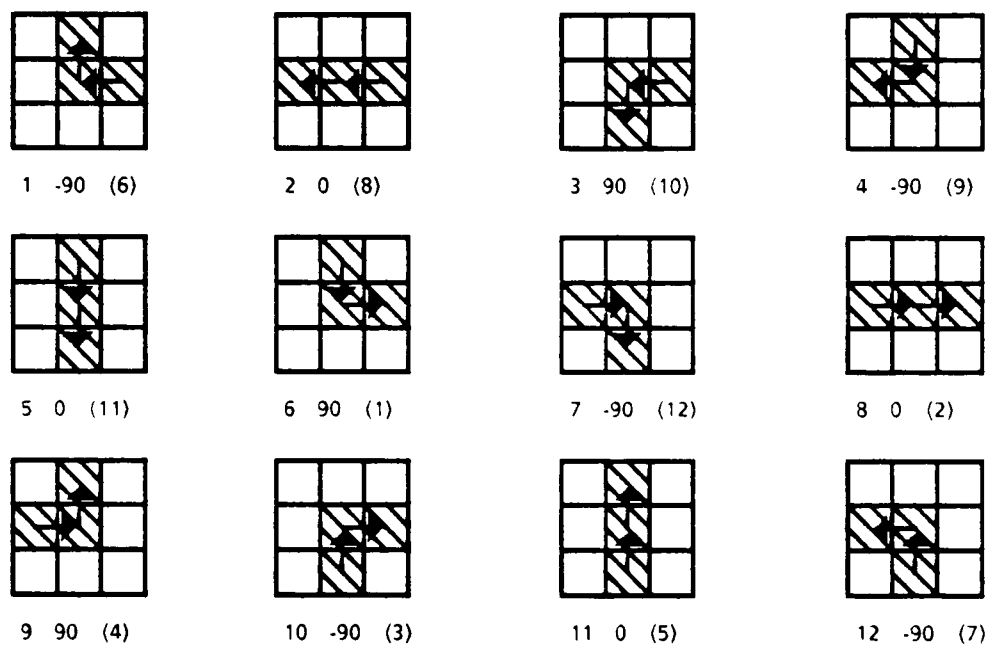


Figure 2.4-2. Configuration of the neighborhood of a multiple pixel for 4-connected contours.

algorithms. Chain codes are derived for 8-connected contours and then for 4-connected contours.

1-Chain Code generation for 8-connected contours.

The chain code first described by Freeman [35] approximates contours by a sequence of linear fits that consists of 8 standard line segments [Fig. 2.4-3(a)]. The code for a contour is then the chain vector V of length k

$$V = a_1 a_2 a_3 \dots a_k$$

where a_i is an integer between 0 and 7 oriented in the direction

$$\phi = \frac{\pi}{4} a_i$$

measured counter-clockwise from the X axis of a rectangular X - Y coordinate system). The length of each line segment is 1 or $\sqrt{2}$ depending on whether the corresponding a_i is even or odd. The complex representation of link a_i can be written in phasor notation as:

$$\left[1 + \left(\frac{\sqrt{2}-1}{2} \right) \left(1 - (-1)^{a_i} \right) \right] \left| \frac{\pi}{4} a_i \right. \quad (2.4-5)$$

An example of a chain code of a circular object is shown in Fig. 2.4-4(a). It is worth noting that the 8-connectivity scheme chosen does not allow replacement of a diagonal vector by 2 laterals (vertical and horizontal, or horizontal and vertical). The detailed chain links and corresponding

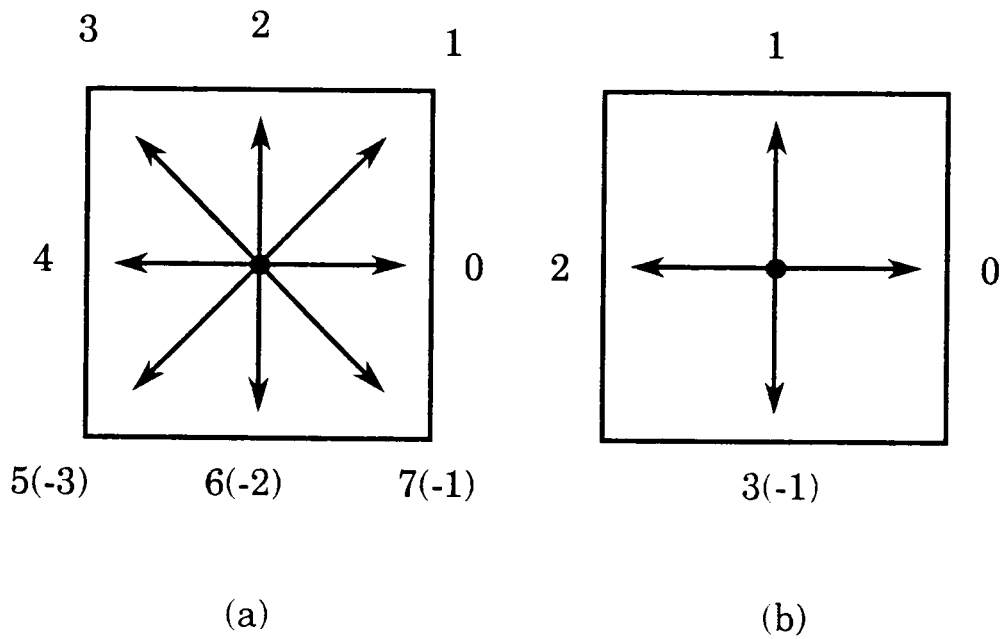


Figure 2.4-3. Chain code link labels for (a) 8-connected chains, (b) 4-connected chains.

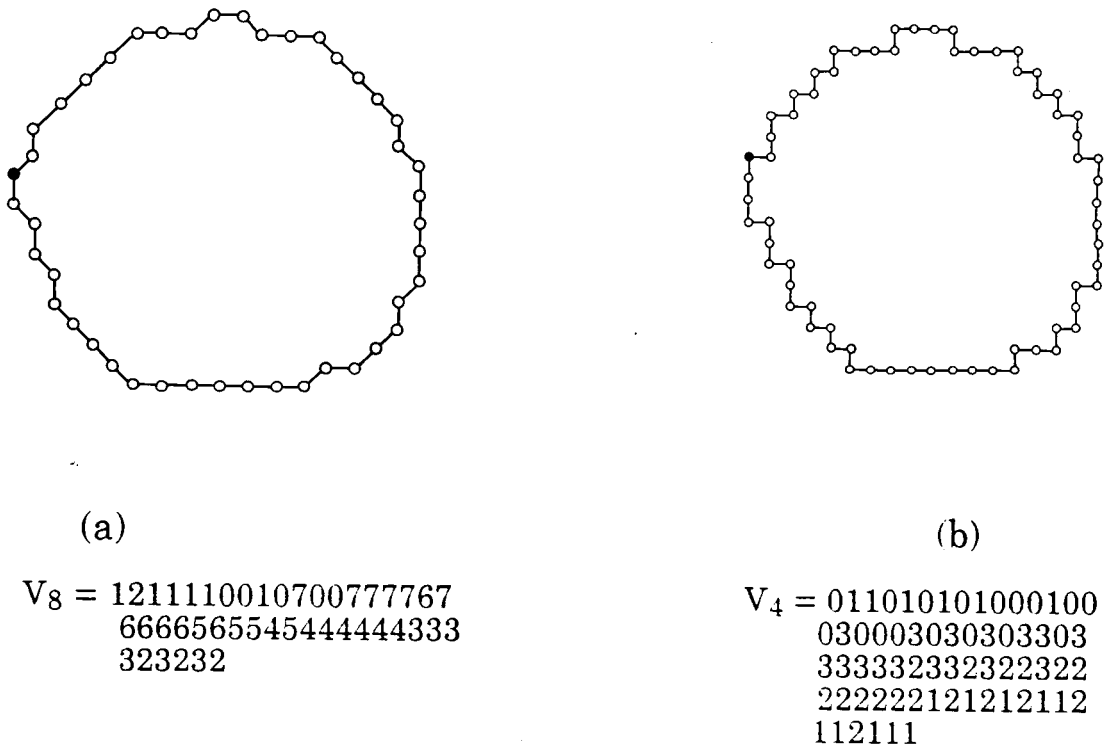


Figure 2.4-4 Example of (a) 8-neighbor (b) 4-neighbor coded chain from the contour of a circular object.

codes generated from the dominant feature of Fig. 1.3-6 (pp. 20) are shown in Fig. 2.4-5.

2-chain code generation for 4-connected contours.

The same contour which can be described in an 8-connectivity scheme can be described in a 4-connectivity scheme. Since only lateral links are authorized, the chain code vector of length k can be expressed as:

$$V = a_1 a_2 a_3 \dots a_k \quad (2.4-6)$$

where, this time, a_i is an integer between 0 and 3 oriented in the direction of

$$\phi = \frac{\pi}{2} a_i$$

(measured counter-clockwise from the X axis of an rectangular X-Y coordinate system). What is special about this presentation is that the length of each link is always unity. The complex representation of link a_i can be written in phasor notation as

$$1 \mid \frac{\pi}{2} a_i \quad (2.4-7)$$

An example of a chain code of the same circular object presented in Fig. 2.4-4(a) is shown in Fig. 2.4-4(b) using 4-connectivity. It is interesting to

note that the number of samples needed to represent the circular shape is significantly higher in the case of the 4-connectivity scheme than in the 8-connectivity case.

Now that the generation of both ℓ_i and $\Delta\phi_i$ is assured, the computation of the Fourier Descriptors is straightforward. We first present the case of 4-connected chains.

Recall that the Fourier Descriptors are given in Eqs. (2.2.-11) through (2.2.-13) by

$$m_0 = -\pi - \frac{1}{L} \sum_{i=1}^m \ell_i \Delta\phi_i$$

$$-a_k = \frac{1}{k\pi} \sum_{i=1}^m \Delta\phi_i \sin \frac{2\pi k \ell_i}{L}$$

$$b_k = \frac{1}{k\pi} \sum_{i=1}^m \Delta\phi_i \cos \frac{2\pi k \ell_i}{L}$$

where

$$\ell_i = \sum_{k=1}^i \Delta\ell_k$$

With reference to Fig. 2.4-2 we see that

$$\Delta\phi_i = \phi_i - \phi_{i-1}$$

where

$$\phi_i = \left(\frac{\pi}{2}\right) a_i \quad (2.4-8)$$

hence $\Delta\phi_i$ can be written as

$$\Delta\phi_i = \frac{\pi}{2} (a_i - a_{i-1}) = \frac{\pi}{2} \psi_i \quad i = 2, 3, \dots, m$$

$$\Delta\phi_1 = \frac{\pi}{2} a_1$$

Recall that a_i is always an integer between 0 and 3 (0, 1, 2, and -1). As shown in Fig. 2.4-2 $\Delta\phi_i$ takes the values of

$$-\frac{\pi}{2}, 0, \frac{\pi}{2}$$

and b_i takes the values of -1, 0, 1. If we adopt the notation $a_k = -2a^*_k$ and $b^*_k = 2b_k$, Eqs. (2.2-12) and (2.2-13) can be written as

$$-a^*_k = \frac{1}{k} \sum_{i=1}^m \psi_i \sin \frac{2\pi k \ell_i}{L}$$

$$b^*_k = \frac{1}{k} \sum_{i=1}^m \psi_i \cos \frac{2\pi k \ell_i}{L}$$

Equivalently,

$$-a_k^* = \left\{ \sum_{\substack{i=1 \\ \Psi_i=1}}^m \sin \frac{2\pi k \ell_i}{L} - \sum_{\substack{i=1 \\ \Psi_i=-1}}^m \sin \frac{2\pi k \ell_i}{L} \right\} \quad (2.4-9)$$

$$b_k^* = \left\{ \sum_{\substack{i=1 \\ \Psi_i=1}}^m \cos \frac{2\pi k \ell_i}{L} - \sum_{\substack{i=1 \\ \Psi_i=-1}}^m \cos \frac{2\pi k \ell_i}{L} \right\} \quad (2.4-10)$$

where a_k^* and b_k^* are now a pure summation. To make use of the notation introduced at the beginning of this section, let us examine the integrand to the sine and cosine in a_k and b_k . We have shown that $\Delta \ell_k$ is always unity, consequently,

$$\ell_i = \sum_{k=1}^i \Delta \ell_k = \sum_{k=1}^i \Delta \ell = i \Delta \ell = i \frac{L}{m}$$

where $\Delta \ell$ is the unit length in which the contour is measured, L is the total length of the countour, and m is the number of samples taken. This simple replacement is far from insignificant. Recall that in the case where the integrand of sine and cosine are not regular, that is, where the quantity $i/(L/m)$ is not an integer, our task of a_k and b_k evaluation is primarily confronted by m^2 , sine, and cosine computations. Obviously this computational inconvenience is beyond the efficiency of the FFT (or similar) algorithm. The expressions giving a_k and b_k are then

$$-a_k^* = \frac{1}{k} \left\{ \sum_{\substack{i=1 \\ \psi_i=1}}^m \sin \frac{2\pi ki}{m} - \sum_{\substack{i=1 \\ \psi_i=-1}}^m \sin \frac{2\pi ki}{m} \right\} \quad (2.4-11)$$

$$b_k^* = \frac{1}{k} \left\{ \sum_{\substack{i=1 \\ \psi_i=1}}^m \cos \frac{2\pi ki}{m} - \sum_{\substack{i=1 \\ \psi_i=-1}}^m \cos \frac{2\pi ki}{m} \right\} \quad (2.4-12)$$

Eqs. (2.4-1) through (2.4-4) may be used to write the expressions giving a_k^* and b_k^* in compact form. If we denote $C = (c_1, c_2, \dots, c_m)'$ and $Z = (z_1, z_2, z_3, \dots, z_m)'$ then Eqs. (2.4-11) and (2.4-12) can be reduced to:

$$\begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_m \end{bmatrix} = \begin{bmatrix} W \\ W^2 \\ W^3 \\ \vdots \\ W^m \end{bmatrix} \begin{bmatrix} W & W^2 & W^3 & \dots & W^m \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_m \end{bmatrix}$$

or

$$C = W W' Z = \underline{W} Z$$

Because z_i is 1 or -1 this matrix multiplication will reduce to pure sums.

Let us rewrite $\underline{W}$ as follows:

$$W = W_{-1} + W_0 + W_1 \quad (2.4-13)$$

where

$$(W_{-1})_{ij} = (W)_{ij} \quad \text{if } \Delta\phi_i = -1 \quad (2.4-14)$$

$$(W_0)_{ij} = (W)_{ij} \quad \text{if } \Delta\phi_i = 0 \quad (2.4-15)$$

$$(W_{+1})_{ij} = (W)_{ij} \quad \text{if } \Delta\phi_i = 1 \quad (2.4-16)$$

Note that this matrix filling is not time-consuming. Furthermore this procedure is done column-by-column and takes a maximum of $2m$ additions. For randomly generated contours the series $\{\psi_i\}$ is uniformly distributed among $-1, 0$, and 1 , hence reducing the maximum bound given on the number of additions needed to compute the Fourier Descriptors. The most common contours in image identification and matching are composed of fairly long line segments favoring the case of $\Delta\phi_i \equiv 0$ and reducing further the number of additions. This can be attributed to some properties of the shape originally sampled or to a prefiltering the chain code undergoes for noise reduction purposes. The matrices W_{-1} , W_0 , and W_{+1} are a partition of W , hence Eq. (2.4-18) can be expressed as

$$C = (W_{-1} + W_0 + W_{+1})I = W^*I \quad (2.4-17)$$

where I is an m -dimensional vector filled in with 1's: $I = (1, 1, \dots, 1)'$. Equation (2.4-17) shows that a term $c_k = 1/k(b_k^* - ja_k^*)$ is merely the sum of the terms in the k th row of the matrix W^* . Note that so far the evaluation of the c_k 's did not require complex multiplications, but to compute a_k and b_k from c_k one complex multiplication is needed.

Similar development can be done for contours coded with 8-neighbor chains. Recall that in this case ϕ and $\Delta\ell_i$ are given by

$$\phi_i = \frac{\pi}{4} a_i \quad (2.4-18)$$

$$\Delta\ell_i = 1 + \left(\frac{\sqrt{2} - 1}{2} \right) \left(1 - (-1)^{a_i} \right) \quad (2.4-19)$$

where a_i is an integer between -3 and 4 (mod 8). To compute a_k and b_k from Eqs. (2.2-11) and (2.2-12), $\Delta\phi_i$ and ℓ_i should be written in a suitable form to ease the computation and maximize the efficiency of the overall matching procedure. The $\Delta\phi_i$ (see section 2.2) can be computed from Eq. (2.4-19) by

$$\Delta\phi_i = \phi_i - \phi_{i-1} = \frac{\pi}{4} (a_i - a_{i-1}) \quad (2.4-20)$$

With reference to the patterns used in 8-neighbor coded chains [Fig. 2.4-1] the values of $\Delta\phi_i$ are computed and marked under each pattern. The bend takes only 5 distinct values, specifically

$$-\frac{\pi}{2}, -\frac{\pi}{4}, 0, \frac{\pi}{4}, \text{ and } \frac{\pi}{2}$$

The values of

$$-\frac{\pi}{a}, 0, \text{ and } \frac{\pi}{2}$$

are equally probable, and so are the bend values of

$$-\frac{\pi}{2}, \text{ and } \frac{\pi}{2}$$

The computation of the arc-length is more complicated and influences significantly the global computation scheme. The arc length ℓ_i is given by

$$\ell_i = \sum_{p=1}^i \Delta \ell_p \quad (2.4-21)$$

$$\begin{aligned} \ell_i &= \sum_{p=1}^i 1 + \left(\frac{\sqrt{2}-1}{2} \right) \left(1 - (-1)^{a_p} \right) \\ &= i + \frac{\sqrt{2}-1}{2} \left\{ \sum_{p=1}^i \left(1 - (-1)^{a_p} \right) \right\} \\ &= i + \frac{\sqrt{2}-1}{2} i - \frac{\sqrt{2}-1}{2} \cdot \sum_{p=1}^i (-1)^{a_p} \\ &= \left(\frac{\sqrt{2}-1}{2} \right) i - \left(\frac{1-\sqrt{2}}{2} \right) \cdot \sum_{p=1}^i (-1)^{a_p} \end{aligned} \quad (2.4-22)$$

If we adopt the following notation

$$f(i) = \sum_{p=1}^i (-1)^{a_p} \quad (2.4-23)$$

$$\alpha = \frac{1 + \sqrt{2}}{2} \quad (2.4-24)$$

$$\beta = \frac{1 - \sqrt{2}}{2} \quad (2.4-25)$$

$$c_k = \pi \{a_k^* - j b_k^*\} \quad (2.4-26)$$

the complex form of the Fourier Descriptors is then given by

$$c_k = \frac{1}{m} \sum_{i=1}^m \Delta\phi_i e^{-j \frac{2\pi k \Delta\ell}{L} \{ \alpha i + \beta f(i) \}} \quad (2.4-27)$$

or

$$c_k = \frac{1}{m} \sum_{i=1}^m \Delta\phi_i \exp \left\{ -j \frac{2\pi k i}{\frac{L}{\alpha \Delta\ell}} \right\} \cdot \exp \left\{ -j \frac{2\pi k f(i)}{\frac{L}{\beta \Delta\ell}} \right\} \quad (2.4-28)$$

The latter expression is symmetric because $f(i)$ takes integer values only. The function $f(i)$ is merely a continuously updated count of the even- and odd- valued links

$$n_e^{(i)} - n_o^{(i)}$$

Respectively.

$$f(i) = n_e^{(i)} - n_o^{(i)} \quad (2.4-29)$$

Consequently the total length of the contours is

$$L = \Delta\ell(n_e + n_o \sqrt{2}) \quad (2.4-30)$$

where n_e and n_o are the total number of even- and odd-valued links in the contour, respectively. The argument of the exponential terms is very similar and "describes" the contours of lengths L/α and L/β . Because of the exponential terms in Eq. (2.4-28), the computation of c_k will require at least m^2 multiplications if $\Delta\phi_i$ is a staircase function with respect to i . In the Discrete Fourier Transform (DFT), exponential terms are computed easily or stored because of their regularity [Eqs. (2.4-2) and (2.4-3)]. Conversely, in this case the exponential terms are not regular because they depend on the distribution of the even- and odd-valued links. In the conventional DFT and FFT algorithms it has been assumed that the speed at which a contour is traversed is constant, resulting in a regular sampling. In the case of contours represented by 8-connected chains, on the other hand, the speed is variable, hence giving an irregular sampling. In fact the speed is constant throughout a link, but depends on whether the code link is even or odd.

For this reason the use of 8-connected contours in spectral analysis is not efficient because of irregularity in the sampling rate. It is true that an 8-connected chain is in general about 2/3 the length of the 4-connected chain representing the same contour, but this reduction in length, even though significant for storage and the number of multiplications required, does not compensate for the inconvenience

introduced by a non-uniform sampling which prohibits the use of the FFT algorithm or even the conventional Discrete Fourier Transform algorithm.

2.5. Elliptic Description of Contours

Assuming that a chain code is traversed at a constant speed s_1 , the time needed to follow a particular link a_i is

$$\Delta t_i = 1 + \left(\frac{\sqrt{2}-1}{2} \right) \left(1 - (-1)^{a_i} \right). \quad (2.5-1)$$

The total time required to run that chain, starting at an arbitrary point, is given by

$$t_p = \sum_{i=1}^P \Delta t_i \quad (2.5-2)$$

The time required to run the chain a full turn will be denoted T . If the total number of links is k , $T = T_k$; the travel distances in the x and y directions along one link for the 4-connectivity scheme are given by

$$\Delta x_i = \text{sign}(1 - a_i) \text{sign}(3 - a_i) \quad (2.5-3)$$

and

$$\Delta y_i = \text{sign}(2 - a_i) \text{sign}(a_i) \quad (2.5-4)$$

where

$$\text{sign}(u) = \begin{cases} 1 & u > 0 \\ 0 & u = 0 \\ -1 & u < 0 \end{cases}$$

Similary, the travel distances in the x and y directions are given for the 8-connectivity scheme by

$$\Delta x_i = \text{sign}(6 - a_i) \text{sign}(2 - a_i) \quad (2.5-5)$$

$$\Delta y_i = \text{sign}(4 - a_i) \text{sign}(a_i) \quad (2.5-6)$$

For both schemes the total distances traversed along several links are

$$x_p = \sum_{i=1}^p \Delta x_i \quad (2.5-7)$$

and

$$y_p = \sum_{i=1}^p \Delta y_i \quad (2.5-8)$$

As was mentioned in previous sections, the function describing x with t as a parameter is periodic and hence can be expected as follows

$$x(t) = X_0 + \sum_{n=0}^{\infty} \alpha_n \cos \frac{2\pi nt}{T} + \beta_n \sin \frac{2\pi nt}{T} \quad (2.5-9)$$

The function $x(t)$ is periodic and so is its derivative; consequently $x'(t)$ is expandable:

$$x'(t) = \sum_{n=0}^{\infty} a_n \cos \frac{2\pi nt}{T} + \beta_n \sin \frac{2\pi nt}{T}, \quad (2.5-10)$$

where

$$a_n = \frac{2}{T} \int_0^T x'(t) \cos \frac{2\pi nt}{T} dt \quad (2.5-11)$$

$$\beta_n = \frac{2}{T} \int_0^T x'(t) \sin \frac{2\pi nt}{T} dt \quad (2.5-12)$$

The coefficient a_n should be expressed as a function of the discrete values of t and x :

$$\begin{aligned} a_n &= \frac{2}{T} \int_0^T x'(t) \cos \frac{2\pi nt}{T} dt \\ &= \frac{2}{T} \sum_{i=1}^K \int_{t_{i-1}}^{t_i} \frac{\Delta x_i}{\Delta t_i} \cos \frac{2\pi nt}{T} dt \\ &= \frac{2}{T} \sum_{i=1}^K \frac{\Delta x_i}{\Delta t_i} \int_{t_{i-1}}^{t_i} \cos \frac{2\pi nt}{T} dt \\ &= \frac{1}{n\pi} \sum_{i=1}^K \frac{\Delta x_i}{\Delta t_i} \left(\sin \frac{2\pi nt_i}{T} - \sin \frac{2\pi nt_{i-1}}{T} \right) \end{aligned} \quad (2.5-13)$$

Similarly β_n can be expressed as

$$\beta_n = \sum_{i=1}^K \frac{\Delta x_i}{\Delta t_i} \left(\cos \frac{2\pi n t_{i-1}}{T} - \cos \frac{2\pi n t_i}{T} \right) \quad (2.5-14)$$

The derivative $x'(t)$ is computed directly from

$$x(t) = X_0 + \sum_{n=1}^{\infty} \alpha_n \cos \frac{2\pi n t}{T} + \beta_n \sin \frac{2\pi n t}{T}$$

$$x'(t) = \sum_{n=1}^{\infty} \left\{ \frac{2\pi n}{T} \alpha_n \right\} \cos \frac{2\pi n t}{T} + \left\{ \frac{2\pi n}{T} \beta_n \right\} \sin \frac{2\pi n t}{T} \quad (2.5-15)$$

Equating terms in Eqs. (3.1-10) and (3.1-15) yields

$$\frac{2\pi n}{T} \alpha_n = \frac{1}{\pi n} \sum_{i=1}^K \frac{\Delta x_i}{\Delta t_i} \left(\sin \frac{2\pi n t_i}{T} - \sin \frac{2\pi n t_{i-1}}{T} \right)$$

and

$$\alpha_n = \frac{T}{2\pi^2 n^2} \sum_{i=1}^K \frac{\Delta x_i}{\Delta t_i} \left(\sin \frac{2\pi n t_i}{T} - \sin \frac{2\pi n t_{i-1}}{T} \right) \quad (2.5-16)$$

Using the same development,

$$\beta_n = \frac{T}{2\pi^2 n^2} \sum_{i=1}^K \frac{\Delta x_i}{\Delta t_i} \left(\cos \frac{2\pi n t_i}{T} - \cos \frac{2\pi n t_{i-1}}{T} \right) \quad (2.5-17)$$

If $y(t)$ is expanded in the same manner, the following relations hold true:

$$y(t) = Y_0 + \sum_{n=1}^{\infty} \gamma_n \cos \frac{2\pi n t}{T} + \delta_n \sin \frac{2\pi n t}{T}$$

$$\gamma_n = \frac{T}{2\pi^2 n^2} \sum_{i=1}^K \frac{\Delta x_i}{\Delta t_i} \left(\sin \frac{2\pi n t_i}{T} - \sin \frac{2\pi n t_{i-1}}{T} \right) \quad (2.5-18)$$

$$\delta_n = \frac{T}{2\pi^2 n^2} \sum_{i=1}^K \frac{\Delta x_i}{\Delta t_i} \left(\cos \frac{2\pi n t_i}{T} - \cos \frac{2\pi n t_{i-1}}{T} \right) \quad (2.5-19)$$

The expressions of α_n , β_n , γ_n , and δ_n , given by Eqs (3.1-16) through (3.1-19) are similar to those given by Eqs. (2.2-11) and (2.2-12). As was mentioned in Section 2.4, the computation of the Fourier Descriptors cannot take advantage of the FFT algorithm for the 8-connected contours, whereas this task is reduced practically to mere summations for the 4-connected scheme. The same remarks hold true for the Elliptic Fourier Descriptors developed in this section. Furthermore, in later sections we will show that for matching purposes only 8 to 12 harmonics are required for satisfactory description of the contour in question. For such a small number of coefficients, the computation task is reduced dramatically and could be considered to be practically nonexistent when compared to the preprocessing steps of filtering, contour extraction, thinning, and curve-coding. Other advantages of the Elliptic features over the Fourier Descriptors introduced in the latter section are the ease of curve reconstruction and the ready adaptability to visual and graphical interpretation.

2.6. Approximation of Curves by a Finite Number of Harmonics

Shapes, contours, and chain-codes are widely used in target matching and identification. In practice only truncated series of the infinite sums developed in the last section are used to describe the original contours. Truncation is an ideal low-pass filtering; hence a cutoff frequency needs to be set. Selecting a cutoff frequency necessitates the smallest usable spacing between the grid lines of the classification space. This completely defines the resolution within which the reconstruction procedure is performed [41,42].

Factors such as global target size, the size of meaningful features, noise, and the measure of similarity influence significantly the grid spacing chosen and the number of harmonics required to describe properly the contour at hand. We now relate the spacing grid (ϵ) to the number of harmonics used in reconstructing a curve. For this purpose we define

$$X_N = x_N(t) = \alpha_n \cos \frac{2\pi nt}{T} + \beta_n \sin \frac{2\pi nt}{T} \quad (2.6-1)$$

$$X_N = y_N(t) = \gamma_n \cos \frac{2\pi nt}{T} + \delta_n \sin \frac{2\pi nt}{T} \quad (2.6-2)$$

From this definition

$$x(t) = X_0 + \sum_{n=1}^{\infty} X_N(t) \quad (2.6-3)$$

$$y(t) = Y_0 + \sum_{n=1}^{\infty} Y_N(t) \quad (2.6-4)$$

The series pairs (X_N, Y_N) and (X_∞, Y_∞) each define a curve; they will be denoted by C_N and C_∞ , respectively. One possible measure of closeness between C_N and C_∞ is the maximum of the Supremums of the difference in the x and y directions. The distance between the 2 curves or their chain-codes is

$$D(C_\infty, C_N) = D(V_\infty, V_N) = \max \left[S_{t=0}^T |x_\infty(t) - x_N(t)|, S_{t=0}^T |y_\infty(t) - y_N(t)| \right] \quad (2.6-5)$$

where $S_{t=0}^T(Z)$ denotes the supremum of Z over the time range between $t=0$ and $t=T$. To evaluate the distance $D(C_\infty, C_N)$ we shall use $x_\infty(t)$ and $x_N(t)$ in complex form:

$$x_\infty(t) = \sum_{n=-\infty}^{+\infty} \lambda_n \exp \left\{ j \frac{2\pi n t}{T} \right\} \quad (2.6-6)$$

with

$$\lambda_n = \frac{1}{T} \int_0^T x_\infty(t) \exp \left\{ -j \frac{2\pi n t}{T} \right\} \quad (2.6-7)$$

The truncated series is then

$$x_N(t) = \sum_{k=-N}^N \lambda_n \exp \left\{ j \frac{2\pi n t}{T} \right\} \quad (2.6-8)$$

The total error between $x_\infty(t)$ and $x_N(t)$ is

$$S_{t=0}^T |x_\infty(t) - x_N(t)| = \sum_{n=-\infty}^{-N-1} |\lambda_n| + \sum_{n=N+1}^{\infty} |\lambda_n| \quad (2.6-9)$$

Using Eq. (3.2-7) and integrating by parts we obtain

$$\begin{aligned} T \lambda_n &= \int_0^T x_\infty(t) \exp \left\{ -j \frac{2\pi nt}{T} \right\} dt \\ &= \left[x_\infty(t) \frac{T}{-j2\pi} \exp \left\{ -j \frac{2\pi nt}{T} \right\} \right]_0^T - \int_0^T x'(t) \frac{T}{-j2\pi n} \exp \left\{ -j \frac{2\pi nt}{T} \right\} dt \end{aligned}$$

Since $x(t)$ and

$$\exp \left\{ -j \frac{2\pi nt}{T} \right\}$$

are periodic with period T ,

$$\lambda_n = \frac{1}{j2\pi n} \int_0^T x'(t) \exp \left\{ -j \frac{2\pi nt}{T} \right\} dt \quad (2.6-10)$$

The derivative $x'(t)$ is piece-wise defined since $x(t)$, and is piece-wise differentiable; when $x(t)$ does not exist it is defined by

$$x'(t) = \frac{x'(t^+) + x'(t^-)}{2} \quad (2.6-11)$$

Integrating by part the right term of Eq. (2.6-10) using the fact that $x'(t)$ and $\exp\{-j 2\pi nt/T\}$ are periodic yields

$$\lambda_n = \frac{1}{4\pi^2 n^2} \int_0^T x''(t) \exp \left\{ -j \frac{2\pi nt}{T} \right\} dt \quad (2.6-12)$$

Consequently,

$$|\lambda_n| \leq \int_0^T |x''(t)| dt \quad (2.6-13)$$

$$|\lambda_n| \leq \int_0^T |x''(t)| dt \quad (2.6-13)$$

The integral

$$\int_0^T |x''(t)| dt$$

is the exact Variation of $x'(t)$; it will be bounded by

$$V_{t=0}^T(x'(t)) \quad (2.6-14)$$

Recall that the purpose of this development is to relate the grid-line spacing (ε) to the number of harmonics (N). If we partition the x-y plane in a square grid with sides ε , stating that 2 curves or vectors are "close" is equivalent to saying that $V_x(t)$ and $V_N(t)$ are within a rectangle ε_x by ε_y at any time. To achieve this, we should set

$$\varepsilon_x = S_{t=0}^T |x_\infty(t) - x_N(t)| \quad (2.6-15)$$

$$= \sum_{n=N+1}^{\infty} |\lambda_n| + \sum_{n=-\infty}^{-N-1} |\lambda_n| \quad (2.6-16)$$

$$\leq 2 \sum_{n=N+1}^{\infty} \frac{T}{4\pi^2 n^2} V_0^T(x'(t))$$

$$\leq \frac{T}{2\pi^2} V_0^T(x'(t)) \sum_{n=N+1}^{\infty} \frac{1}{n^2}$$

We can show that

$$\sum_{n=N+1}^{\infty} \frac{1}{n^2} \leq \frac{1}{N} \quad (2.6-17)$$

hence,

$$\varepsilon_x(N) \leq \frac{T}{2\pi^2 N} V_0^T(x'(t)) . \quad (2.6-18)$$

Similarly, if only M terms of the $y_x(t)$ series are used to construct $y_M(t)$, a limit on the grid-line spacing is imposed by:

$$S_0^T \left| y_x(t) - y_N(t) \right| \leq \frac{T}{2\pi^2 M} V_0^T(y'(t)) \quad (2.6-19)$$

or

$$\varepsilon_y(M) \leq \frac{T}{2\pi^2 M} V_0^T(y'(t)) \quad (2.6-20)$$

In practice M is taken equal to N , unless we intentionally need to compensate for an aspect ratio different from 1; in this case some harmonics will appear in the computation in one direction and not in the other. If we assume that $M=N$, the condition on the grid-line spacing becomes

$$\varepsilon_x = \varepsilon_y = \varepsilon \leq \max \left[S_0^T \left| x_\infty(t) - x_N(t) \right|, S_0^T \left| y_\infty(t) - y_N(t) \right| \right]$$

$$\varepsilon \leq \frac{T}{2\pi^2 M} \max \left[V_0^T(x'(t)), V_0^T(y'(t)) \right] \quad (2.6-21)$$

Equivalently, if the grid-line spacing is already fixed, N harmonics will construct a curve within an error ε providing that N verifies:

$$N \leq \frac{T}{2\pi^2 \varepsilon} \max \left[V_0^T(x'(t)), V_0^T(y'(t)) \right] \quad (2.6-22)$$

Until now all the derived expressions depend on $V_0^T(x'(t))$ and $V_0^T(y'(t))$; let us relate these quantities to the chain-code elements. We defined $V_0^T(x'(t))$ by

$$V_0^T(x'(t)) = \int_0^T \left| x''(t) \right| dt \quad (2.6-23)$$

Since $x(t)$, $x'(t)$, and $x''(t)$ are piece-wise constant, continuous, and differentiable,

$$V_0^T(x'(t)) = \sum_{i=1}^N \left| x'_i - x'_{i-1} \right| \quad (2.6-24)$$

with

$$x'_0 = x'_N \quad (2.6-25)$$

where x'_i is the projection on the x -axis of the speed along link i .

Assuming similar notation for $V_0^T(y'(t))$, the following relations hold true:

$$\begin{aligned} V_0^T(y'(t)) &= \int_0^T \left| y'(t) \right| dt \\ &= \sum_{i=1}^N \left| y'_i - y'_{i-1} \right| \end{aligned} \quad (2.6-26)$$

with

$$y'_0 = y'_N \quad (2.6-27)$$

where y'_i is the projection on the y-axis of the speed along link i . With reference to Figs. (2.4-1) and (2.4-2), the values of x'_i and y'_i required to compute $V_0^T(x'(t))$ and $V_0^T(y'(t))$ are listed in Table (3.2-1).

Table (3.2-1): 8-N and 4-N projections of travel speed along link a_i

a_i	x'_i	y'_i	a_i	x'_i	y'_i
0	1	0			
1	$\sqrt{2}/2$	$\sqrt{2}/2$	0	1	0
2	0	1	1	0	1
3	$\sqrt{2}/2$	$-\sqrt{2}/2$	2	-1	0
4	-1	0	3	0	-1
5	$-\sqrt{2}/2$	$-\sqrt{2}/2$	4-N		
6	0	-1			
7	$\sqrt{2}/2$	$-\sqrt{2}/2$			
8-N					

2.7. Elliptic Properties of the Fourier Features

In this section we shall show that the Fourier Descriptors defined in the previous section exhibit a very useful property which allows each 2-dimensional projection of a closed-line to be broken into an infinitude of harmonically related ellipses. For common contour shapes used in matching and recognition, the size of the ellipses diminishes rapidly with increasingly higher harmonics. This property will enable us to normalize the Fourier Descriptors in a very efficient way.

Let us start by defining the following:

$$X_n = \alpha_n \cos \frac{2\pi nt}{T} + \beta_n \sin \frac{2\pi nt}{T} \quad (2.7-1)$$

$$Y_n = \gamma_n \cos \frac{2\pi nt}{T} + \delta_n \sin \frac{2\pi nt}{T} \quad (2.7-2)$$

$$u_n = (X_n, Y_n)' \quad (2.7-3)$$

$$e_n = \left(\cos \frac{2\pi nt}{T}, \sin \frac{2\pi nt}{T} \right) \quad (2.7-4)$$

$$F_n = \begin{bmatrix} \alpha_n & \beta_n \\ \gamma_n & \delta_n \end{bmatrix} \quad (2.7-5)$$

With these definitions, a curve can be described as

$$x(t) = X_0 + \sum_{n=1}^N X_n \quad (2.7-6)$$

$$y(t) = Y_0 + \sum_{n=1}^N Y_n \quad (2.7-7)$$

As we have shown for the Fourier Descriptors defined as a function of the curve arc-length, one way of normalizing them would be to drop the DC component which is only a bias term, one not carrying shape information that might discriminate between contours during matching and recognition phases. The space DC components X_0 and Y_0 also are dropped here so that the rebuilt contours would show in the same space.

Using the last notation we have

$$u_n = F_n e_n \quad (2.7-8)$$

$$e_n = F_n^{-1} u_n .$$

Since e_n is a unitary vector,

$$\begin{aligned} (F_n^{-1} u_n)' (F_n^{-1} u_n) &= 1 \\ u_n' (F_n^{-1})' (F_n^{-1}) u_n &= 1 \end{aligned} \quad (2.7-9)$$

If the matrix

$$(F_n^{-1})' F_n^{-1}$$

is denoted by G_n , then

$$(F_n^{-1})' F_n = G_n = \frac{1}{(\alpha_n^2 \delta_n^2 - \beta_n^2 \gamma_n^2)} \begin{bmatrix} \gamma_n^2 + \delta_n^2 & -(\alpha_n \beta_n + \beta_n \delta_n) \\ -(\alpha_n \gamma_n + \beta_n \delta_n) & \alpha_n^2 + \beta_n^2 \end{bmatrix} \quad (2.7-10)$$

Equation (3.3-9) can be rewritten as

$$u_n' G_n u_n = 1 \quad (2.7-11)$$

which is the standard equation of an ellipse. Equivalently,

$$\frac{\frac{X_n^2}{\alpha_n^2 \delta_n^2 - \beta_n^2 \gamma_n^2}}{\gamma_n^2 + \delta_n^2} + \frac{\frac{X_n Y_n}{\beta_n^2 \gamma_n^2 - \alpha_n^2 \delta_n^2}}{2(\alpha_n \gamma_n + \beta_n \delta_n)} + \frac{\frac{Y_n^2}{\alpha_n^2 \delta_n^2 - \beta_n^2 \gamma_n^2}}{\alpha_n^2 + \beta_n^2} = 1 \quad (2.7-12)$$

Since the terms $x(t)$ and $y(t)$ are summations over the X_n and Y_n space series, a contour can be seen as a superposition of phasor ellipses with the proper magnitude and phases. Intuitively, when we look at a contour, we first get an idea about its global shape, which is carried mathematically by α_1 , β_1 , γ_1 , and δ_1 , the first degree phasor. As we get into details by perceiving smaller features, the higher order components enter into the picture. As we shall see in the example which will be presented in a following section, smooth contours can be satisfactorily recovered by using as few as 8 phasors. Conversely, if a contour contains sharp

corners and high curvature points, a surprisingly high number of phasors are required to recover the original contour within a specified error bound. This latter remark suggests that matching and recognition techniques can be used only in situations where a global shape description is sufficient for discrimination. If features such as angle and corners are dominant and compulsory for recognition, this technique is of little help.

We now propose a way of making the Fourier features independent of rotation and starting point. In order to do this, we shift the time $t' = t + \tau$ and analyze the effect of this shift on the α_n 's, β_n 's, γ_n 's, and δ_n 's. Equation (3.3-8) can be written for t' as

$$\begin{bmatrix} X_n(t+\tau) \\ Y_n(t+\tau) \end{bmatrix} = \begin{bmatrix} \alpha_n & \beta_n \\ \gamma_n & \delta_n \end{bmatrix} \begin{bmatrix} \cos \frac{2\pi n}{T} (t+\tau) \\ \sin \frac{2\pi n}{T} (t+\tau) \end{bmatrix} \quad (2.7-13)$$

Decomposing the sine and cosine terms yields new dependency coefficients between $u_n(t')$ and $e_n(t')$ which we shall denote by

$\alpha_n^\tau, \beta_n^\tau, \gamma_n^\tau, \text{ and } \delta_n^\tau$:

$$\alpha_n^\tau = \alpha_n \cos \frac{2\pi n\tau}{T} + \beta_n \sin \frac{2\pi n\tau}{T} \quad (2.7-14)$$

$$\beta_n^\tau = -\alpha_n \sin \frac{2\pi n\tau}{T} + \beta_n \cos \frac{2\pi n\tau}{T} \quad (2.7-15)$$

$$\gamma_n^t = \gamma_n \cos \frac{2\pi n\tau}{T} + \delta_n \sin \frac{2\pi n\tau}{T} \quad (2.7-16)$$

$$\delta_n^t = -\gamma_n \sin \frac{2\pi n\tau}{T} + \delta_n \cos \frac{2\pi n\tau}{T} \quad (2.7-17)$$

or equivalently,

$$\begin{bmatrix} \alpha_n^t & \beta_n^t \\ \gamma_n^t & \delta_n^t \end{bmatrix} = \begin{bmatrix} \alpha_n & \beta_n \\ \gamma_n & \delta_n \end{bmatrix} \begin{bmatrix} \cos \frac{2\pi n\tau}{T} & -\sin \frac{2\pi n\tau}{T} \\ \sin \frac{2\pi n\tau}{T} & \cos \frac{2\pi n\tau}{T} \end{bmatrix} \quad (2.7-18)$$

Taking the transpose of both sides yields

$$\begin{bmatrix} \alpha_n^t & \gamma_n^t \\ \beta_n^t & \delta_n^t \end{bmatrix} = \begin{bmatrix} \cos \frac{2\pi n\tau}{T} & \sin \frac{2\pi n\tau}{T} \\ -\sin \frac{2\pi n\tau}{T} & \cos \frac{2\pi n\tau}{T} \end{bmatrix} \begin{bmatrix} \alpha_n & \gamma_n \\ \beta_n & \delta_n \end{bmatrix} \quad (2.7-19)$$

The latter matrix relation shows that when the starting point is shifted with τ , the Fourier features are "rotated" with an angle $\theta_n = n \cdot 2\pi\tau/T$. Note that this angle depends on n ; if the first phasor is rotated by θ_1 , the second phasor is rotated by $2\theta_1$, and so forth. When the starting point travels along the contour in one direction, the phasors will rotate in the other direction with a speed higher and higher for higher order phasors. By the time the first phasor has finished its complete revolution, the

phasor of order n would have completed its n^{th} complete turn. Note that the matrix transformation given in Eq. (3.3-19) is a rotation in the phasors' space. That holds true only for a set of phasors of the same order because of the dependency of the rotation angle on n . This transformation is not orthogonal, nor does it preserve distance within the phasors' space.

2.8. Normalization of the Elliptic Descriptors

One way of normalizing the Fourier features calculated from an arbitrary starting point is to keep constant the angle between a chosen phasor and one of the coordinate system axes. Since we have deleted the zero-order phasor because it does not carry any shape information, a natural way of normalizing the elliptic descriptors would be to align, say, the x -axis with the major axis of the first-order ellipse. Obviously all other choices would be plausible. In fact, in some applications it is more meaningful to normalize the Fourier features by aligning these axes with a specific phasor which carries better information than the first phasor for the problem at hand [41].

If we decide to align the x -axis with the positive (right-half plane) of phasor n , we proceed as follows. The n^{th} phasor components can be derived from Eq. (3.4-1) by setting $t' = 0$.

$$\begin{bmatrix} X_n(t') \\ Y_n(t') \end{bmatrix} = \begin{bmatrix} \alpha_n^\tau & \beta_n^\tau \\ Y_n^\tau & \delta_n^\tau \end{bmatrix} \begin{bmatrix} \cos \frac{2\pi n\tau'}{T} \\ \sin \frac{2\pi n\tau'}{T} \end{bmatrix} \quad (2.8-1)$$

or

$$\begin{bmatrix} X_n \\ Y_n \end{bmatrix} = \begin{bmatrix} \alpha_n^\tau & \beta_n^\tau \\ Y_n^\tau & \delta_n^\tau \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad (2.8-2)$$

$$X_1 = \alpha_n^\tau \quad (2.8-3)$$

$$Y_1 = \alpha_n^\tau \quad (2.8-4)$$

The x-axis is aligned with the positive major axis of the nth phasor if

$$\frac{d}{d\tau} \left\{ (\alpha_n^\tau)^2 + (Y_n^\tau)^2 \right\} = 0 \quad , \quad (2.8-5)$$

$$\frac{d^2}{d\tau^2} \left\{ (\alpha_n^\tau)^2 + (Y_n^\tau)^2 \right\} > 0 \quad , \quad (2.8-6)$$

and

$$-\frac{\pi}{2} < \frac{2\pi n\tau}{T} \leq \frac{\pi}{2} \quad .$$

Using Eqs. (3.4-3) and (3.4-4) and denoting $\cos(2\pi n\tau/T)$ and $\sin(2\pi n\tau/T)$ by c and s , respectively, Eq. (3.4-5) can be written as

$$\begin{aligned} \frac{d}{d\tau} \left\{ c^2 \alpha_n^2 + s^2 \beta_n^2 + c^2 \gamma_n^2 + s^2 \delta_n^2 + 2 \cos \alpha_n \beta_n + 2 \cos \gamma_n \delta_n \right\} &= 0 \\ 2 \sin \frac{2\pi n \tau}{T} \left\{ -\alpha_n^2 + \beta_n^2 - \gamma_n^2 + \delta_n^2 \right\} + 2 \cos 2 \frac{2\pi n \tau}{T} \left\{ \alpha_n \beta_n + \gamma_n \delta_n \right\} &= 0 \end{aligned} \quad (2.8-7)$$

Equivalently

$$\tau_n = \frac{T}{4\pi n} \tan^{-1} \left\{ \frac{2(\alpha_n \beta_n + \gamma_n \delta_n)}{\alpha_n^2 - \beta_n^2 + \gamma_n^2 - \delta_n^2} \right\} \quad (2.8-6)$$

For an alignment along the first phasor,

$$\tau_1 = \frac{T}{4\pi} \tan^{-1} \left\{ \frac{2(\alpha_1 \beta_1 + \gamma_1 \delta_1)}{\alpha_1^2 - \beta_1^2 + \gamma_1^2 - \delta_1^2} \right\} \quad (2.8-7)$$

and the corresponding rotation angle is

$$\eta_1 = \frac{2\pi \tau_1}{T}$$

The transformed features are in general

$$\begin{bmatrix} \alpha_n^\tau & \gamma_n^\tau \\ \beta_n^\tau & \delta_n^\tau \end{bmatrix} = \begin{bmatrix} \cos n\eta_1 & \sin n\eta_1 \\ -\sin n\eta_1 & \cos n\eta_1 \end{bmatrix} \begin{bmatrix} \alpha_n & \gamma_n \\ \beta_n & \delta_n \end{bmatrix}$$

The latter relation expresses the effect of the time shift only; to align effectively the axes with the n th phasor we rotate the original x - and y -axes by ϕ_n , which satisfies Eqs. (3.4-5) and (3.4-6). Readily,

$$\tan \phi_n = \frac{\gamma_n^\tau}{\alpha_n^\tau} \quad (2.8-8)$$

or

$$\tan \phi_1 = \frac{\gamma_1^\tau}{\alpha_1^\tau}$$

for $\tau = \tau_1$. The transformed features which will be termed by

$$\alpha_n^{\tau\phi}, \beta_n^{\tau\phi}, \gamma_n^{\tau\phi}, \text{ and } \delta_n^{\tau\phi}$$

are given by

$$\begin{bmatrix} \alpha_n^{\tau\phi} & \beta_n^{\tau\phi} \\ \gamma_n^{\tau\phi} & \delta_n^{\tau\phi} \end{bmatrix} = \begin{bmatrix} \cos \phi_1 & \sin \phi_1 \\ -\sin \phi_1 & \cos \phi_1 \end{bmatrix} \begin{bmatrix} \alpha_n & \beta_n \\ \gamma_n & \delta_n \end{bmatrix} \begin{bmatrix} \cos n\eta_1 & -\sin n\eta_1 \\ \sin n\eta_1 & \cos n\eta_1 \end{bmatrix} \quad (2.8-9)$$

In the following section we shall present the example of a square decomposition in the elliptic space.

2.9. Illustrative Example

To illustrate the use of the technique presented in Sections 2.5 through 2.8, we experiment here with a fairly easy example. Assume we have to decompose the square of side a having 2 of its sides on the x - and y -axis and lying completely in the first quadrant. If this square is traversed in a counterclockwise direction at constant speed v ,

the projections $x(t)$ and $y(t)$, as shown in Fig. 2.9-1 through 2.9-3, are given by:

$$x(t) = \begin{cases} vt & a \leq t < a/v \\ a & a/v \leq t < 2a/v \\ -vt + 3a/v & 2a/v \leq t < 3a/v \\ 3a/v & 3a/v \leq t < 4a/v \end{cases} \quad (2.9-1)$$

$$y(t) = \begin{cases} 0 & 0 \leq t < a/v \\ vt + a/v & a/v \leq t < 2a/v \\ a & 2a/v \leq t < 3a/v \\ -vt + 4a/v & 3a/v \leq t < 4a/v \end{cases} \quad (2.9-2)$$

Following the notation used in Section 2.5 we have

$$a_0 = \frac{1}{T} \int_0^T x(t) dt \quad (2.9-3)$$

with

$$T = \frac{4a}{v} \quad (2.9-4)$$

$$a_0 = \frac{v}{4a} \int_0^{\frac{4a}{v}} x(t) dt = \frac{a}{2} \quad (2.9-5)$$

The general form of the harmonics a_n is given by

$$a_n = \frac{2}{T} \int_0^T x(t) \cos \frac{\pi nvt}{2a} dt \quad (2.9-6)$$

$$= \frac{v}{2a} \int_0^{\frac{4a}{v}} x(t) \cos \frac{\pi nvt}{2a} dt \quad (2.9-7)$$

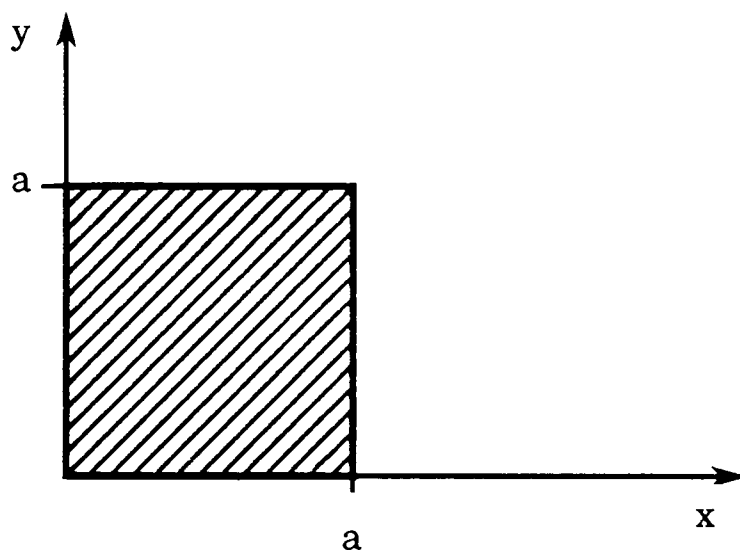


Figure 2.9-1 Decomposition of a Square in harmonics

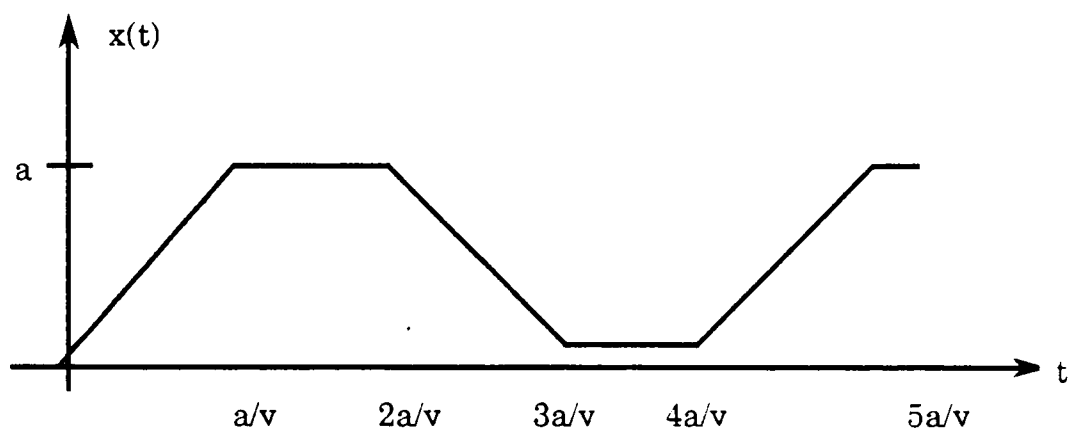


Figure 2.9-2 Projection $x(t)$.

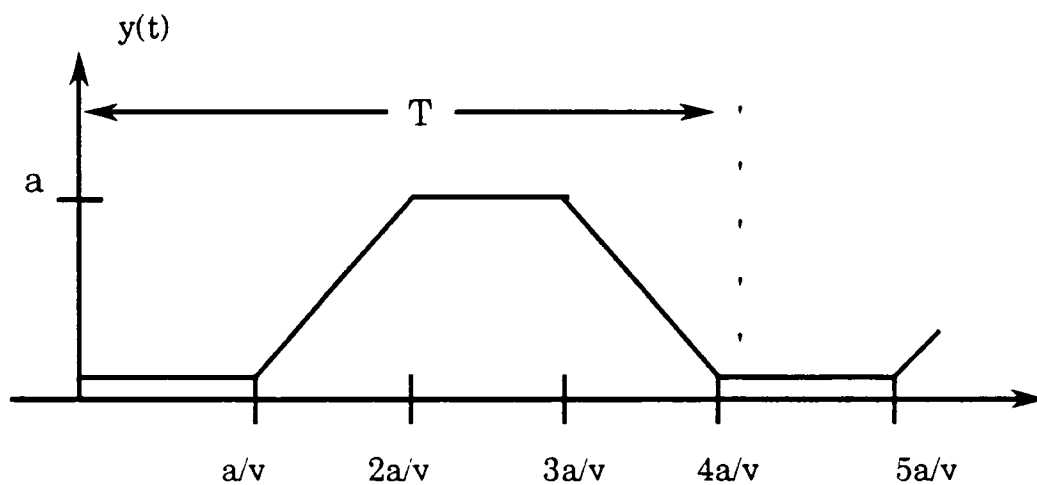


Figure 2.9-3. Projection $y(t)$

making use of the analytical expressions of $x(t)$, we show that

$$\alpha_n = \frac{4a \cos n\pi}{\pi^2 n^2} \quad (2.9-8)$$

Using similar computations:

$$\beta_n = \frac{4a}{\pi^2 n^2} \sin n \frac{\pi}{2} \quad (2.9-9)$$

and consequently

$$x(t) = \frac{a}{2} + \frac{4a}{\pi^2} \sum_{n=1}^{\infty} \frac{\cos n\pi}{n^2} \cos \frac{\pi n v t}{2a} + \frac{\sin n\pi/2}{n^2} \sin \frac{\pi n v t}{2a}$$

The expressions of γ_n and δ_n can be derived as in Eqs. (2.9-8) and (2.9-9), but we see that

$$y(t) = x\left(t - \frac{a}{v}\right) \quad (2.9-10)$$

hence,

$$\begin{aligned} y(t) &= \alpha_0 + \sum_{n=1}^{\infty} \alpha_n \cos \left\{ \frac{\pi n v}{2a} \left(t - \frac{a}{v} \right) \right\} + \beta_n \sin \left\{ \frac{\pi n v}{2a} \left(t - \frac{a}{v} \right) \right\} \\ &= \alpha_0 + \sum_{n=1}^{\infty} \alpha_n \cos \left(\frac{\pi n v}{2a} - n \frac{\pi}{2} \right) + \beta_n \sin \left(\frac{\pi n v}{2a} - n \frac{\pi}{2} \right) \\ &= \alpha_0 + \sum_{n=1}^{\infty} \alpha_n \sin \frac{\pi n v}{2a} \sin n \frac{\pi}{2} - \beta_n \sin \frac{\pi}{2} \cos \frac{\pi n v}{2a} \end{aligned}$$

The corresponding expressions of γ_n and δ_n are

$$\gamma_0 = \alpha_0 \quad (2.9-11)$$

$$\gamma_n = -\beta_n \sin n \frac{\pi}{2} \quad (2.9-12)$$

$$\delta_n = \alpha_n \sin n \frac{\pi}{2} \quad (2.9-13)$$

or

$$\gamma_0 = \frac{a}{2} \quad (2.9-14)$$

$$\gamma_n = \frac{-4a}{\pi^2 n^2} \quad (2.9-15)$$

$$\delta_n = \frac{-4a}{\pi^2 n^2} \quad (2.9-16)$$

The projection $y(t)$ can then be written as

$$y(t) = \frac{a}{2} - \frac{4a}{\pi^2} \sum_{n=1}^{\infty} \frac{1}{n^2} \cos \frac{\pi n v}{2a} + \frac{1}{n^2} \sin \frac{\pi n v t}{2a} \quad (2.9-17)$$

It is worth noting that all the expressions of α_n , β_n , γ_n , and δ_n were independent of the speed at which the contour is traversed. This fact has no contradiction since α_n , β_n , γ_n and δ_n measure repetitiveness which can be sensed at any speed, providing that speed is constant throughout the full scan of the contour.

To illustrate the remarks made in Section 2.7 regarding the decomposition of a general contour form into basic harmonics (ellipses), we construct the square shown in Fig. 2.9-1 by adding phasors of higher and higher order. In our case, to ease the computation involved, we have

chosen a square which has a circular symmetry. Our harmonics will then be circles. As shown in the different steps of reconstruction [Fig. 2.9-4], the global shape of the contour is filled in by the first harmonics, and then smaller and smaller details are completed by higher harmonics.

In this chapter we have covered the theoretical concepts of curve definition, shape description, and chain-code matching techniques. They will be applied in Chapter 4.

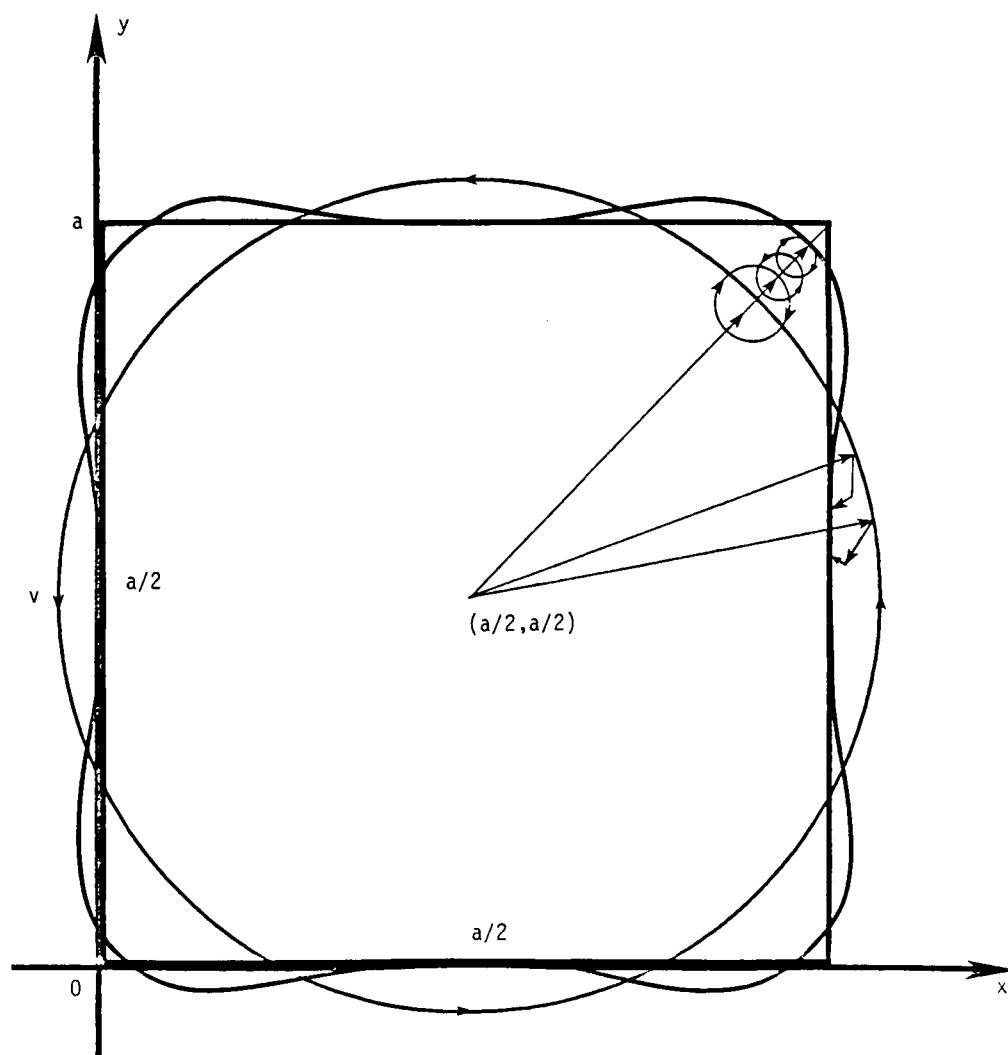


Figure 2.9-4. Phasor decomposition.

CHAPTER 3

MOTION DETECTION VIA AREA CORRELATION

3.1. Tracking by Correlation

To place this problem of tornado detection and location in a suitable mathematical framework which aids in Image Processing, a theoretical matching model should be developed. For a 2-D or 3-D scene, suppose that at each spatial coordinate $\underline{x}$ and time t , we have 2 sensors, one measuring the background textual intensity, and the other measuring the tracked object intensity. From this hypothetical decomposition, the total image intensity can be reconstructed by weighting these 2 components in the following manner [43]:

$$P(\underline{x},t) = f(\underline{x},t) c(\underline{x},t) + [1-c(\underline{x},t)] b(\underline{x},t) \quad (3.1-1)$$

$P(\underline{x},t)$ = picture function intensity at the sensor location $\underline{x}$ and time t .

$b(\underline{x},t)$ = background textual function intensity at the sensor location $\underline{x}$ and time t .

$f(\underline{x},t)$ = tracked object function intensity at the sensor location $\underline{x}$ and time t .

$c(\underline{x},t)$ = scene characteristic function at the sensor location $\underline{x}$ and time t .

$\underline{x} = (x,y,z)$ ' for the 3-D case.

$\underline{x} = (x,y)'$ for the 2-D case.

The expression $c(\underline{x},t)$ can be viewed as a weighting function which is introduced to separate the background from the tracked object. If the picture formation can be realized in a controllable manner, the resulting intensities would carry valuable information about the tracked object. Ideally, in a single-target problem, the scene characteristic function can be defined as

$$c(\underline{x},t) = d(\underline{x} - \underline{x}_0, t - t_0) \quad (3.1-2)$$

if the target is detected at the position $\underline{x}_0$ at time t_0 . The impulse Dirac function is denoted by $d(\bullet)$. The latter formulation of $c(\underline{x},t)$ is sometimes referred to as a "mask" function which "turns on" and "turns off" the target. It can be defined in several forms such as a thresholding function regionwise constant, a filtering transfer function in the spatial domain, and a number of others.

For the remainder of this section $c(\underline{x},t)$ is chosen to be a thresholding function to separate the regions of interest from the background. This choice is not valid if the processing of the picture includes operations in the frequency domain. The sharp transitions introduced by the scene characteristic function can result in a ringing effect [18] which might induce a significant perturbation into the picture function distribution, not only in the neighborhood of the sharp transitions, but in the whole picture as well [43,44]. In our work, the preprocessing phase did not exceed a simple local averaging. This was done in order to reduce the effect of some scattered data. Despite the fact

that each pixel of these radar images is an average of about 1,600 samples taken along several radials of the radar beam, abrupt changes are noticeable between adjacent data points. This is mainly due to the spatial distribution of the tracked object (clouds) and the high noise level in radar images [15,44].

The matching algorithms which are being developed rely on the basic assumption that in time-varying imagery the appearance of the images processed does not change significantly from frame to frame. Hence, a matching method based on correlation would be a good candidate for the problem at hand. For two 3-D picture function intensities taken at a time delay t_0 , equal to an interframe time, the cross-correlation function [18] is defined by

$$P(x_1, x_2, x_3, t) \cap P(x_1, x_2, x_3, t + t_0) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} P(u_1, u_2, u_3, t) P(u_1 + x_1, u_2 + x_2, u_3 + x_3, t + t_0) du_1 du_2 du_3 \quad (3.1-3)$$

This function measures the similarity between the spatial distribution of the picture intensities at times t and $t + t_0$. Because of the complexity of the data collection scheme and the length of the interframe time, 30 seconds, the comparison algorithms will be performed only on successive frames. The intensity function of the picture at time t will be denoted by $P_1(i, j)$; the intensity function of the picture at time $t + t_0$ will be denoted by $P_2(i, j)$. The continuous spatial coordinates (x_1, x_2, x_3) are replaced by the discrete coordinates (i, j) because

we are matching pictures of the same elevation (x_3 is constant) and using discrete data.

3.2. Cross-Correlation Coefficient in Discrete Images

With reference to image P_1 in Fig. 3.2-1, we consider a $(2n+1)$ by $(2n+1)$ sub-image A_1 at coordinates (i,j) . The objective is to search for the highest correlation of this sub-image in P_2 . The search is carried out in an area B_2 of P_2 of size $2(n+n_1)+1$ by $2(n+n_j)+1$ centered at a point with coordinates $(i+i_s, j+j_s)$, as shown in Figs. 3.2-2 and 3.2-3. Around each point (p, q) of the $2n_1+1$ by $(2n_j+1)$ sub-image A_2 , a new $(2n+1)$ by $2n+1$ sub-image is defined. This sub-image belongs to P_2 while A_1 belongs to P_1 . Corresponding to A_1 , we define $P_{1n}(i,j)$ by:

$$\bar{P}_{1n}(i,j) = \frac{\sum_{x=-n}^n \sum_{y=-n}^n P_1(i+x, j+y)}{(2n+1)^2} \quad (3.2-4)$$

Corresponding to C_2 , we define $P_{2n}(p,q)$ by

$$\bar{P}_{2n}(p,q) = \frac{\sum_{x=-n}^n \sum_{y=-n}^n P_2(p+x, q+y)}{(2n+1)^2} \quad (3.2-5)$$

The cross-correlation coefficient between the two arrays A_1 and C_2 is a function of i and j (as A_1 is), and p and q (as C_2 is) and can be defined as

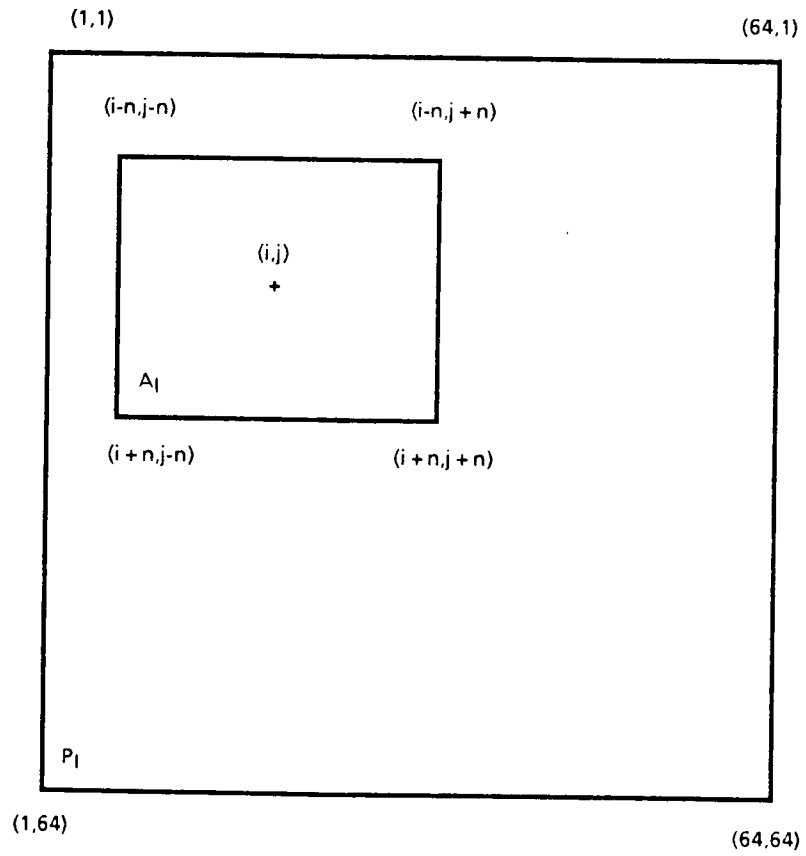
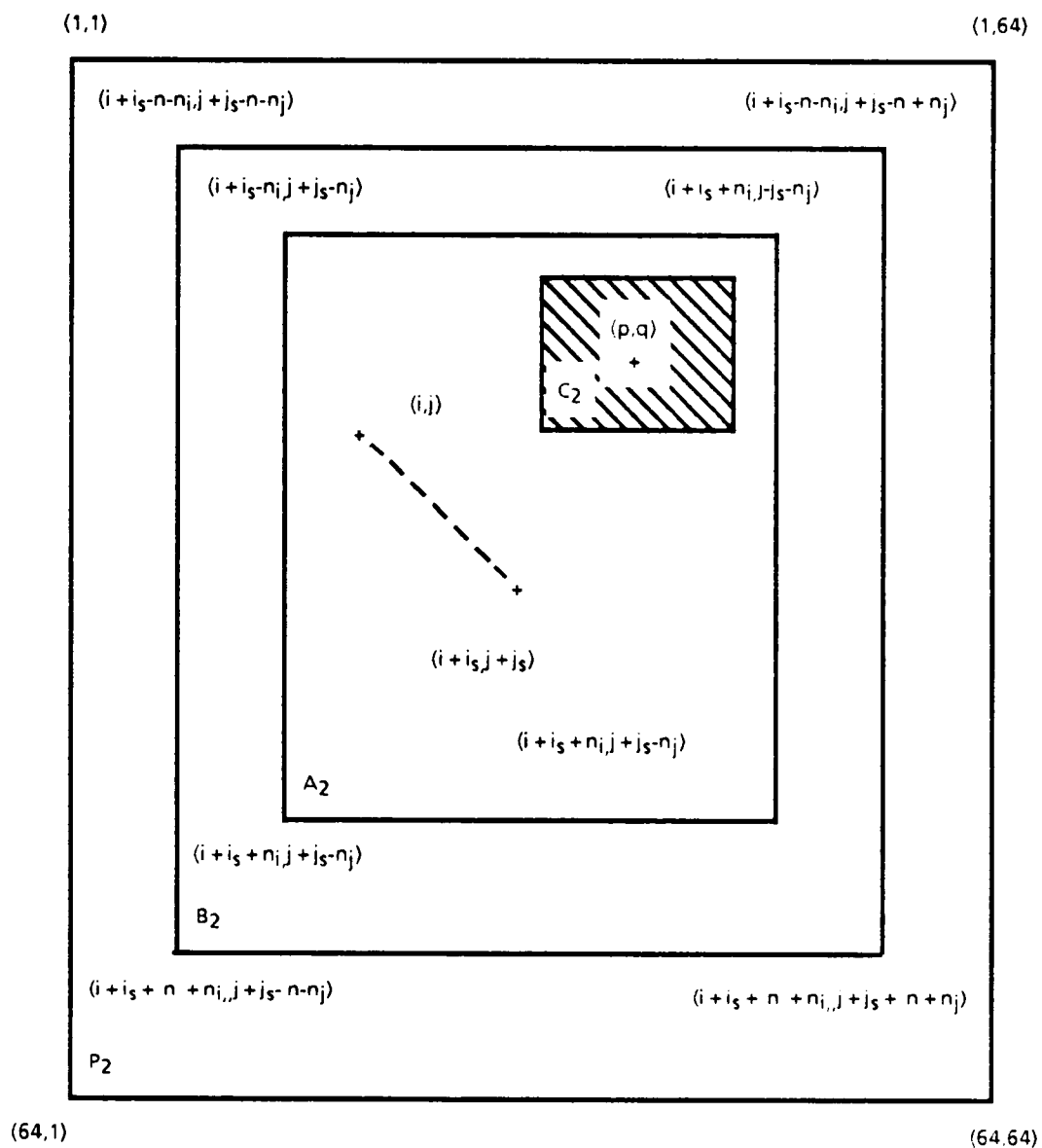


Figure 3.2-1. Initial image P_1 .

Figure 3.2-2. Search image P_2

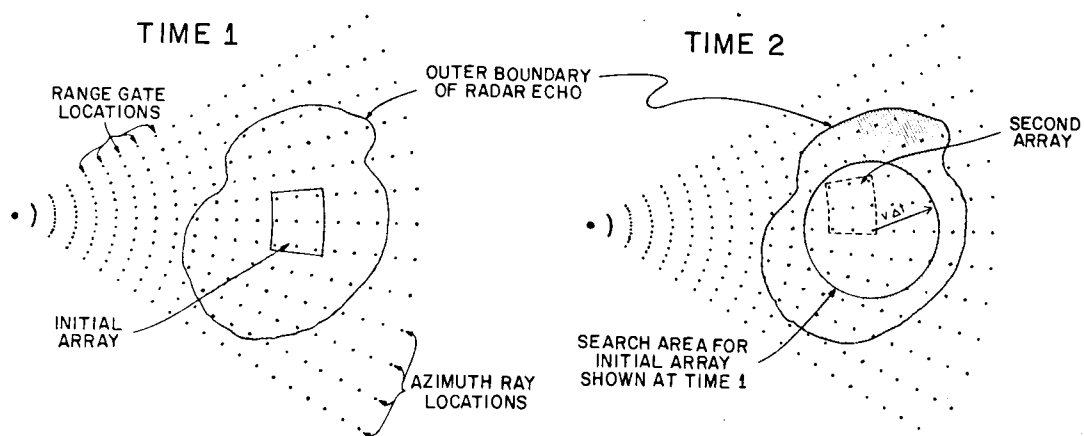


Figure 3.2-3. Representation of a storm scanned by a radar located at the origin.

$$L = \frac{\sum_{x=-n}^n \sum_{y=-n}^n [P_1(i+x, j+y) - \bar{P}_{1n}(i, j)] [P_2(p+x, q+y) - \bar{P}_{2n}(p, q)]}{\left[\sum_{x=-n}^n \sum_{y=-n}^n (P_1(i+x, j+y) - \bar{P}_{1n}(i, j))^2 \sum_{x=-n}^n \sum_{y=-n}^n (P_2(p+x, q+y) - \bar{P}_{2n}(p, q))^2 \right]^{1/2}} \quad (3.2-6)$$

The cross-correlation coefficient is defined for each point (i, j) , and for all the values of p and q where

$$i + i_s - n_i \leq p \leq i + i_s + n_i \quad (3.2-7)$$

$$j + j_s - n_j \leq q \leq j + j_s + n_j. \quad (3.2-8)$$

Thus at each point (i, j) , the $(2n_i + 1)(2n_j + 1)$ values of $L(i, j, p, q)$ can be thought of as a $(2n_i + 1)$ by $(2n_j + 1)$ matrix M , the elements of which represent the cross-correlation coefficients between the initial subimage A_1 and each of the subimages C_2 defined at each point (p, q) of subimage A_2 . With a simple change of the variables p and q to u and v , the expression of $L(i, j, u, v)$ gives the u^{th} and v^{th} element of this matrix; that is,

$$u = p - i - i_s + n_i + 1 \quad (3.2-9)$$

$$v = q - j - j_s + n_j + 1 \quad (3.2-10)$$

$$p = u + i + i_s - n_i - 1 \quad (3.2-11)$$

$$q = v + j + j_s - n_j - 1 \quad (3.2-12)$$

$$1 \leq u \leq 2n_i + 1 \quad (3.2-13)$$

$$1 \leq v \leq 2n_i + 1 \quad (3.2-14)$$

and

$$L(i,j,p,q) = \frac{\sum_{x=-n}^n \sum_{y=-n}^n (P_1 - \bar{P}_{1n})(P_2 - \bar{P}_{2n})}{\left[\sum_{x=-n}^n \sum_{y=-n}^n (\bar{P}_1 - \bar{P}_{1n})^2 (P_2 - \bar{P}_{2n})^2 \right]^{1/2}} \quad (3.2-15)$$

where

$$P_1 = P_1(i+x, j+y) \quad (3.2-16)$$

$$\bar{P}_{1n} = \bar{P}_{1n}(i,j) \quad (3.2-17)$$

$$P_2 = P_2(u+i+i_s-n_i+x-1, v+j+j_s-n_j+y-1) \quad (3.2-18)$$

$$\bar{P}_{2n} = \bar{P}_{2n}(u+i+i_s-n_i+x-1, v+j+j_s-n_j+y-1) \quad (3.2-19)$$

For every value of i and j , the function $L(i,j,u,v)$ is a two-dimensional array; it will be termed the likelihood-ratio matrix. It is important to note that the size of this matrix is independent of the size of the $(2n+1)$ by $(2n+1)$ subregion of P_1 initially chosen, which will be referred to as an operator in subsequent discussions. The choice of n will be discussed in detail in the following section. The values of n_i and n_j determine the size of the search area. Their choice depends on the size of the operator and the maximum expected travel distance of point (i,j) in the time interval t_0 . In all the experiments discussed in this work, n_i and

n_j were chosen equal to approximately twice the size of the operator. The parameters i_s and j_s allow a reduction in computation by predicting a region for the location of (i,j) in P_2 based on knowledge of wind direction. If this knowledge is not available, these parameters can be set to zero, and the size of the search area increased [Fig. 3.2-1].

In conventional correlation (often called template or prototype matching), the search procedure is terminated by assigning to the area A_1 , the subimage C_2 of B_2 , which maximizes the function $L(i,j,u,v)$ with respect to u and v . This assignment is mathematically valid. In our case, where we have to build a classifier the decisions of which are going to be physically interpreted, this assignment should be statistically valid. In other words, the dichotomization process should be done with a fairly low probability error, which is a merit figure translating the level of confidence placed in the classifier. The more selective the similarity measure $L(i,j,u,v)$ is, the lower is the probability error associated with it. On the other hand, if the similarity measure is not selective, the classifier decision is ineffective because its randomness generally results in a high probability error. Under this condition, the classifier is no longer optimum.

When applying conventional correlation to the set of data mentioned in Sections 3.4 and 4.2, several results were meaningless. This failure was the result of a high degree of randomness in the assignments which were based on maximizing $L(i,j,u,v)$ with respect to u and v . This failure precluded the use of a straightforward maximization method. Instead, a strategy in which the assignment mentioned earlier

takes into account the distribution of $L(i,j,u,v)$ as a function of u and v was implemented. For this purpose, we defined a local function $f(u,v)$ at each point (i,j) which has the value of the Likelihood-ratio matrix at (u,v) . This function f , like L , is bounded by -1 and $+1$, where -1 corresponds to maximum dissimilarity, and $+1$ corresponds to maximum similarity. In this work the range of interest is 0 to $+1$ because our focus is on detecting similarity in reflectivity data, which is expressed only by positive quantities.

With reference to the structure of the function $f(u,v)$, we distinguish 3 important cases.

Case 1: The function has one large maximum (i.e., typical value 0.9) around an optimal point (i_{opt}, j_{opt}) , which may be surrounded by some local maxima the heights of which are clearly dominated by the first one (i.e., typical value of 0.3). A clear-cut decision can be made by assigning the spatial coordinates (i_{opt}, j_{opt}) as a unique, highly probable mapping of the spatial coordinates (i,j) . For properly chosen operators and search areas, this form of $f(u,v)$ is common throughout regions containing distinguishable features.

Case 2: The function has a fairly large maximum (i.e., typical value of 0.6) along with one or more peaks slightly dominated by the first one (i.e., typical value of 0.3). Unfortunately, this form is the typical response of the likelihood ratio matrix to a small size operator (i.e., 2 by 2 or 3 by 3) and pictures without dominant features. The presence of very narrow and high peaks in the function is meaningless in this case. In fact, the operator is so small that the two compared regions could be

found to be almost identical without being physically related. This is due to a lack of consistency in the comparison procedure under this scheme of detection. Under these conditions, a decision function cannot be derived to forecast the new position of the portion of the storm surrounding the spatial coordinates (i,j) .

Case 3: The function has a fairly flat form throughout the entire space. Most values of f for this type of form are in the mid-range (i.e., 0.5 to 0.6). Frequently, this form is a result of a uniform region tracked in a larger uniform region. When the gradient in the initial array is low, the likelihood ratio matrix repeats the average of the intensities of the cell to which the initial array is compared. This averaging process gives a blurring effect which reduces further the chance of recognition. This tracking difficulty in low gradient regions has been encountered in our earlier work [17,30].

The performance of the approach outlined above was noted and commented upon in the final chapter. From the latter remarks we see that the conventional correlation is not successful because of the following reasons:

- (1) The radar cloud data is textureless; abrupt changes are due to noise only. This is traced to a natural physical phenomenon which governs the water-density forming clouds: diffusion.
- (2) Meaningful features occupy large areas of the storm and are far beyond the dimensions of any operator of reasonable size from a computation-efficient standpoint.

- (3) At a low spatial resolution, the data have a significantly high level of "salt and pepper" noise [18]. For an operator a few pixels wide (2,3,5, or 7), the error induces a very high level of uncertainty. This noise is due to the natural movement in clouds. This is stressed even more in violent tornadic storms. Other serious error sources are the radar noise return and the beam width.

To overcome these difficulties and to upgrade the computational efficiency of the whole system we propose the following improvements.

In order to avoid correlating an initial template with a poor candidate from the consecutive image, we pre-scan the search area (A_2) to determine potential matches. To do this, we propose a fast similarity measure based on the Minimum Distance Classifier (MDC), which is defined as follows [18]. Two patterns C_a and C_b are said to be similar if the normalized Euclidean distance between them is below a certain threshold:

$$d(C_a, C_b) = d(a, b) = \frac{\|a - b\|}{\|a\| \|b\|} \quad (3.2-20)$$

If for a subimage C_2 the distance $d(A_1, C_2)$ is higher than a certain threshold distance (chosen to be 0.6 in all the experiments in this work), that subimage is dropped in the following steps [Fig. 1.3-7 (pp. 21)]. As for the remaining C_2 's, those achieving a degree of similarity high enough to be considered for the best match, a correlation test is performed on them. At the end of this test a partial matrix M is

generated for each (i,j) point of the original template. The matrix is partial in the sense that the value of $L(i,j,p,q)$ is computed for all (p,q) points of A_2 which have passed the MDC Test. As was mentioned above, for high values of L (close to 1) we achieve a good match, whereas for low values of the cross-correlation coefficient (close to 0) we achieve a poor match. Realistically speaking, a cross-correlation coefficient lower than 0.7 represents a poor match. Two unit vectors having a cross-correlation coefficient of 0.7 means that these 2 vectors are 45° apart.

In the following experiment we adopted a technique based on a Dismissal Probability Matrix (DPM) technique which works as follows [38]:

- (1) If the matrix coefficients are less than a certain threshold (set to be 0.6 in the experiments of sections 3.5 and 4.2), that coefficient is dropped to zero.
- (2) If all the coefficients are less than that threshold, the corresponding pattern is assumed to be dissipated, and no speed vector is assigned to it.
- (3) For the patterns having non-zero cross-correlation coefficients, the final estimated speed is computed as the weighted average of each possible assignment. The weighting coefficient is the cross-correlation coefficient itself.

The results of the application of this technique to synthetic data are presented in the following section. Chapter 4 deals with the use of this technique too, but in real radar data.

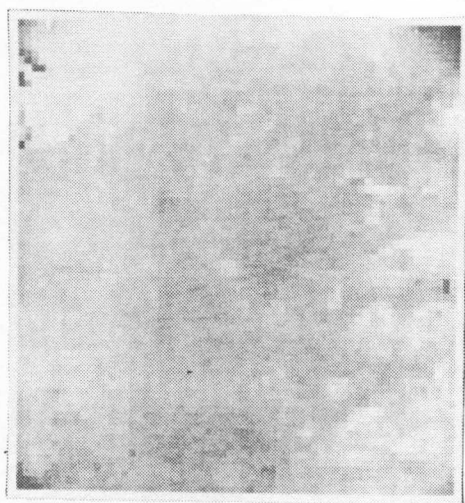
3.3. Detection of Simulated Motion in Random Texture Images

The tracking technique discussed above was evaluated with several simulated and real radar images known to be taken during severe weather conditions. The purpose of these experiments was two-fold: (1) to establish the minimum size of the operator needed to yield reliable circulation detection, and (2) to evaluate the effectiveness of the method in detecting circulation in real data.

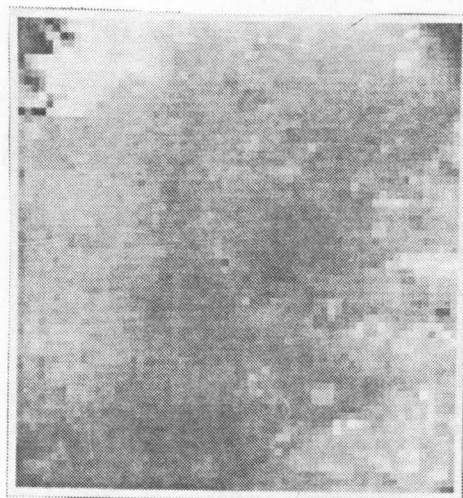
3.3.1. Manual Rotation

In order to study the effects of the size of the operator under carefully controlled conditions, a random texture image was digitized in a 64 by 64 array and noise was added to the resulting digital image in order to reduce the effect of features which would artificially aid the correlation technique. The results are shown in Figs. 3.3-1(a) and 3.3-1(b). Figure 3.3-1(c) shows the difference-image of these images.

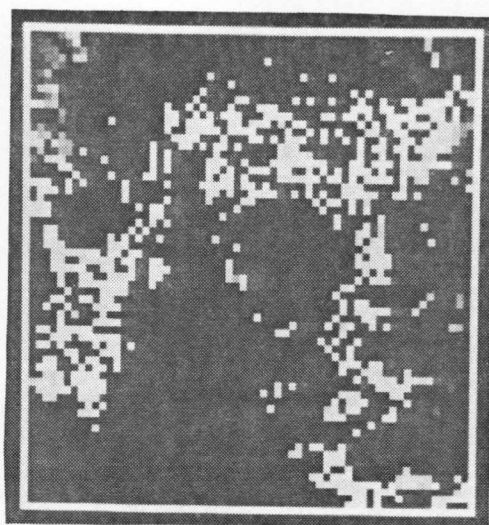
The results of using 7 by 7, 5 by 5, 3 by 3, and down to 2 by 2 operators are shown in Figs. 3.3-2(a) through 3.3-2(d), respectively. It is noted that both the 7 by 7 and 5 by 5 operators were quite effective in detecting the rotation. The 3 by 3 operator gave a good "hint" about the presence and direction of rotation, but a number of TREC (Tracking Radar Echoes by Correlation) vectors are erratic. Finally, the 2 by 2 operator yielded meaningless results. (A 1 by 1 operator also was used and the results were even more random than those shown for the 2 by 2



(a)



(b)



(c)

Figure 3.3-1. (a) Initial noise image, (b) Manual rotation simulation from (a), (c) Difference-image from (a) and (b).

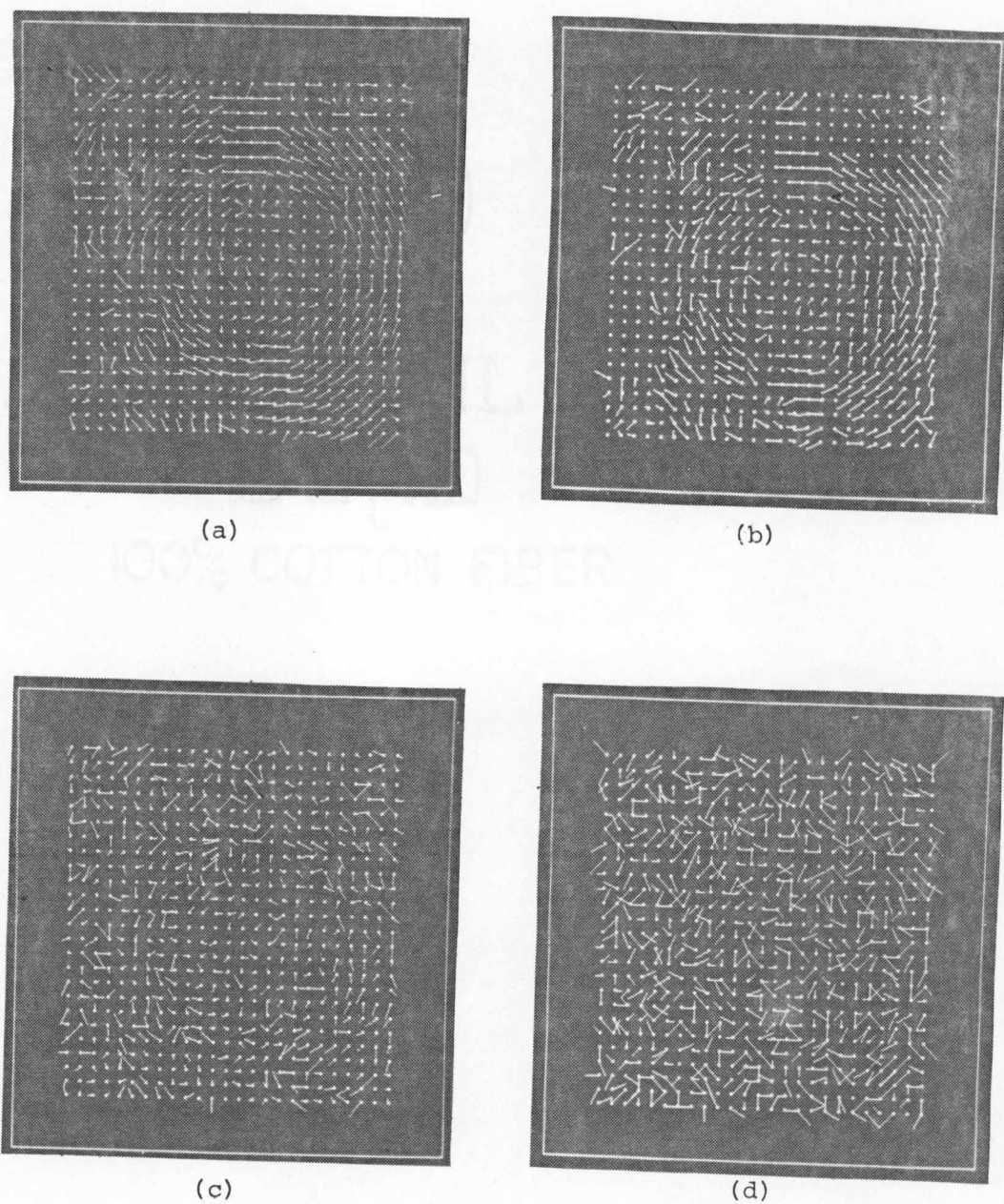


Figure 3.3-2. Circulation detection in noise image using (a) 7 by 7 Operator (b), 5 by 5 Operator, (c) 3 by 3 Operator, and 2 by 2 Operator.

operator.) The reason why operators of less than 5 by 5 failed in this particular case can be attributed to the fact that there were no distinct, trackable features in the image in the small areas defined by these operators.

3.3.2. Computer Rotation

The vortex simulated in the images discussed in the previous section provided a smooth transition between corresponding pixels. In this section, we consider a similar experiment, but with rotation being accomplished by digital techniques. This operation results in a severe truncation of the spatial coordinates and produces a degradation effect similar to the scattering and gathering effects that take place within radar cells (the intensity value of each cell is calculated by integration through the cell area).

The image used in this experiment is shown in Fig. 3.3-3(a), and the results of a 10° digital rotation is shown in Fig. 3.3-3(b). Figure 3.3-3(c) shows the difference-image of these 2 images. The overall rotation effect is much more noticeable in Fig. 3.3-1(c) than in 3.3-3(c), an expected result caused by the dissimilarities introduced by digital truncation. As in the previous section, the tracking algorithm was tested with operators of size 7 by 7, 5 by 5, 3 by 3, and 2 by 2. The results, shown in Figs. 3.3-4(a) through 3.3-4(d), are quite similar to those shown in the previous section, with the exception of the 3 by 3 and 2 by 2 operators. These two

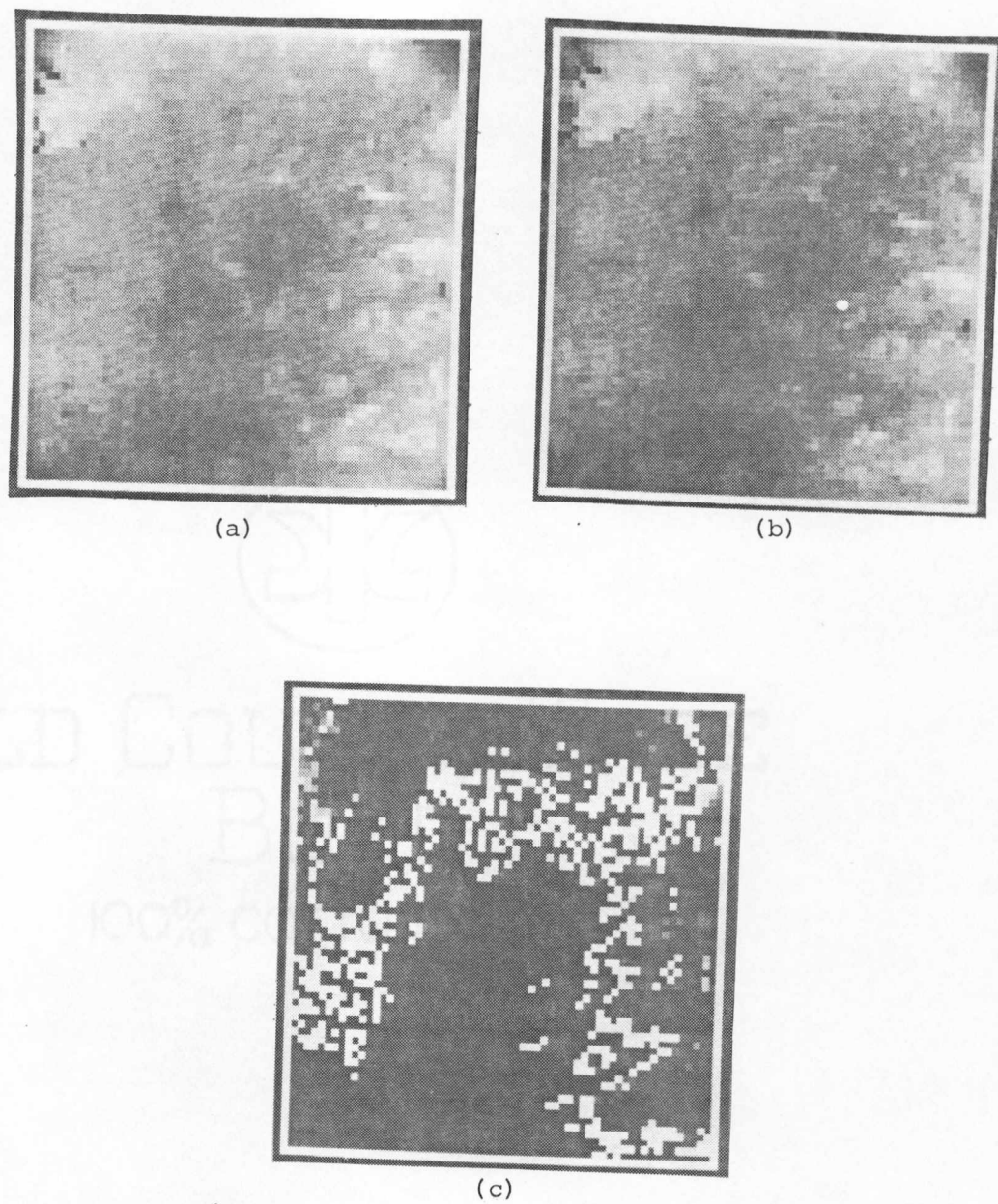
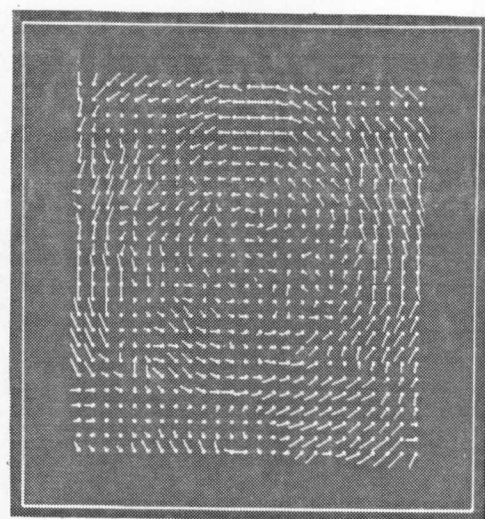
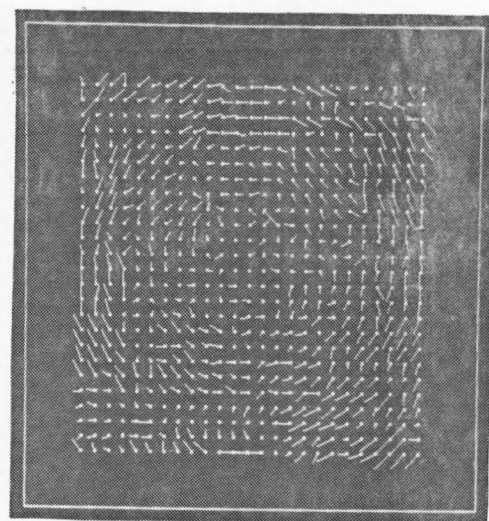


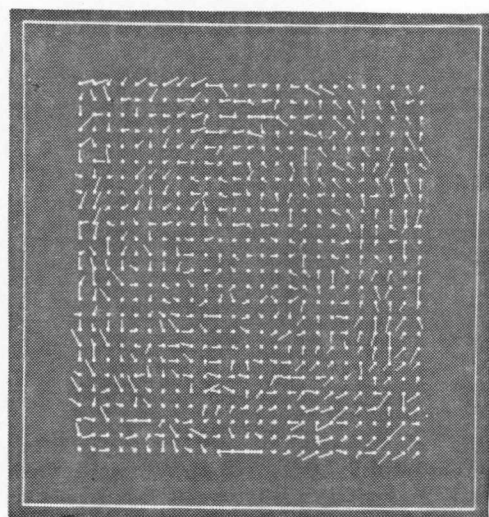
Figure 3.3-3. (a) Initial noise image, (b) Computer-simulated rotation from (a), (c) Difference-image from (a) and (b).



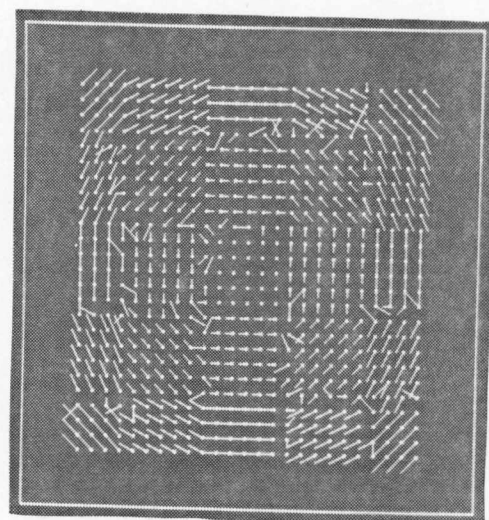
(a)



(b)



(c)



(d)

Figure 3.3-4. Circulation detection of computerized rotation (a) 7 by 7 Operator, (b) 5 by 5 Operator, (c) 3 by 3 Operator, and (d) 2 by 2 Operator.

operators performed slightly better in the digitally rotated picture. This can be attributed to the fact that, due to the digital rotation process, entire regions of size comparable to these operators were essentially moved intact near the center of rotation, and the distortion was still bearable by the shape of the operators. This is verified by noting that performance degraded considerably as a function of distance from the center of rotation. In the outer region, the actual distance traveled by a rotated pixel is considerably larger than that traveled by a pixel closer to the center. This effect is visible even for the 5 by 5 and 7 by 7 operators, which yielded slightly erratic results far from the center of rotation.

3.4. Detection of Simulated Motion in Real Radar Images

As a second, more realistic case of computer rotation we used the radar image shown in Fig. 3.4-1(a). This picture represents a real-case storm which has generated a tornado. The storm, shown from the lowest elevation, covers the largest area up to that time in the sector displayed. A rotated image corresponding to Fig. 3.4-1(a) is shown in Fig. 3.4-1(b). The center of the rotation was chosen to simulate a real-case tornado in such a configuration. The center was situated in the top edge of the storm, and the rotation angle was chosen to be 10° , clockwise. Figure 3.4-1(c) shows the image-difference of the previous two. A careful examination of the difference reveals a few pixels retracing the rotation. Figures 3.4-2(a) through (d) show the results obtained using 7 by 7, 5 by

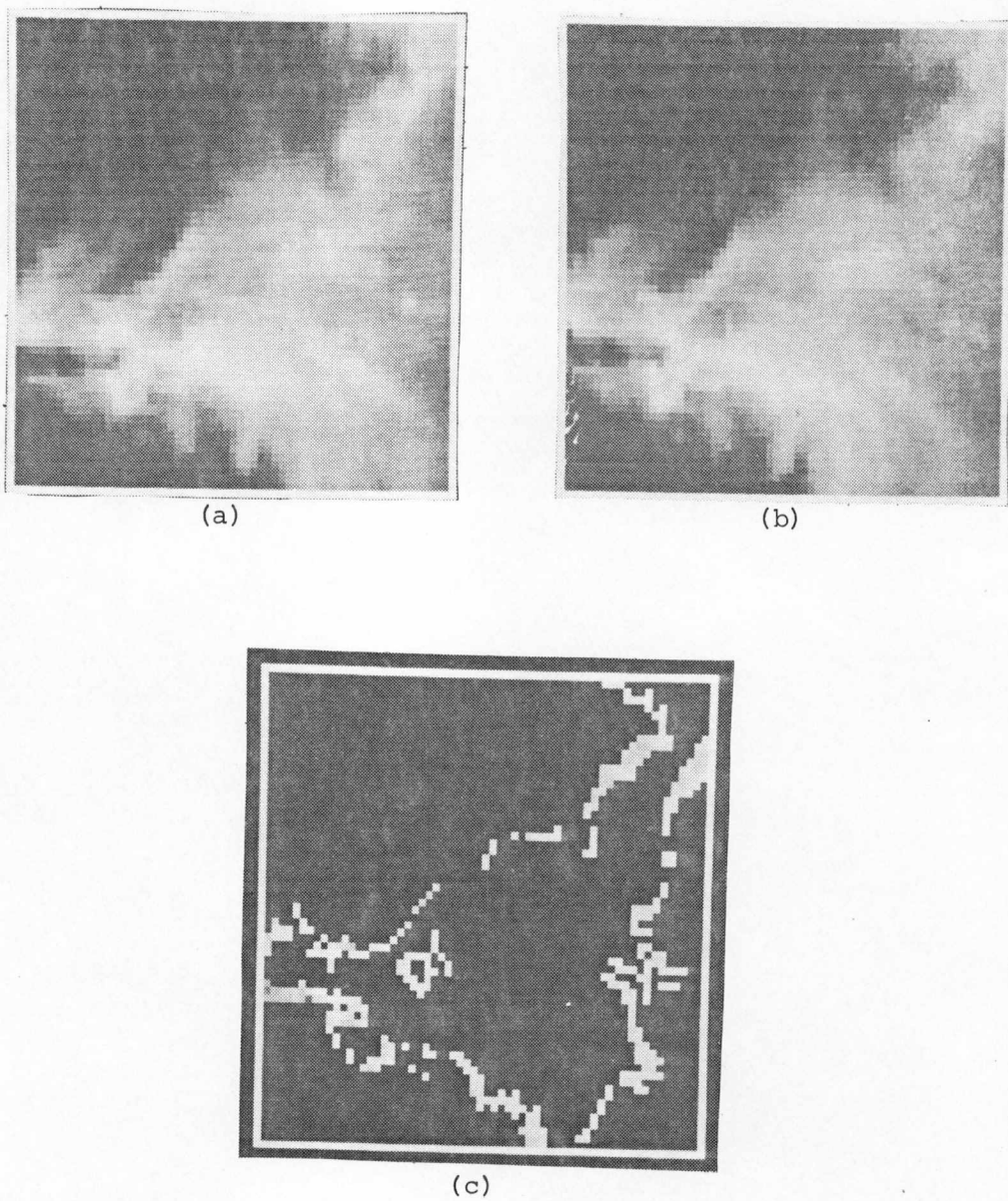
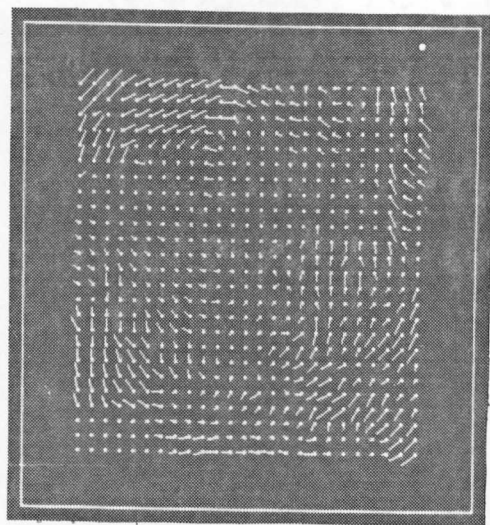
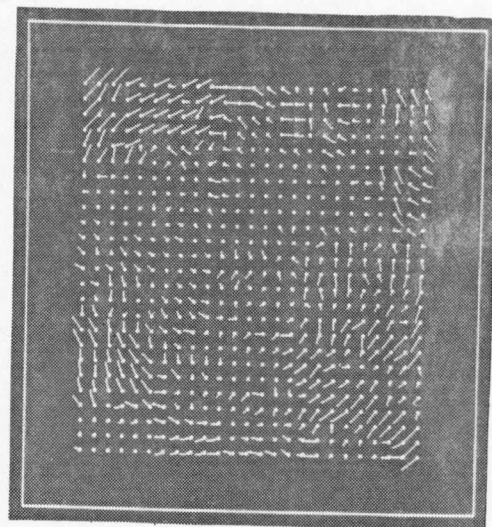


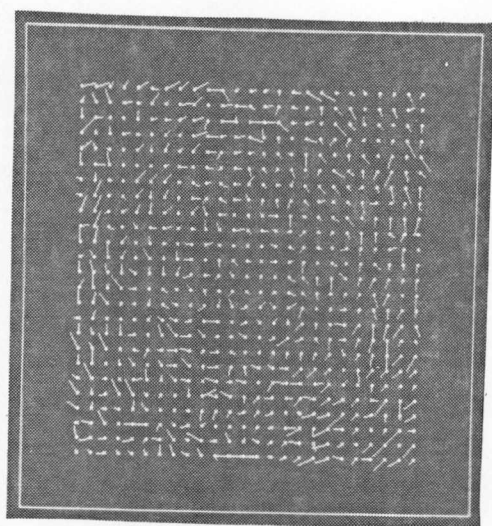
Figure 3.4-1. (a) State of a storm shortly before the occurrence of 3 tornadoes, (b) A computerized rotation simulation from (a), (c) Difference-image from (a) and (b).



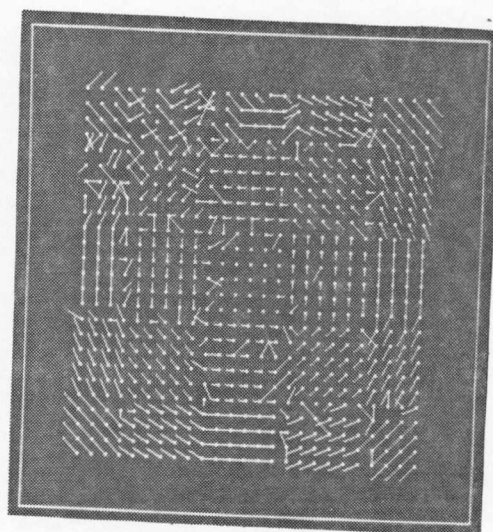
(a)



(b)



(c)



(d)

Figure 3.4-2. Circulation detection in radar images (a) 7 by 7 Operator, (b) 5 by 5 Operator, (c) 3 by 3 Operator, and 2 by 2 Operator.

5, 3 by 3, and 2 by 2 operators. As expected, the 7 by 7 and 5 by 5 operators described the motion with reasonable precision. The results of the 3 by 3 and 2 by 2 operators are correct up to a certain search radius. As in the case discussed above, the larger the operator size is, the wider the region of successful tracking. The 7 by 7 and 5 by 5 operators are time consuming procedures, but the precision of the results drawn from these tests have been excellent. The 2 by 2 and 3 by 3 operators are fast, but their success is limited to the region near the center of rotation. These results indicate that the tracking algorithm has great potential for detecting circulation in radar data. The 5 by 5 operator is optimal in the sense of accuracy and computation efficiency.

CHAPTER 4

SHAPE ANALYSIS AND CORRELATION IN CIRCULATION DETECTION

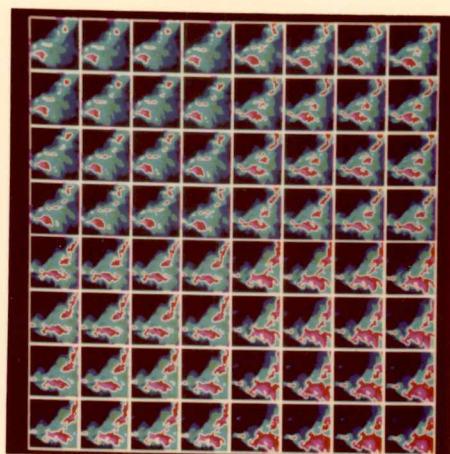
4.1. Shape Extraction and Matching

The concepts presented in Chapters 2 and 3 regarding shape analysis and correlation were applied to iso-reflectivity contours extracted from image radar data. From a single cloud image, a family of contours is extracted and thinned using the techniques mentioned in Section 1.3. Each contour is then coded and described by its Fourier Descriptors and matched against the Fourier Descriptors of each contour in a library prepared for this purpose. In this section the technique summarized in Fig. 1.3-1 (pp. 14) will be illustrated using a real case tornadic storm.

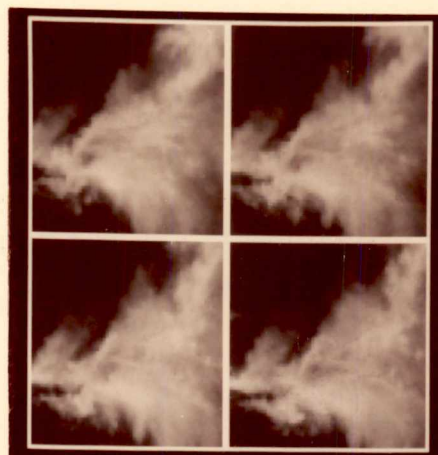
On April 10, 1979, between 5 pm and 6 pm, the northeast part of Texas and the southwest part of Oklahoma were struck by a severe thunderstorm which turned out to be highly tornadic. The storm intensified significantly around the area of Wichita Falls, TX. The data collected by NOAA was generated by a WSR-57 radar; the delay in azimuth was of 1 degree (1°), and the lag was 30 seconds. The figures in this section show 64 successive radar scans of the area of interest. These pictures read as follows: the first quadrant (top left) shows 16 images which read chronologically row-by-row from left to right. Similarly, the second quadrant (top right), the third quadrant (bottom left), and the fourth quadrant (bottom right) show the remainder of the sequence. The

intensities displayed are consistent with the conventional VIP display (0 to 6) going from clear to weak, to moderate, strong, very strong, intense, and then finally to extreme. The color codes are in the same order and go increasingly from black to blue, green, cyan, red, magenta, and finally to yellow. Figure 4.1-2 shows 4 detailed monochrome scans of the area during the tornado formation. All the individual pictures presented in this section are 64 by 64 pixels; the gray level varies from 0 to 63.

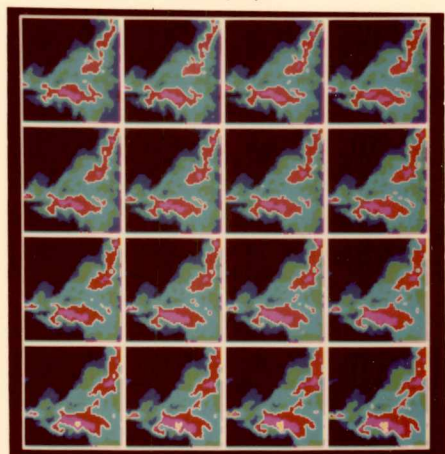
Following preprocessing, the sequence is checked for the existence of characteristic shapes, the appearance of which is directly related to tornadic activities. Figure 4.1-3 shows a 16-image sequence which exhibits a steady hook-like echo. This hook-like structure shows up better in Fig. 4.1-4 through 4.1-8 where the displays were restricted to a more confined area and the intensity was reduced in range. Singular shapes similar to the one shown in Fig. 4.1-8 are extracted from each scan image, coded, and then stored. Figure 4.1-6 shows that the hook-like structure was not casual. In fact each picture of the scan exhibits a hook-like structure of a certain maturity. The sequence was chosen to illustrate the birth of the hook (quadrants 1 and 2), its maturity (quadrant 3), and its dissipation (quadrant 4). Each of these shapes is then matched against all the prototypes contained in the shape library. Examples from this library are shown in Fig. 4.1-9(a) through 4.1-9(c) after preprocessing, segmentation, and thinning. The existence of a hook-like shape was confirmed in all the images of quadrant 3 [Fig. 4.1-3], giving a good match and, more importantly, indicating a constancy and repetitiveness in location and size of the shape.



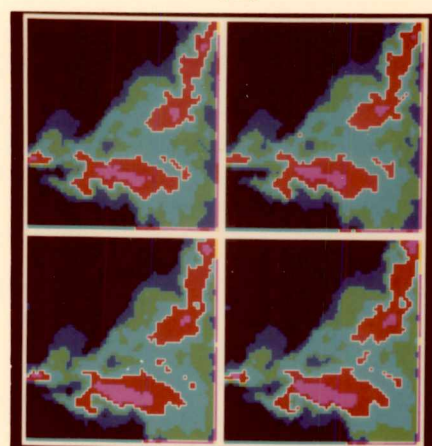
(1)



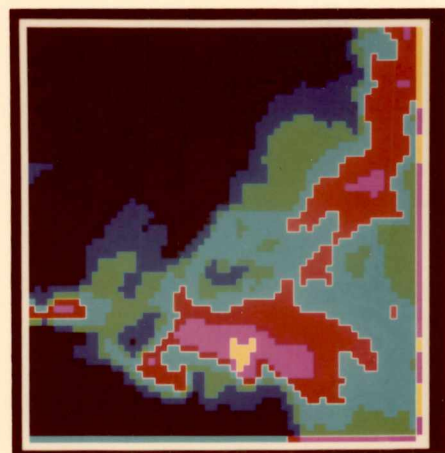
(2)



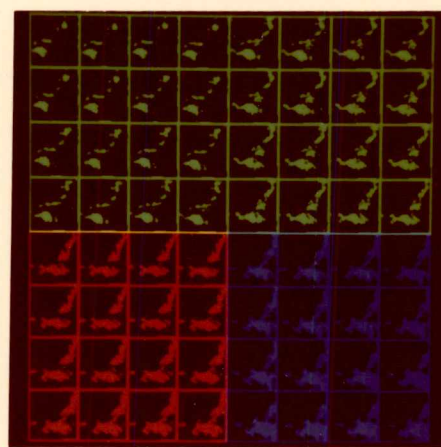
(3)



(4)



(5)



(6)

Figure 4.1- (1) Sequence of coded Radar scans, (2) Monochrome reflectivity display from (1), (3) Hook-like shapes in mature stage, (4) Four-image sequence from (3), (5) Coded display showing mature Hook-like echo, (6) Display of the sequence shown in (1) for $VIP \geq 4$.

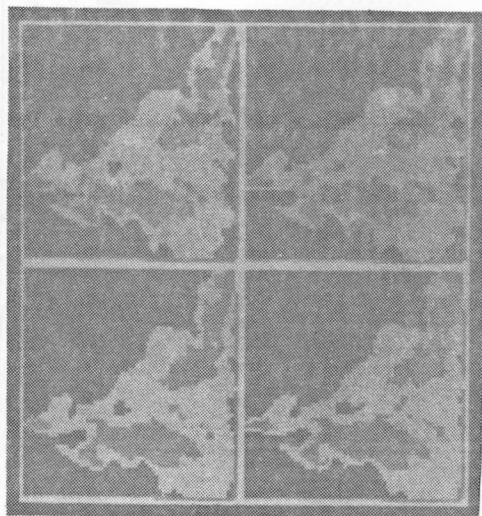


Figure 4.1-7. Hook contours extracted from Fig. 4.1-5 for 3 distinct VIP values.

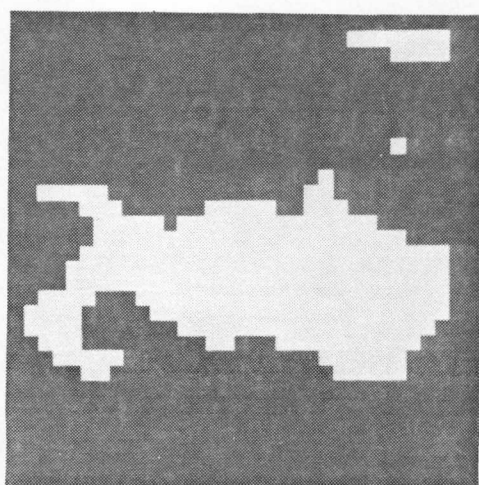
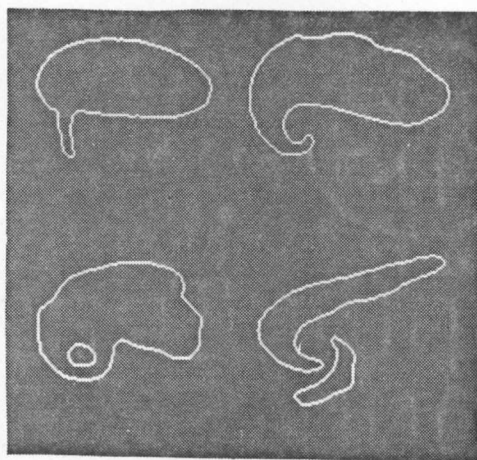
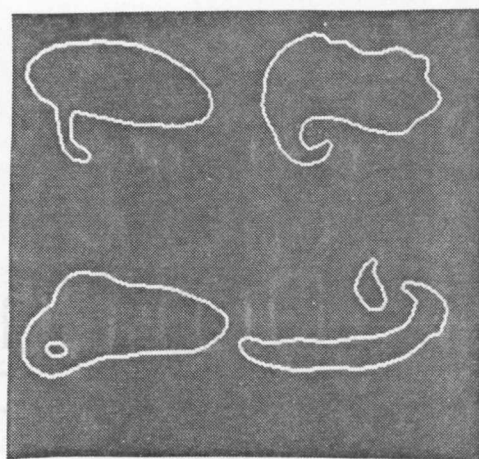


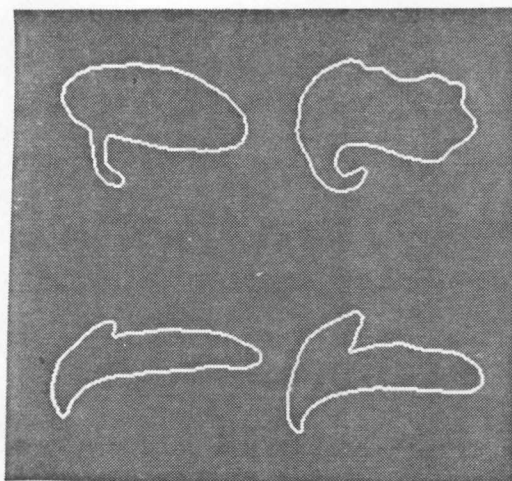
Figure 4.1-8. A single contour extracted from Fig. 4.1-5.



(a)



(b)



(c)

Figure 4.1-9 Examples of library prototype contours: (a) Type 1 of Pendant, Hook, Doughnut, and spiral; (b) Type 2 of Pendant, Hook, Doughnut, and Spiral; (c) Type 2 of Pendant and hook, Type 1 and 2 from Bird.

4.2. Air Flow by Correlation

As mentioned earlier in Chapter 1, once a characteristic shape is detected and confirmed, the hook area of the storm cell is declared tornadic and a search for circulation in it is started. In this case the search area was reduced to 31 by 15 pixels. Using the techniques discussed in Chapter 3, a TREC field is derived for this area. Shown in Figs. 4.2-1 and 4.2-2 are the results of correlation between the first and second and the second and third images of the sequence displayed in Fig. 4.1-3. By examining the top middle of the display, we see a tentative rotational motion. Moreover, the area clearly shows, that all the features of a tornadic pattern, such as the RFD (Rear Flank Downdraft), the UD (Up Draft), and the FFD (Front Forward Downdraft), are very similar to the features showed in Fig. 4.2-3, which represents a reliable tornado model [31]. The core of the tornado is situated in the middle top of Fig. 4.2-1; the lines of convergence and divergence in the TREC pattern confirm this conclusion. Once again, the repetitiveness of this pattern throughout the entire image sequence is a strong suggestion of the validity of the conclusions that the pattern detected is in fact a tornado pattern. Figure 4.2-4 shows a more detailed and complete view of the area of tornadic activities than Fig. 4.2-1. Interestingly, the area in the image which exhibits a tornadic pattern is the one identified by the prescan performed through shape analysis. The air flow map shown in Fig. 4.2-4 is shown again in Fig. 4.2-5, superimposed to the dominant

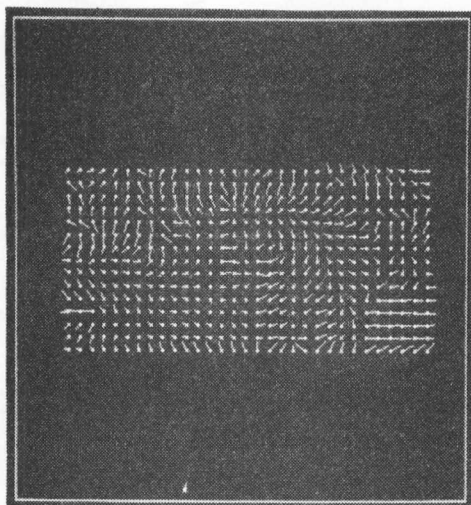


Figure 4.2-1. TREC Vectors of image #1 and #2 from Fig. 4.1-1.

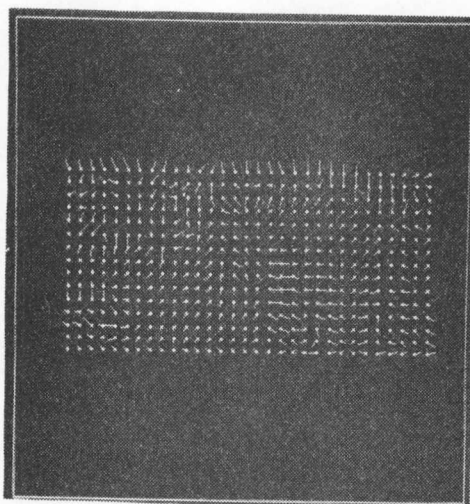


Figure 4.2-2. TREC Vectors of image #2 and #3 from Fig. 4.1-1.

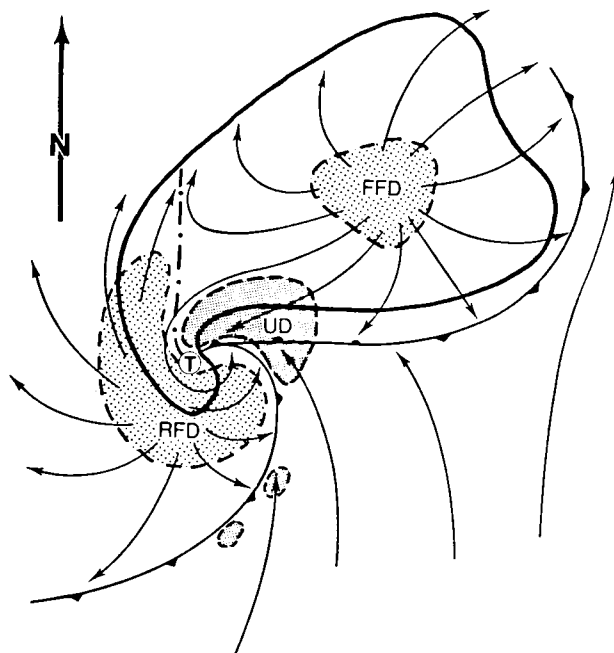


Figure 4.2-3. Air flow in a tornado model.

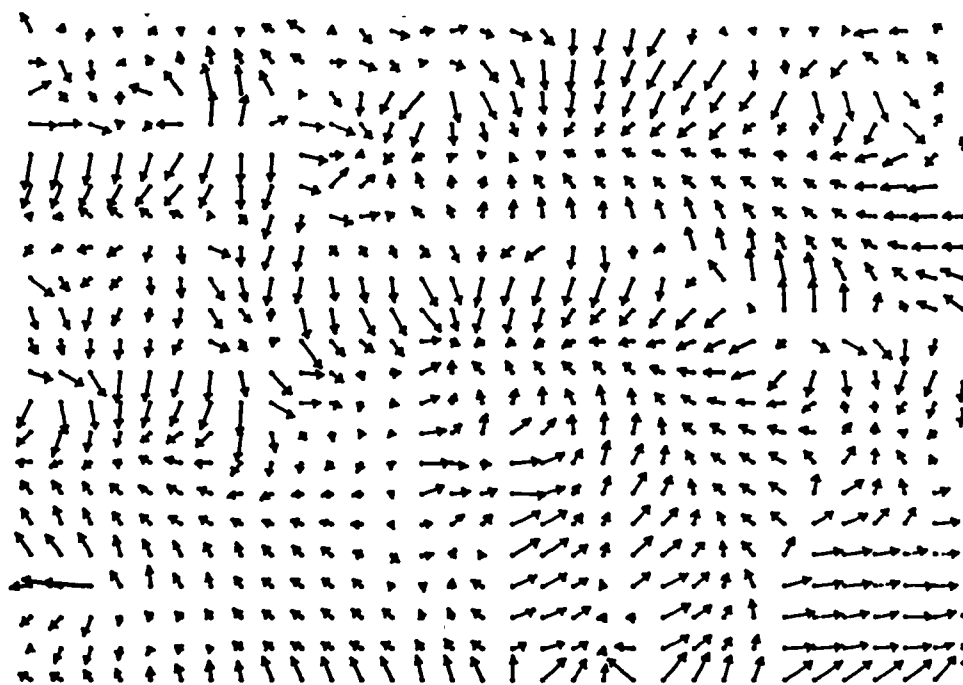


Figure 4.2-4. Typical Example of Air Flow Map derived from the case of April 10, 1979.

features of the tornado model shown in Fig. 4.2-3. This again stressed the fact that the results derived from this analysis conformed with what actually happened because of the high similarity between the pattern retrieved by the proposed method and the tornado air flow model.

4.3. Measurement of Circulation in Cloud Images

To further automate the procedure of tornado detection and location, a measure of circulation is used. This measure computes the circulation at each image point by adding the rotational effect of all the TREC vectors within a square centered around the element where the circulation is computed. Figure 4.3-1 gives the circulation measure at each point of the storm image where a TREC vector was computed, specifically around the hook shown in Fig. 4.1-8, or the selected area of Fig. 4.2-2. The two-dimensional function shown in Figure 4.3-1 achieves its maximum in the middle top of the display, exactly where the TREC pattern showed the tornado core.

The results presented in Chapter 4 illustrate the techniques and the processing steps required to automate tornado detection and location. In addition to displaying their theoretical validity, which was proved in earlier studies [12,13,14,21], these techniques also showed great potential for addressing motion problems in radar data.

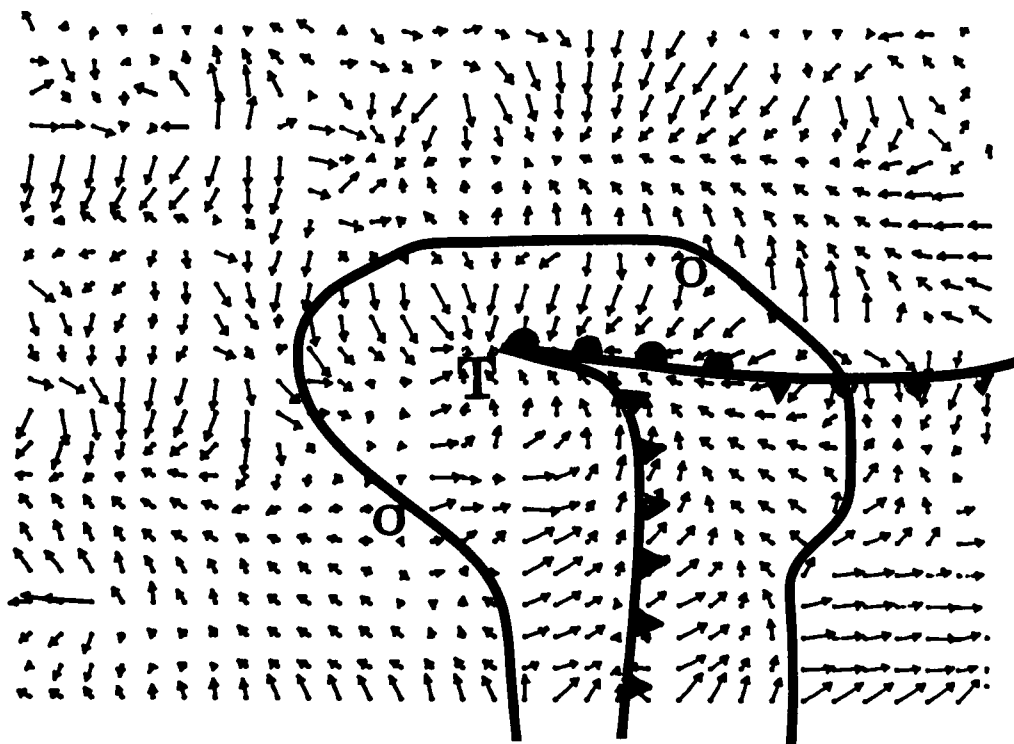


Figure 4.2-5. Air Flow Map superimposed to dominant tornado features.

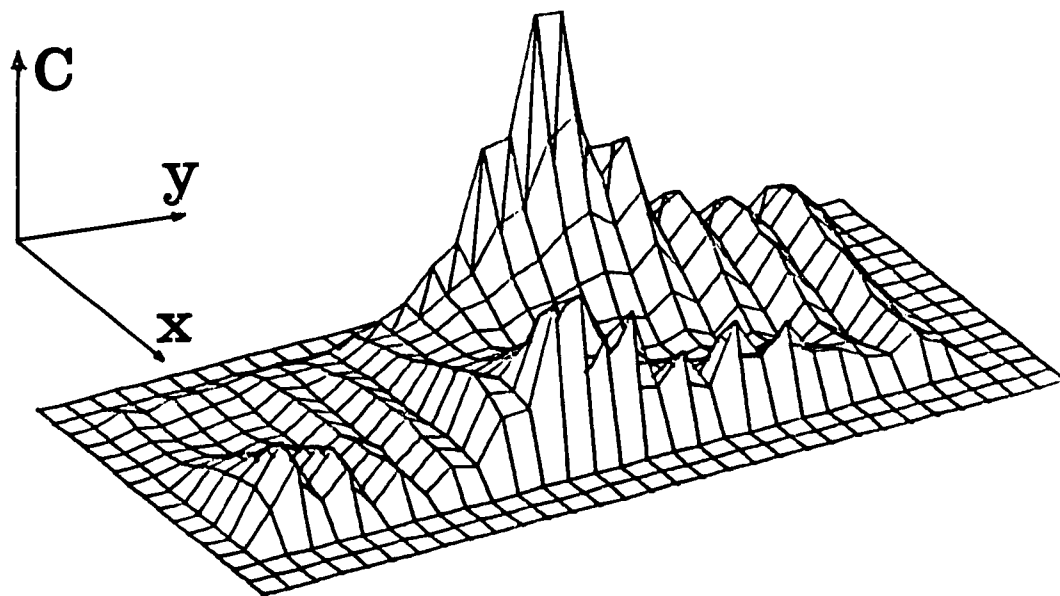


Figure 4.3-1. Circulation mapping from the data in Fig. 4.2-1.

CHAPTER 5

CONCLUSIONS

Throughout this work, we have introduced a hybrid technique capable of motion discrimination. A global description of this technique is illustrated in Figure 1.3-1 (pp. 14). It covers noise reduction, image segmentation and curve thinning, curve encoding, matching, and recognition, optical flow by correlation, and finally detection and location of image areas of potential circulation. In Chapter 1 we have shown the interconnections between the different subtasks of the recognition approach through an example. Chapter 2 was devoted to shape analysis introducing curve definitions and shape descriptors. The technique allowing the generation of air flow maps by area matching via correlation is detailed in Chapter 3 along with some demonstrative test experiments. An application of these techniques with a real-case tornadic image-sequence was presented in Chapter 4. The conclusions drawn based on this method were consistent with what actually happened. This hybrid approach based on shape detection and area correlation is a potential real-time cloud tracking technique because all algorithms used were tailored to be fast, effective, and fully compatible with each other.

LIST OF REFERENCES

LIST OF REFERENCES

1. Brooks, H. B., "A Summary of Some Radar Thunderstorm Observations," Bull. Amer. Meter. Soc., Vol. 27, pp. 557-563, 1946.
2. Byers, H. R. and R. R. Braham, Jr., The Thunderstorm, U.S. Gov't Printing Office, Washington, D.C., 1949.
3. Hilst, G. R. and J. A. Russo, Jr., "An Objective Extrapolation Technique for Semi-Conservative Fields with an Application to Radar Patterns," Tec. Memo. No. 3., Contract AF 30-635-14459, Travelers Wea. Res. Center, Hartford, Conn., 1960.
4. Kesler, Edwin III and J. A. Russo, Jr., "A Program for the Assembly and Display of Radar-Echo Distributions," J. Appl. Meter., vol. 5, pp. 582-593, 1963.
5. Blackmer, R. H., R. O. Duda, and R. Rebah, "Application of Pattern Recognition Techniques for Digitized Weather Radar Data," Final report, Contract 1-36072, Stanford Res. Inst., Menlo Park, CA, 1973.
6. Austin, G. L. and A. Bellon, "The Use of Digital Weather Radar Records for Short-Term Precipitation Forecasting," Quart. J. Roy. Meter. Soc., vol. 100, pp. 658-664, 1974.
7. Zawadzki, I. I., "Statistical Properties of Precipitation Patterns," J. Appl. Meter. vol. 12, pp. 459-472, 1973.
8. Ostlund, S. S., "Computer Software for Rainfall Analysis and Echo Tracking of Digitized Radar Data," NOAA Tech. Memo. ERL WNP-15, U.S. Dept. of Commerce, NOAA Environ. Research Lab., 1974.
9. Wiggert, V., S. S. Ostlund, G. Lockett, and J. Stewart, "Computer Software for the Assessment of Growth Histories of Weather Radar Echoes," NOAA Tech. Memo. ERL WMP-35, U.S. Dept. of Commerce, NOAA Environ. Research Lab., 1976.
10. Alaka, M. A., J. P. Charba, and R. C. Elvander, "Thunderstorm Prediction for Use in Air Traffic Control (0-6 Hours Time Range)," Report No. FAA-RD-77-40, NOAA, Silver Spring, MD 1977.
11. Rinehart, R. E., "Internal Storm Motions from a Single Non-Doppler Weather Radar," Technical Note NCAR/TN-146-STR, Nat. Center for Atm. Res., Boulder, CO, 1979.
12. Smythe, G. R., "Correction of Doppler-Radar Velocities with Application to Retrieving the Transverse Wind," Report No. 4,

- CIMMS, Univ. of Oklahoma-Nat. Oceanic and Atm. Adm., Norman, Oklahoma 1981.
13. Yachida M. M., Asada and S. Tsuji, "Automatic Analysis of Moving Images," IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-3, No. 1, pp. 12-20, January 1981.
 14. R. E. Bogner, "Pattern Recognition via Observation Correlations," IEEE Trans. pattern Anal. Machine Intell., vol. PAMI-2, No. 2, pp. 128-133, march 1981.
 15. Sclove S. L., "Pattern Recognition in Image Processing Using Interpixel Correlations," IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-3, No. 2, pp. 206-208, March 1981.
 16. Chen P. C. and T. Pavildis, "Segmentation by Texture Using Correlation," IEEE trans. Pattern Anal. Machine Intell., vol. PAMI-5, No. 1, January 1983.
 17. Abidi M. A. and R. C. Gonzalez, "Application of Pattern Recognition Techniques to Within-Storm Motion Detection," Final Report to NOAA, Contract NA-SAC-00072, June 1983.
 18. Gonzalez, R. C. and P. Wintz, Digital Image Processing, Addison Wesley, Reading, MA, 1977.
 19. Hall, E. L., Computer Image Processing and Recognition, Academic Press, New York, 1979.
 20. Elliot H., D. B. Cooper, F. S. Cohen , and P. F. Symosek, "Implementation, Interpretation, and Analysis of a Suboptimal Boundary Finding Algorithm," IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-4, No. 2, March 1982.
 21. Pavlidis T., "A Review of Algorithms for Shape Analysis," Compt. Graphics Image Processing, vol. 7, pp. 243-258, 1978.
 22. Zahn C. T., "A Review of Algorithms for Shape Analysis," Compt. Graphics Image Processing, vol. C-21, pp. 269-281, March 1972.
 23. Persoon E. and K. S. Fu, "Shape Discrimination Using Fourier Descriptors," IEEE Trans. Syst., Man, Cybern., vol. SMC-7, pp. 170-179, March 1977.
 24. Meiri A. Z., "On Monocular Perception of 3-D Moving Objects," IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-2, No. 6, November 1980.
 25. Dudani S. A. et al., "Aircraft Identification by Moment Invariants," IEEE Trans. Comput., vol. C-26, pp. 39-46, January 1977.

26. Hu M. K., "Visual Pattern Recognition by Moment Invariants," IEEE Trans. Inform. Theory, vol. IT-8, pp. 179-187, February 1962.
27. Rosenfeld A., "Digital Straight Line Segments," IEEE Trans. Comput., vol. C-23, pp. 1264-1269, December 1974.
28. Wallace T. P., O. R. Mitchell and K. Fukunaga, "Three-Dimensional Shape Analysis Using Local Shape Descriptors," IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-3, No. 3, May 1983.
29. Wallace T. P., O. R. Mitchell and K. Fukunaga, "Three-dimensional Shape Analysis Using Local Descriptors," in Proc. IEEE Pattern Recog. Image Processing Conf., August 1979.
30. Abidi, M. A. and R. C. Gonzalez, "Application of Pattern Recognition Techniques to Within-Storm Motion Detection," Interim Report, NOAA, Contract NA-SAC-00072, October 1982
31. Grebe R., An Outline of Severe Local Storms with Morphology of Associated Radar Echoes, NOAA Technical memorandum NWS TC echoes, June 82.
32. Fujita, T. T., "Proposed Mechanism of Tornado Formation from Rotating Thunderstorms," Preprints, 8th Severe Local Storms Conf., Amer. Meteor. Soc., Boston, 1973, pp. 191-196.
33. Stentiford, F. W. M and R. G. Mortiner, "Some New Heuristics for Thinning Binary Handprinted Characters for OCR," IEEE Trans. Systems, Man, and Cyber., Vol. SMC-13, No. 1, pp. 81-81, 1983.
34. Davis, E. R. and A. P. N. Plummer, "Thinning Algorithms: A Unique and a New Methodology," Pattern Recognition, Vol. 14, No. 1-6, pp. 53-63, 1981.
35. Freeman, H., "Computer Processing of Line Drawing Images", Comput. Surv., No. 6, pp. 59-97, 1974.
36. Abidi, M. A. and R. C. Gonzalez, "Motion Detection in Radar Images," Proc. Int. Conf. of Pattern Recognition, Montreal, Canada, July 1984.
37. Tou, J. T. and R. C. Gonzalez, Pattern Recognition Principles, Addison-Wesley, Reading, Mass., 1974.
38. Goshtasby A., S. H. Gage and J. F. Bartholic, "A Two-Stage Cross Correlation Approach to Template Matching," Trans IEEE Pattern Anal. Mach. Intell. Vol. PAMI-6, No. 3, pp. 374-378, May 1984.

39. Zahn C. T. and R. Z. Roskies, "Fourier Descriptors for Closed Curves," IEEE Trans. Comput., Vol. C-21, pp. 269-281, 1972.
40. Richard C. W. and H. Hamani, "Identification of Objects using Fourier Descriptors of the Boundary Curve," IEEE Trans. Systems, Man, Cybern., Vol. SMC-4, pp. 371-378, July 1974.
41. Granlund G. H., "Fourier Preprocessing for Hand Print Character Recognition," IEEE Trans. Comput., Vol. C-21, pp. 195-201, 1972.
42. Wallace, T. P. and O. R. Mitchell, "Analysis of Three-Dimensional Movement Using Fourier Descriptors," IEEE Trans. Pattern. Anal. Mach. Intell., Vol. PAMI-2, No. 6, pp. 583-588, November 1980.
43. Schalkoff R. J. and E. S. McVey, "A Model and Tracking Algorithm for a Class of Video Targets," IEEE Trans. Pattern Anal. Mach. Intell. vol. PAMI-4, No. 1, pp. 2-10, January 1982.
44. Frost V. S., J. A. Stiles K. S. Shanmugan and J. C. Holtzman, "A Model for Radar Images and its Application to Adaptive Digital Filtering of Multiplicative Noise," IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-4, No. 2, March 1982.

APPENDIX A
CORRELATION

```

C
C      NAME: RFCOR.FOR
C
C      This is the main program for image Correlation .
C      In ALL programs :
C          SP1 is the original image
C          SP2 is the second image
C
C      CHARACTER*36 NAME1,NAME2,NAME3
C      LOGICAL*1 TEM(64,64),DUM(64*64),TYPE,SIZE,TITLE(8)
C      INTEGER SP1(64,64)
C      INTEGER SP2(64,64)
C      INTEGER IMM1(16,16),IMM2(16,16),IO(10,10),JO(10,10)
C      INTEGER LI(10,10),LJ(10,10),PH(10,10)
C      REAL V(10,10),X1(10,10),X2(10,10),T(10,10),RM(10,10)
C      INTEGER P,CI1,CJ1,CI2,CJ2
C      INTEGER ISTBLK,NDBLK,MONTH,DAY,YEAR,FLAG
C      REAL XIOP , YJOP
C      REAL VO,RMAX
C      EQUIVALENCE (TEM(1,1),DUM(1))
C
C      Enter initial Picture
C
C      CALL GETFNM(' ENTER FILE 1 : ','PIC',NAME1,NCHAR)
C      OPEN(UNIT=1,NAME=NAME1,TYPE='OLD',ACCESS='DIRECT',
2     FORM='UNFORMATTED',RECORDSIZE=128)
C      CALL PICRD(1,DUM,1,1)
C      CALL RDHDR(TEM,TYPE,SIZE,ISTBLK,NDBLK,TITLE,MONTH,
2     DAY,YEAR,FLAG)
C      CALL PICRD(1,DUM,ISTBLK,8)
C
C
C      DO 1 I=1,64
C      DO 1 J=1,64
C      SP1(I,J)=TEM(J,I)
C      IF(SP1(I,J).LT.0) SP1(I,J)=256+SP1(I,J)
1     CONTINUE
C
C      Enter Second Picture
C
C      CALL GETFNM(' ENTER FILE 2 : ','PIC',NAME2,NCHAR)
C      OPEN(UNIT=2,NAME=NAME2,TYPE='OLD',ACCESS='DIRECT',
2     FORM='UNFORMATTED',RECORDSIZE=128)
C      CALL PICRD(2,TEM,1,1)
C      CALL RDHDR(TEM,TYPE,SIZE,ISTBLK,NDBLK,TITLE,MONTH,
2     DAY,YEAR,FLAG)
C      CALL PICRD(2,TEM,ISTBLK,8)
C      DO 2 I=1,64
C      DO 2 J=1,64
C      SP2(I,J)=TEM(J,I)
C      IF(SP2(I,J).LT.0) SP2(I,J)=256+SP2(I,J)
2     CONTINUE
C
C      CLOSE(UNIT=1)

```

```

      CLOSE(UNIT=2)
C
C      Output File
C
      CALL GETFNM(' Enter output file : ','DAT',NAME3,NC)
      OPEN(UNIT=1,NAME=NAME3,TYPE='NEW',ACCESS='SEQUENTIAL
2    ',FORM='FORMATTED')
C
C      Main Parameters input
C
      TYPE 700
700    FORMAT(' Enter length of the Unit Cell (Odd) : ',,$)
      ACCEPT *, IUC
      TYPE 701
701    FORMAT(' Enter length of the Search Area(Odd): ',,$)
      ACCEPT *, ISA
      TYPE 800
800    FORMAT(' Enter Istart , Jstart : ',,$,$)
      ACCEPT *, ISTART , JSTART
      TYPE 801
801    FORMAT(' Enter Iend , Jend : ',,$,$)
      ACCEPT *, IEND , JEND
      TYPE 900
900    FORMAT(' Enter Istep , Jstep : ',,$,$)
      ACCEPT *, ISTEP , JSTEP
      TYPE 902
902    FORMAT(' Enter RTH (0.6 - 1.0) and VIPTH : ',,$,$)
      ACCEPT *, RTH , VIPTH
C
C
10    CONTINUE
C
C      # OF TESTS STORED
C
      N1=1+(IEND-ISTART)/ISTEP
      N2=1+(JEND-JSTART)/JSTEP
C
C      CORRELATION OF CELLS
C
      ICOUNT=0
C
      NBMES=0
      DO 11 I=ISTART,IEND,ISTEP
      DO 11 J=JSTART,JEND,JSTEP
      IF(SP1(I,J).GE.VIPTH) NBMES=NBMES+1
11    CONTINUE
      WRITE(1,100) NBMES
100   FORMAT(4X,I6)
      DO 3 I=ISTART , IEND , ISTEP
      DO 3 J=JSTART , JEND , JSTEP
      CI1=I
      CJ1=J
      CI2=I
      CJ2=J
C

```

```

      IF(SP1(I,J).LT.VIPTH) GO TO 3
C
      IF(IUC.EQ.2) GO TO 180
C
      XIOP=FLOAT(I)
      YJOP=FLOAT(J)
C
C      Main Correlation CALL for 3 by 3 , 5 by 5 , 7 by 7 ,
C      ... Operators
C
      CALL COR(SP1,64,SP2,64,CI1,CJ1,IUC,CI2,CJ2,ISA,V0,
2      P,IMM1,IMM2,X1,X2,T,XIOP,YJOP,RMAX,RTH,ICOUNT)
C
      GO TO 190
180    CONTINUE
C
C      Main correlation CALL for 2 by 2 Operator
C
      CALL COR2(SP1,SP2,ISA,I,J,IOP,JOP)
      XIOP=FLOAT(IOP)
      YJOP=FLOAT(JOP)
190    CONTINUE
      XX=XIOP
      YY=YJOP
      WRITE(1,200) I , J , XX , YY
200    FORMAT(4X,2(I6),3X,2(F8.3))
C
C
3      CONTINUE
      CLOSE(UNIT=1)
C
      STOP
      END
C

```

```

C      NAME:   RMATOP.FOR
C
C      SUBROUTINES used for Correlation
C
C*****C
C      SUBROUTINE CALPHI(I,J,IOP,JOP,IP)
C*****C
C
C      This SUBROUTINE calculates the angle P
C      in degrees ( 0 to 359 ) between the
C      horizontal and the vector joining
C      A(I,J) and AOP(IOP,JOP)
C
C*****C
C
C      IF(J.EQ.JOP) GO TO 1
C      X=(FLOAT(I-IOP))/(FLOAT(J-JOP))
C      PP=(180.0/3.14159)*ATAN(X)
C      IP=ABS(PP)
C      IF(IOP.LT.I.AND.JOP.LT.J) IP=180-IP
C      IF(IOP.LT.I.AND.JOP.GT.J) GO TO 2
C      IF(IOP.GT.I.AND.JOP.LT.J) IP=180+IP
C      IF(IOP.GT.I.AND.JOP.GT.J) IP=360-IP
C      IF(IOP.LT.I.AND.JOP.EQ.J) IP=90
C      IF(IOP.GT.I.AND.JOP.EQ.J) IP=270
C      IF(IOP.EQ.I.AND.JOP.GT.J) IP=0
C      IF(IOP.EQ.I.AND.JOP.LT.J) IP=180
C      IF(IOP.EQ.I.AND.JOP.EQ.J) IP=0
C      2  CONTINUE
C      RETURN
C      END
C
C
C*****C
C      SUBROUTINE COR(IM1,M1,IM2,M2,CI1,CJ1,N1,CI2,CJ2,N2,
C      1  V0,IP,IMM1,IMM2,X1,X2,T,XIOP,YJOP,RMAX,RTH,ICOUNT)
C*****C
C
C      This SUBROUTINE corrolates a region of dimension N1
C      having a centre at CI1,CJ1 from the Matrix IM1( M1
C      by M1) with all the sub-regions of the search area
C      of dimension N2 centred at CI2,CJ2 from the Matrix
C      IM2(M2 by M2). The result is given as : the velocity
C      V0 which the module of the vector separating the 2
C      centers of the mached area . The angle PHI0 given in
C      degrees ( 0 to 359 ) measures the deviation of the
C      velocity vector with respect to the horizontal line.
C
C*****C

```

```

C
C
      INTEGER IM1(M1,M1) , IM2(M2,M2) , IMM1(N1,N1)
      INTEGER IMM2(N2,N2)
      REAL X1(N1,N1) , X2(N1,N1) , T(N1,N1)
      REAL MAX(200) , XI(200) , YJ(200)
      INTEGER CI1,CJ1,CI2,CJ2,N1,N2,IP
      REAL S1,S2,S11,S12,S22

```

```

C
C
C      TESTS ON THE COMPATIBILITIES
C

```

```

      IF(N1.GT.M1.OR.N2.GT.M2.OR.N1.GT.N2) GO TO 6
      NN1=(N1-1)/2
      NN2=(N2-1)/2

```

```

C
C      Set-up for Correlation
C

```

```

      KI1=CI1-NN1
      KI2=CI1+NN1
      KJ1=CJ1-NN1
      KJ2=CJ1+NN1
      DO 1 I=KI1,KI2
      DO 1 J=KJ1,KJ2
      LI=I-CI1+NN1+1
      LJ=J-CJ1+NN1+1
      IMM1(LI,LJ)=IM1(I,J)
1      CONTINUE

```

```

C
C
      KI1=CI2-NN2
      KI2=CI2+NN2
      KJ1=CJ2-NN2
      KJ2=CJ2+NN2
      DO 2 I=KI1,KI2
      DO 2 J=KJ1,KJ2
      LI=I-CI2+NN2+1
      LJ=J-CJ2+NN2+1
      IMM2(LI,LJ)=IM2(I,J)
2      CONTINUE

```

```

C
C
      NB=0
      NIJ=N2-N1+1
      DO 22 K=1,N1
      DO 22 L=1,N1
      X1(K,L)=FLOAT(IMM1(K,L))
22      CONTINUE
      CALL PEE(N1,N1,X1,X1,T)
      CALL SUMM(N1,N1,T,S11)
      CALL SUMM(N1,N1,X1,S1)
      DO 5 I=1,NIJ
      DO 5 J=1,NIJ

```

```

C
C

```

```

DO 3 K=1,N1
DO 3 L=1,N1
KI=K+I-1
LJ=L+J-1
X2(K,L)=FLOAT(IMM2(KI,LJ))
3    CONTINUE
C
C
CALL PEE(N1,N1,X1,X2,T)
CALL SUMM(N1,N1,T,S12)
CALL SUMM(N1,N1,X2,S2)
CALL PEE(N1,N1,X2,X2,T)
CALL SUMM(N1,N1,T,S22)
A=S12-((S1*S2)/(N1**2))
B=SQRT((S11-((S1**2)/(N1**2)))*(S22-((S2**2)/(N1**2)
2  ))) + 0.0001
R=A/B
C
C    R is the cross-correlation coefficient
C
IOP=CI2+I-(N2-N1)/2-1
JOP=CJ2+J-(N2-N1)/2-1
C
C
IF(R.LT.RTH) GO TO 5
NB=NB+1
MAX(NB)=R
XI(NB)=FLOAT(IOP)
YJ(NB)=FLOAT(JOP)
C
C
5    CONTINUE
C
C
XIOP=0.0
YJOP=0.0
RTOT=0.0
C
C    storage of R values for all search area elements
C
DO 505 KKK=1,NB
XIOP=XIOP+XI(KKK)*MAX(KKK)
YJOP=YJOP+YJ(KKK)*MAX(KKK)
RTOT=RTOT+MAX(KKK)
505  CONTINUE
C
IF(NB.EQ.0) GO TO 506
C
XIOP=XIOP/RTOT
YJOP=YJOP/RTOT
RMAX=RTOT/FLOAT(NB)
GO TO 508
C

```

```

506    CONTINUE
C
      ICOUNT=ICOUNT+1
      WRITE(5,507) ICOUNT
507    FORMAT(' There is no optimal for this point ,#',I5)
C
C
C
      XIOP=FLOAT(CI1)
      YJOP=FLOAT(CJ1)
508    CONTINUE
C
C    Calculation of Magnitude and Phase .
C
      CALL CALPHI(CI1,CJ1,IOP,JOP,IP)
      VO=SQRT(FLOAT((CI1-IOP)**2+(CJ1-JOP)**2))
6      CONTINUE
C
C
      RETURN
      END
C
C
C*****C
      SUBROUTINE SUMM(M,N,M1,SUM)
C*****C
C
C    This SUBROUTINE calculates the sum of the
C    elements of a matrix M1 (M by N) .
C
C
C
      REAL    M1(M,N)
      REAL SUM
      SUM=0.0
      DO 1 I=1,M
      DO 1 J=1,N
          SUM=SUM+M1(I,J)
1      CONTINUE
      RETURN
      END
C
C
C*****C
      SUBROUTINE PEE(M,N,M1,M2,M12)
C*****C
C
C    THIS SUBROUTINE CALCULATES THE MATRIX PRODUCT
C    ELEMENT BY ELEMENT M12(M BY N) OF 2 MATRICES
C    M1(M BY N) AND M2(M BY N)
C
C
C
      REAL M1(M,N)
      REAL M2(M,N)
      REAL M12(M,N)
C

```

```

      DO 1 I=1,M
      DO 1 J=1,N
      M12(I,J)=(M1(I,J))*(M2(I,J))
1     CONTINUE
      RETURN
      END

C
C
C
C*****C
      SUBROUTINE MAX(M,N,M1,XMAX)
C*****C
C
C      THIS SUBROUTINE CALCULATES THE MAXIMUM OF THE
C      THE ELEMENTS OF A MATRIX M1(M BY N)
C
C
      REAL M1(M,N)
      XMAX=M1(1,1)
      DO 1 I=1,M
      DO 1 J=1,N
          IF(XMAX.LT.M1(I,J)) XMAX=M1(I,J)
1     CONTINUE
      RETURN
      END

C
C
C*****C
      SUBROUTINE COR2(SP1,SP2,ISA,I,J,IOP,JOP)
C*****C
C
C      This program search for the new position of a 2 by 2
C      matrix from SP2 which is the best match for a 2 by 2
C      matrix from SP1 . ISA is the dimension of the search
C      area . (I,J) gives (IOP,JOP) .
C
C
      INTEGER SP1(64,64) , SP2(64,64)
C
C
      A1=SP1(I,J)
      B1=SP1(I,J+1)
      C1=SP1(I+1,J)
      D1=SP1(I+1,J+1)
C
      I1=I-ISA/2
      I2=I+ISA/2
      J1=J-ISA/2
      J2=J+ISA/2
C
      RMAX=-2
      DO 3 L=I1,I2
      DO 3 M=J1,J2

```

```

A2=SP2(L,M)
B2=SP2(L,M+1)
C2=SP2(L+1,M)
D2=SP2(L+1,M+1)

C
C
S1=A1+B1+C1+D1
S2=A2+B2+C2+D2
S11=A1**2+B1**2+C1**2+D1**2
S12=A1*A2+B1*B2+C1*C2+D1*D2
S22=A2**2+B2**2+C2**2+D2**2

C
C
A=S12-((S1*S2)/4.0)
B=SQRT((S11-((S1**2)/4.0))*(S22-((S2**2)/4.)))+0.0001

C
C
R is the value of the cross-correlation
C

R=A/B
IF(R.GE.RMAX) GO TO 1
GO TO 2
1  RMAX=R
   IOP=L
   JOP=M
2  CONTINUE
3  CONTINUE
C
C
RETURN
END

```

APPENDIX B

SHAPE

```

C      NAME:  KF1.FOR
C
C      This program performs
C
C                      Filtering,
C                      Segmentation
C                      Thinning
C                      Lines Filtering
C                      Lines Coding
C
C                      for a first  image
C      to be SEGMENT-MATCHED against a second one .
C
C
C      LOGICAL*1 SP1(128,128) , HSP1(512) , NAME1(36)
C      LOGICAL*1 NAME2(36)
C      LOGICAL*1 SP11(5,126) , IOB(4000) , JOB(4000)
C
C
C
111    TYPE 1000
1000   FORMAT('/', ' Enter 1 to continue : ', $)
        ACCEPT *, K1
        IF(K1.NE.1) GO TO 2
        CALL GETFNM(' Enter input file : ', 'PIC', NAME1, NCH)
        OPEN(UNIT=1, NAME=NAME1, TYPE='OLD', ACCESS='DIRECT',
1       FORM='UNFORMATTED', RECORDSIZE=128)
        CALL PICRD(1, SP1, 1, 1)
        CALL RDHDR(SP1, , IDSTBK)
        CALL PICRD(1, SP1, IDSTBK, 32)
C
C
C
        TYPE 1100
1100   FORMAT(' Enter 1 for thresholding : ', $)
        ACCEPT *, KK
        IF(KK.EQ.1) CALL STH1(SP1)
        TYPE 1200
1200   FORMAT(' ENTER 1 FOR BINARY GRADIENT : ', $)
        ACCEPT *, KK
        IF(KK.EQ.1) CALL SBG1(SP1)
        TYPE 1500
1500   FORMAT(' Enter 1 for averaging : ', $)
        ACCEPT *, KK
        IF(KK.EQ.1) CALL SAV1(SP1, SP11)
        TYPE 2000
2000   FORMAT(' Enter 1 for slicing : ', $)
        ACCEPT *, KK
        IF(KK.EQ.1) CALL SSL1(SP1, IOB, JOB)
        TYPE 3000
3000   FORMAT(' Enter 1 for lining : ', $)
        ACCEPT *, KK
        IF(KK.EQ.1) CALL SLN1(SP1)
C
C

```

```

TYPE 4000
4000  FORMAT('  Enter 1 for deleting : ', $)
      ACCEPT *,KK
      IF(KK.EQ.1) CALL SDE2(SP1,IOB,JOB)
C
C
C
888   CONTINUE
C
C
C
      CALL GETFNM('  Enter output file : ', 'PIC', NAME2, NC)
      OPEN(UNIT=2, NAME=NAME2, TYPE='NEW', ACCESS='DIRECT',
1     FORM='UNFORMATTED', RECORDSIZE=128)
      CALL NEWHDR(HSP1, 'S', 9, NAME2)
      CALL PICWT(2, HSP1, 1, 8)
      CALL PICWT(2, SP1, 9, 32)
      CLOSE(UNIT=1)
      CLOSE(UNIT=2)
      GO TO 111
2     CONTINUE
      STOP
      END
C
C
C
C     PROGRAM OF THRESHOLDING AT 2 , 4 , 8  LEVELS .
C     THIS PROGRAM ASSURES SMOOTHING ALSO .
C
C*****C
      SUBROUTINE SSL1(SP1,IOB,JOB)
C*****C
C
C
C     LOGICAL*1 SP1(128,128) , IOB(4000) , JOB(4000)
C
C
C
TYPE 2000
2000  FORMAT('  Enter Channel   0 , 1 , 2 or 3 : ', $)
      ACCEPT *,K2
      K2=K2+1
      GO TO (12,34,56,78),K2
C
C
C
12    DO 2 I=1,128
      DO 2 J=1,128
      IF(SP1(I,J).GE.0) GO TO 1
      SP1(I,J)=0
      GO TO 2
1     SP1(I,J)=-1
2     CONTINUE
      GO TO 7

```

```

C
C
34      DO 4 I=1,128
        DO 4 J=1,128
          IF(SP1(I,J).LT.64.AND.SP1(I,J).GT.-65) GO TO 3
          SP1(I,J)=0
          GO TO 4
3        SP1(I,J)=-1
4        CONTINUE
        GO TO 7
C
C
C
56      DO 6 I=1,128
        DO 6 J=1,128
          IF((SP1(I,J).LT.32.AND.SP1(I,J).GT.-33).OR.(SP1(I,J)
1        .LT.76.AND.SP1(I,J).GT.63).OR.(SP1(I,J).LT.-64.AND.S
2        P1(I,J).GT.-97)) GO TO 5
          SP1(I,J)=0
          GO TO 6
5        SP1(I,J)=-1
6        CONTINUE
78       CONTINUE
C
C
      TYPE 2500
2500    FORMAT(' Enter 4 for 4 levels , 2 for 2 levels ', $)
      ACCEPT *,KK
      IF(KK.NE.4) GO TO 211
      DO 21 I=1,128
      DO 21 J=1,128
        IF(SP1(I,J).GE.64) SP1(I,J)=64
        IF(SP1(I,J).LE.63.AND.SP1(I,J).GE.0) SP1(I,J)=0
        IF(SP1(I,J).LE.-1.AND.SP1(I,J).GE.-64) SP1(I,J)=-1
        IF(SP1(I,J).LE.-65) SP1(I,J)=-65
21      CONTINUE
        GO TO 212
211     CALL STH1(SP1)
212     CONTINUE
      TYPE 4000
4000    FORMAT(' Enter 1 for no processing , 2 for GR1 ,
2        3 FOR SL1 ', $)
      ACCEPT *,KK
      GO TO (170,27,37) KK
27      CONTINUE
C
C
      CALL SGR1(SP1,IOB,JOB)
C
      GO TO 170
37      CONTINUE
C
      CALL SGR1(SP1,IOB,JOB)
C

```

```

C
7  CONTINUE
   IB=2
   JB=2
   DO 13 I=IB,127
   DO 13 J=IB,127
   IF(SP1(I,J).EQ.0) GO TO 13
   IA=SP1(I-1,J-1)+SP1(I-1,J)+SP1(I-1,J+1)+SP1(I,J-1)
1  +SP1(I,J+1)+SP1(I+1,J-1)+SP1(I+1,J)+SP1(I+1,J+1)
   IF(IA.LT.-1) GO TO 13
   SP1(I,J)=0
   IF(I.LE.2.OR.J.LE.2) GO TO 13
   IB=I-2
   JB=J-2
   GO TO 7
13  CONTINUE
C
C  Deletion of inconsistent data
C
   DO 9 I=1,128
   DO 9 J=1,127,126
   DO 9 K=1,2
   SP1(I,J+K-1)=0
   SP1(J+K-1,I)=0
   IB=2
   JB=2
9  CONTINUE
C
   DO 10 I=IB,127
   DO 10 J=JB,127
   IF(SP1(I,J).EQ.0) GO TO 10
   IA=SP1(I-1,J)+2*SP1(I,J+1)+4*SP1(I+1,J)+8*SP1(I,J-1)
   IF(IA.NE.-5.AND.IA.NE.-10) GO TO 10
   SP1(I,J)=0
   IF(I.LE.2.OR.J.LE.2) GO TO 10
   I=I-2
   J=J-2
   GO TO 9
10  CONTINUE
C
C
   DO 11 I=2,127
   DO 11 J=2,127
   IA=SP1(I,J)+2*SP1(I,J+1)+4*SP1(I+1,J+1)+8*SP1(I+1,J)
   IF(IA.NE.-5.AND.IA.NE.-10) GO TO 11
   SP1(I,J)=0
   SP1(I,J+1)=0
   SP1(I+1,J)=0
   SP1(I+1,J+1)=0
11  CONTINUE
   DO 16 I=2,127
   DO 16 J=2,127
   IF(SP1(I,J).EQ.0) GO TO 16
   IA=SP1(I,J-1)*SP1(I,J+1)*SP1(I-1,J)*SP1(I+1,J)

```

```

        IF(IA.EQ.0) SP1(I,J)=-2
16      CONTINUE
        DO 17 I=1,128
        DO 17 J=1,128
        IF(SP1(I,J).NE.-2) GO TO 18
        SP1(I,J)=-1
        GO TO 17
18      SP1(I,J)=0
17      CONTINUE
170     CONTINUE
C
C
C
        RETURN
        END
C
C
C*****C
        SUBROUTINE SGR1(SP1,IOB,JOB)
C*****C
C
C
        LOGICAL*1 SP1(128,128)
        LOGICAL*1 IOB(4000) , JOB(4000)
C
C      Gradient
C
        DO 10 I=1,128
        DO 10 J=1,127,126
        DO 10 K=1,2
        SP1(I,J+K-1)=127
        SP1(J+K-1,I)=127
10      CONTINUE
C
C
        K=1
        DO 4 I=1,127
        DO 4 J=1,127
        IF(K.GT.4000) GO TO 44
        IF(SP1(I,J).LT.SP1(I+1,J).OR.SP1(I,J).LT.SP1(I,J+1))
2      GO TO 1
        GO TO 2
1      IOB(K)=I
        JOB(K)=J
        K=K+1
2      IF(SP1(I+1,J+1).LT.SP1(I,J+1).OR.SP1(I+1,J+1).LT.SP1
1      (I+1,J)) GO TO 3
        GO TO 4
3      IOB(K)=I+1
        JOB(K)=J+1
        K=K+1
4      CONTINUE
C
44     CONTINUE

```

```

DO 5 I=1,128
DO 5 J=1,128
SP1(I,J)=0
5  CONTINUE
C
C
C
DO 6 KK=1,K-1
SP1(IOB(KK),JOB(KK))=-1
6  CONTINUE
C
C
C
RETURN
END
C
C
C*****C
SUBROUTINE SGR(SP1)
C*****C
C
C
LOGICAL*1 SP1(128,128)
C
C
DO 1 I=1,127
DO 1 J=1,127
IF(SP1(I,J).NE.SP1(I,J+1).OR.SP1(I,J).NE.SP1(I+1,J))
2  SP1(I,J)=10
1  CONTINUE
C
C
DO 3 I=1,128
DO 3 J=1,128
IF(SP1(I,J).EQ.10) GO TO 2
SP1(I,J)=0
GO TO 3
2  SP1(I,J)=-1
3  CONTINUE
RETURN
END
C
C
C
THIS PROGRAM SMOOTHES LINES IN AN IMAGE AND DELETES
THE INCONSISTANT INFORMATION .
C
C*****C
SUBROUTINE SLN1(SP1)
C*****C
C
C
LOGICAL*1 SP1(128,128)
C
C

```

```

DO 6 I=1,126
DO 6 J=1,126
IA=SP1(I,J)+SP1(I+1,J)+SP1(I,J+1)+SP1(I+1,J+1)
IF(IA.NE.-4) GO TO 6
IA=SP1(I-1,J-1)+SP1(I-1,J)+SP1(I,J-1)
IF(IA.NE.0) GO TO 3
SP1(I,J)=0
3  IA=SP1(I-1,J+1)+SP1(I-1,J+2)+SP1(I,J+2)
   IF(IA.NE.0) GO TO 4
   SP1(I,J+1)=0
4  IA=SP1(I+1,J-1)+SP1(I+2,J-1)+SP1(I+2,J)
   IF(IA.NE.0) GO TO 5
   SP1(I+1,J)=0
5  IA=SP1(I+1,J+2)+SP1(I+2,J+1)+SP1(I+2,J+2)
   IF(IA.NE.0) GO TO 6
   SP1(I+1,J+1)=0
6  CONTINUE
C
C
C
DO 1 I=1,127
DO 1 J=1,127
IA=SP1(I,J)+5*SP1(I,J+1)+10*SP1(I+1,J)+20*SP1(I+1,J+
2  1)
   IF(IA.EQ.-16) SP1(I,J)=0
   IF(IA.EQ.-26) SP1(I,J+1)=0
   IF(IA.EQ.-31) SP1(I+1,J)=0
   IF(IA.EQ.-35) SP1(I+1,J+1)=0
1  CONTINUE
7  CONTINUE
C
C
C
RETURN
END
C
C
C  SUBROUTINE FOR NEIGHBORHOOD AVERAGING OF A PICTURE
C  128 BY 128 WITH A RADIUS OF 5 .
C
C*****C
C  SUBROUTINE SAV1(SP1,SP11)
C*****C
C
C  LOGICAL*1 SP1(128,128) , SP11(5,126)
C
C
C  DO 11 I=1,128
C  DO 11 J=1,126
C  IF(SP1(I,J).LT.0) SP1(I,J)=(256+SP1(I,J))/2
C  IF(SP1(I,J).GE.0) SP1(I,J)=(SP1(I,J))/2
11 CONTINUE
DO 2 I=3,7
DO 2 J=3,126

```

```

      AV=0.
      DO 1 II=I-2,I+2
      DO 1 JJ=J-2,J+2
      AV=AV+SP1(II,JJ)
1     CONTINUE
      SP11(I-2,J)=AV/25.+0.5
2     CONTINUE
      DO 4 I=8,126
      DO 4 J=3,126
      AV=0.
      DO 3 II=I-2,I+2
      DO 3 JJ=J-2,J+2
      AV=AV+SP1(II,JJ)
3     CONTINUE
      SP1(I-5,J)=AV/25.+0.5
4     CONTINUE
      DO 5 I=126,8,-1
      DO 5 J=126,3,-1
      SP1(I,J)=SP1(I-5,J)
5     CONTINUE
      DO 6 I=3,7
      DO 6 J=3,126
      SP1(I,J)=SP11(I-2,J)
6     CONTINUE
      DO 99 I=1,128
      DO 99 J=1,128
      IF(SP1(I,J).GT.63) SP1(I,J)=2*SP1(I,J)-256
      IF(SP1(I,J).LE.63) SP1(I,J)=2*SP1(I,J)
99    CONTINUE
      RETURN
      END

C
C
C
C
C      THIS PROGRAM DELETES ALL THE INFORMATION IN A
C      PICTURE EXCEPT LINES . THE I AND THE J COORDINATES
C      REGISTERED .
C
C
C*****C
      SUBROUTINE SDE2(SP1,IOB,JOB)
C*****C
C
C
C      LOGICAL*1 SP1(128,128)
C      LOGICAL*1 IOB(4000) , JOB(4000)
C
C
C      DO 1 I=3,126
C      SP1(I,3)=0
C      SP1(I,126)=0
C      SP1(3,I)=0
C      SP1(126,I)=0
1     CONTINUE

```

C
C
C

DELETE POINTS WITH MORE THAN 2 NEIGHBERS

```

DO 2 I=4,125
DO 2 J=4,125
IF(SP1(I,J).EQ.0) GO TO 2
IA=SP1(I-1,J-1)+SP1(I-1,J)+SP1(I-1,J+1)+SP1(I,J-1)+
1 SP1(I,J+1)+SP1(I+1,J-1)+SP1(I+1,J)+SP1(I+1,J+1)
IF(IA.EQ.-1.OR.IA.EQ.-2) GO TO 2
SP1(I-1,J-1)=0
SP1(I-1,J)=0
SP1(I-1,J+1)=0
SP1(I,J-1)=0
SP1(I,J+1)=0
SP1(I+1,J-1)=0
SP1(I+1,J)=0
SP1(I+1,J+1)=0
2 CONTINUE

```

C
C
C

DELETE OPEN LINES

```

KBCUR=1
KECUR=0

```

C
C

```

DO 3 I=4,125
DO 3 J=4,125
IF(SP1(I,J).EQ.0) GO TO 3
IA=SP1(I-1,J-1)+SP1(I-1,J)+SP1(I-1,J+1)+SP1(I,J-1)+
1 SP1(I,J+1)+SP1(I+1,J-1)+SP1(I+1,J)+SP1(I+1,J+1)
IF(IA.NE.-1) GO TO 3
IBEGIN=I
JBEGIN=J
KBCUR=KECUR+1
KBCUR=KBCUR/7
KBCUR=7*KBCUR+1

```

C
C

```

2 CALL SOL1(SP1,IOB,JOB,IBEGIN,JBEGIN,KBCUR,KECUR,LENG
TH)

```

C
C
3
C
C

CONTINUE

```

DO 4 I=4,125
DO 4 J=4,125
IF(SP1(I,J).EQ.0) GO TO 4
IA=SP1(I-1,J-1)+SP1(I-1,J)+SP1(I-1,J+1)+SP1(I,J-1)+
1 SP1(I,J+1)+SP1(I+1,J-1)+SP1(I+1,J)+SP1(I+1,J+1)
IF(IA.NE.-1) GO TO 44

```

C
C

```

      IBEGIN=I
      JBEGIN=J
      KBCUR=KECUR+1
      KBCUR=KBCUR/7
      KBCUR=7*KBCUR+1
C
C
      CALL SOL1(SP1,IOB,JOB,IBEGIN,JBEGIN,KBCUR,KECUR,LENG
2      TH)
C
C
      CLOSED COUNTOUR TO OPEN .
C
C
44      SP1(I,J)=0
      DO 440 II=I-1,I+1
      DO 440 JJ=J-1,J+1
      IF(SP1(II,JJ).EQ.-1) GO TO 441
440      CONTINUE
441      CONTINUE
      IBEGIN=II
      JBEGIN=JJ
      KBCUR=KECUR+1
      KBCUR=KBCUR/7
      KBCUR=7*KBCUR+1
C
C
      CALL SOL1(SP1,IOB,JOB,IBEGIN,JBEGIN,KBCUR,KECUR,LENG
2      TH)
C
C
      CONTINUE
C
C
      DELETE UNUSEFULL INFORMATION .
C
      DO 444 I=1,128
      DO 444 J=1,128
      SP1(I,J)=0
444      CONTINUE
C
C
      REGISTER POINTS OF THE LINES MEMORIZED .
C
      DO 5 KK=1,KECUR
      SP1(IOB(KK),JOB(KK))=-1
5      CONTINUE
C
C
      WRITE(4,100) ((IOB(I),JOB(I)) , I=1,KECUR)
100      FORMAT(10(3X,2I4))
C
      RETURN
      END

```

```

C
C*****C
  SUBROUTINE SOL1(SP1,IOB,JOB,IBEGIN,JBEGIN,KBCUR,KECU
2  R,LENGHT)
C*****C
C
C
  LOGICAL*1 SP1(128,128)
  LOGICAL*1 IOB(4000) , JOB(4000) , IBEGIN , JBEGIN
  INTEGER KBCUR , KECUR , LENGTH
C
C
  SP1(IBEGIN,JBEGIN)=0
  IOB(KBCUR)=IBEGIN
  JOB(KBCUR)=JBEGIN
  KB1=KBCUR+1
  WRITE(4,101)
101  FORMAT(///,' New line ',/, ' I   KB  KI  IOP  JOP')
  WRITE(4,102) KBCUR,KB1,IBEGIN,JBEGIN
102  FORMAT('      1',2I4,3X,2I4)
C
C
  SERACH NEXT POINT , WE ARE AT THE POINT K,L  AND
  SEARCHING THE NEXT POINT FK,FL .
C
  K=IBEGIN
  L=JBEGIN
C
C
  DO 3 I=1,1000
C
  DO 1 KK=K-1,K+1
  DO 1 LL=L-1,L+1
  IF(SP1(KK,LL).NE.-1) GO TO 1
  KF=KK
  LF=LL
  GO TO 2
1  CONTINUE
  GO TO 4
C
C
  END OF THE LINE .
C
2  SP1(KF,LF)=0
  KI=KBCUR+I
  IOB(KI)=KF
  JOB(KI)=LF
  KECUR=KBCUR+I
  I1=I+1
  WRITE(4,200) I1,KBCUR,KI,KF,LF
200  FORMAT(1X,3I4,3X,2I4)
  K=KF
  L=LF
3  CONTINUE
4  CONTINUE
  LENGTH=KECUR-KBCUR+1

```

```

C
C
      RETURN
      END
C
C*****C
      SUBROUTINE STH1(SP1)
C*****C
C
C
C      THIS PROGRAMM THTRSHHOLDS A PICTURE AT A LEVEL
C      SPECIFIED DURING THE EXECUTION .
C
C
C      LOGICAL*1 SP1(128,128)
C
C
C
C      TYPE 1000
1000  FORMAT(' ENTER LEVEL OF THRESHOLD (0 TO 255) : ', $)
      ACCEPT *,TH
C
C
C
C
C
      DO 1 I=1,128
      DO 1 J=1,128
      IS=SP1(I,J)
      IF(SP1(I,J).LT.0) IS=256+IS
      SP1(I,J)=0
      IF(IS.GE.TH) SP1(I,J)=-1
1      CONTINUE
C
C
      RETURN
      END
C
C*****C
      SUBROUTINE SBG1(SP1)
C*****C
C
C
C      LOGICAL*1 SP1(128,128)
C
C
C
      DO 1 I=2,127
      DO 1 J=2,127
      IF(SP1(I,J).EQ.0) GO TO 1
      AI=SP1(I-1,J-1)*SP1(I-1,J)*SP1(I-1,J+1)*SP1(I,J-1)*
1 SP1(I,J+1)*SP1(I+1,J-1)*SP1(I+1,J)*SP1(I+1,J+1)
      IF(AI.EQ.0.) SP1(I,J)=-10
1      CONTINUE

```

C
C

DO 2 I=1,128
DO 2 J=1,128
CONTINUE

2
C
C

RETURN
END

```

C      NAME :  SAV1.FOR
C
C      SUBROUTINE FOR NEIGHBERHOOD AVERAGING OF A PICTURE
C      128 BY 128 WITH A RADIUS OF 5 .
C
C*****C
C      SUBROUTINE SAV1(SP1,SP11)
C*****C
C
C
C      LOGICAL*1 SP1(128,128) , SP11(5,126)
C
C
C      DO 2 I=3,7
C      DO 2 J=3,126
C      AV=0.
C      DO 1 II=I-2,I+2
C      DO 1 JJ=J-2,J+2
C      AV=AV+SP1(II,JJ)
1      CONTINUE
C      SP11(I-2,J)=AV/25.+0.5
2      CONTINUE
C      DO 4 I=8,126
C      DO 4 J=3,126
C      AV=0.
C      DO 3 II=I-2,I+2
C      DO 3 JJ=J-2,J+2
C      AV=AV+SP1(II,JJ)
3      CONTINUE
C      SP1(I-5,J)=AV/25.+0.5
4      CONTINUE
C      DO 5 I=126,8,-1
C      DO 5 J=126,3,-1
C      SP1(I,J)=SP1(I-5,J)
5      CONTINUE
C      DO 6 I=3,7
C      DO 6 J=3,126
C      SP1(I,J)=SP11(I-2,J)
6      CONTINUE
C      RETURN
C      END

```

```

C      NAME : SE1.FOR
C
C      This program maps the Grey-levels of a given image
C      to ease segmentation . The mapping here is Exponen-
C      tial around the center of the image .
C
C      LOGICAL*1 SP1(64,64)
C      LOGICAL*1 HSP1(512) , NAME2(36)
C
C      TYPE 1000
1000  FORMAT('  ENTER BETA : ', $)
      ACCEPT *,BET
      TYPE 2000
2000  FORMAT('  ENTER # OF LEVELS ', $)
      ACCEPT *,XL
      DO 10 I=1,64
      DO 10 J=1,64
      X1=EXP(-BET*((FLOAT(I)-31.5)**2+(FLOAT(J)-31.5)**2))
      IX1=XL*X1
      X1=IX1/XL
      SI=255.*(1.-X1)
      IF(SI.GT.255.) SI=255.
      IF(SI.GE.128.) SI=SI-256.
      SP1(I,J)=SI
10    CONTINUE
C
      DO 20 I=1,64
      SP1(I,1)=(-1)**I
      SP1(I,64)=(-1)**(I+1)
      SP1(1,I)=(-1)**I
      SP1(64,I)=(-1)**(I+1)
20    CONTINUE
C
      TYPE 3000
3000  FORMAT('  ENTER 1 TO TRANSLATE THE PICTURE : ', $)
      ACCEPT *,KK
      IF(KK.NE.1) GO TO 99
      DO 30 I=63,4,-1
      DO 30 J=63,4,-1
      SP1(I,J)=SP1(I-2,J-2)
30    CONTINUE
C
99    CONTINUE
C
      CALL GETFNM('  Enter output file : ','PIC',NAME2,NC)
      OPEN(UNIT=2,NAME=NAME2,TYPE='NEW',ACCESS='DIRECT',
1     FORM='UNFORMATTED',RECORDSIZE=128)
      CALL NEWHDR(HSP1,,'V',9,,NAME2)
      CALL PICWT(2,HSP1,1,8)
      CALL PICWT(2,SP1,9,8)
      CLOSE(UNIT=2)
      STOP
      END

```

```

C      NAME :  SR1.FOR
C
C      This program searches for closed contours  in  a seg-
C      mented image .
C
C      LOGICAL*1 SP1(128,128) , HSP1(512) , NAME1(36)
C      LOGICAL*1 NAME2(36)
C      LOGICAL*1 IOB1(256) , JOB1(256)
C
C
C
111    TYPE 1000
1000   FORMAT('  Enter 1 to continue : ', $)
      ACCEPT *,K1
      IF(K1.NE.1) GO TO 999
      CALL GETFNM('  Enter input file : ', 'PIC', NAME1, NC)
      OPEN(UNIT=1, NAME=NAME1, TYPE='OLD', ACCESS='DIRECT',
1      FORM='UNFORMATTED', RECORDSIZE=128)
      CALL PICRD(1, SP1, 1, 1)
      CALL RDHDR(SP1, , , IDSTBK)
      CALL PICRD(1, SP1, IDSTBK, 32)
C
C
      IB=2
      JB=2
      DO 9 I=IB, 127
      DO 9 J=JB, 127
      IF(SP1(I, J).EQ.0) GO TO 9
      IA=SP1(I-1, J-1)+SP1(I-1, J)+SP1(I-1, J+1)+SP1(I, J-1)
1      +SP1(I, J+1)+SP1(I+1, J-1)+SP1(I+1, J)+SP1(I+1, J+1)
      IF(IA.NE.-2) GO TO 9
      K=I
      L=J
      IST=I
      JST=J
      IOB1(1)=I
      JOB1(1)=J
      IB=I
      JB=J
      IEN=I+1
      JEN=J-1
      IA=-SP1(I, J+1)-2*SP1(I+1, J+1)
      GO TO (1, 2), IA
1      IDE=I
      JDE=J+1
      GO TO 3
2      IDE=I+1
      JDE=J+1
3      SP1(I, J)=0
      K=IDE
      L=JDE
      K2=2

```

```

C
  4  CONTINUE
C
C
      SP1(K,L)=0
      K1=K
      L1=L
      DO 5 KK=K-1,K+1
      DO 5 LL=L-1,L+1
      IF(SP1(KK,LL).NE.-1) GO TO 5
      KA=KK
      LA=LL
      GO TO 7
5     CONTINUE
6     IF(KA.EQ.K1.AND.LA.EQ.L1) GO TO 8
7     IOB1(K2)=KA
      JOB1(K2)=LA
      K=KA
      L=LA
      K2=K2+1
      GO TO 4
8     IF(K.EQ.IEN.AND.L.EQ.JEN) GO TO 10
      WRITE(5,100)
100    FORMAT(//,' Non-closed contour for the object .')
      GO TO 999
9     CONTINUE
10    CONTINUE
      CALL GETFNM(' Enter output file : ','PIC',NAME2,NCHAR)
      OPEN(UNIT=2,NAME=NAME2,TYPE='NEW',ACCESS='DIRECT',
1     FORM='UNFORMATTED',RECORDSIZE=128)
      CALL NEWHDR(HSP1,,'S',9,,NAME2)
      CALL PICWT(2,HSP1,1,8)
      CALL PICWT(2,SP1,9,32)
      CLOSE(UNIT=1)
      CLOSE(UNIT=2)
      GO TO 111
999   CONTINUE
      STOP
      END

```

```

C      NAME :  ROTV.FOR
C
C      This program simulates local and general rotation
C      in cloud images .
C
C      LOGICAL*1 SP1(64,64) , SP2(64,64)
C      LOGICAL*1 HSP1(512) , HSP2(512)
C      CHARACTER*36 NAME1 , NAME2
C      LOGICAL*1 TY , SI , TI(8)
C      INTEGER SB , NB , MO , DA , YE , FL
C
C      DATA SI/'V'/ , TI/'P','I','C','T','U','R','E',0/
C
C      CALL GETFNM(' Enter input file : ','PIC',NAME1,NCHAR)
C      OPEN(UNIT=1,NAME=NAME1,TYPE='OLD',ACCESS='DIRECT',
2      FORM='UNFORMATTED',RECORDSIZE=128)
C      CALL PICRD(1,SP1,1,1)
C      CALL RDHDR(SP1,TY,SI,SB,NB,TI,MO,DA,YE,FL)
C      CALL PICRD(1,SP1,SB,NB)
C      CLOSE(UNIT=1)
C
C      TYPE 1000
1000    FORMAT(' Enter the center of the rotation (IC,JC) :
2      ',,$,$)
C      ACCEPT * , IC , JC
C      TYPE 2000
2000    FORMAT(' Enter the rotation angle in degrees : ',,$)
C      ACCEPT * , ALPHA0
C      ALPHA0=0.017453*ALPHA0
C      ALPHA=ALPHA0
C      TYPE 3000
3000    FORMAT(' Enter 1 for Variable Rotation : ',,$)
C      ACCEPT * , K
C      IF(K.EQ.1) GO TO 10
C      KK=0
C      GO TO 20
10      CONTINUE
C      KK=1
C      TYPE 4000
4000    FORMAT(' Enter Radius of Half-rotation <0,16> : ',,$)
C      ACCEPT * , RA
20      CONTINUE
C
C
C      ROTATION
C
C
C      TYPE 5000
5000    FORMAT(' ENTETR 1 FOR TYPE : ',,$)
C      ACCEPT * , KKK
C
C
C

```

```

DO 23 I=1,64
  SP2(I,1)=-1
  SP2(I,64)=-1
  SP2(1,I)=-1
  SP2(64,I)=-1
23 CONTINUE
C
C
DO 2 I=2,63
DO 2 J=2,63
  IF(KK.NE.1) GO TO 1
  D1=(I-IC)**2+(J-JC)**2
  D1=SQRT(D1)
  D1=-D1/1.44/RA
  ALPHA=ALPHA0*(EXP(D1))
1 CONTINUE
  AA=57.0*ALPHA
  IF(KKK.EQ.1) TYPE * , AA
  XI=(I-IC)*COS(ALPHA)+(J-JC)*SIN(ALPHA)+IC
  XJ=(J-JC)*COS(ALPHA)-(I-IC)*SIN(ALPHA)+JC
  II=XI+0.5
  JJ=XJ+0.5
  IF(II.GT.63) II=63
  IF(JJ.GT.63) JJ=63
  IF(II.LT.2) II=2
  IF(JJ.LT.2) JJ=2
C
  SP2(I,J)=SP1(II,JJ)
C
C
2 CONTINUE
C
C
100 FORMAT(4X,10I4)
  CALL GETFNM(' Enter Output file : ','PIC',NAME2,NC)
  OPEN(UNIT=2,NAME=NAME2,TYPE='NEW',ACCESS='DIRECT',
1 FORM='UNFORMATTED',RECORDSIZE=128)
  CALL NEWHDR(HSP2,TY,SI,9,8,TI,FL)
  CALL PICWT(2,HSP2,1,8)
  CALL PICWT(2,SP2,9,8)
  CLOSE(UNIT=2)
  STOP
  END

```

```

C      NAME :  B4.FOR
C
C      This program correlates contour segments from images
C      already segmented . It outputs the centroid of the
C      best segment-match and draws a Field vector from the
C      centroid of the segment origin to the centroid of
C      the best match .
C
C      INTEGER I1(1000) , I2(1000) , J1(1000) , J2(1000)
C      INTEGER LI(100) , LJ(100) , IO(100) , JO(100)
C      REAL F1(7) , F2(7) , F3(7)
C
C
C      TYPE 1000
1000  FORMAT(' Enter length of code of image 1 : ', $)
      ACCEPT * , LN1
      TYPE 2000
2000  FORMAT(' Enter length of code of image 2 : ', $)
      ACCEPT * , LN2
      TYPE 3000
3000  FORMAT(' Enter a threshold for a search area : ', $)
      ACCEPT * , TH
C
C
C      READ(1,*) (( I1(I) , J1(I) ) , I=1, LN1 )
C      READ(2,*) (( I2(I) , J2(I) ) , I=1, LN2 )
C
C
C      LN17=LN1/7
C      DO 6 I=1, LN17
C      LI(I)=0
C      LJ(I)=0
C
C      LI(I)=I1(7*(I-1)+4)
C      LJ(I)=J1(7*(I-1)+4)
C      IO(I)=LI(I)
C      JO(I)=LJ(I)
C
C      DO 2 J=1, 7
C      D1=(I1(7*(I-1)+1)-I1(7*(I-1)+J))**2
C      D2=(J1(7*(I-1)+1)-J1(7*(I-1)+J))**2
C      F1(J)=SQRT(D1+D2)
C      CONTINUE
C
C      RMAX=-2.
C      DO 5 K=4, LN2-4
C      D1=(LI(I)-I2(K))**2
C      D2=(LJ(I)-J2(K))**2
C      D12=SQRT(D1+D2)
C      IF(D12.GT.TH) GO TO 5
C      IC=I2(K)
C      JC=J2(K)
C

```

C
C

```
DO 4 KK=K-3,K+3
D1=(I2(K-3)-I2(KK))*2
D2=(J2(K-3)-J2(KK))*2
F2(KK-K+4)=SQRT(D1+D2)
F3(K-KK+4)=SQRT(D1+D2)
```

4

CONTINUE

C
C

```
DO 14 J=1,6
A=F2(J+1)-F2(J)
IF(A.GT.2.0) GO TO 55
```

14

CONTINUE

C
C
C

Segment Correlation

```
CALL SRXY(F1,F2,R)
IF(RMAX.GT.R) GO TO 44
RMAX=R
```

IO(I)=IC

JO(I)=JC

44

CONTINUE

C
C

```
CALL SRXY(F1,F3,R)
IF(RMAX.GT.R) GO TO 55
RMAX=R
```

IO(I)=IC

JO(I)=JC

55

CONTINUE

5

CONTINUE

6

CONTINUE

C
C

TYPE 4000

4000 FORMAT(' Enter LU (4 or 5) ', \$)

ACCEPT *, KK

ILA=LN17/10

DO 7 I=1,10

K=10*(I-1)

WRITE(KK,100) ((LI(K+J) , LJ(K+J)) , J=1,10)

100

FORMAT(//,5X,10(I3,' ',I3,2X),4X)

WRITE(KK,200) ((IO(K+J) , JO(K+J)) , J=1,10)

200

FORMAT(5X,10(I3,' ',I3,2X))

7

CONTINUE

STOP

END

C

```

C
C      Segment Correlation
C
C*****C
      SUBROUTINE SRXY(SF1,SF2,R)
C*****C
C
      REAL SF1(7)
      REAL SF2(7)
C
C
      EP=0.0001
      S1=0.
      S2=0.
      S11=0.
      S22=0.
      S12=0.
C
C
      DO 1 I=1,7
      S1=S1+SF1(I)
      S2=S2+SF2(I)
      S11=S11+(SF1(I))**2
      S22=S22+(SF2(I))**2
      S12=S12+SF1(I)*SF2(I)
1    CONTINUE
C
C
      R1=S12-(S1*S2)/7.0
      R2=S11-(S1**2)/7.0
      R3=S22-(S2**2)/7.0
      R=R1/(SQRT(ABS(R2*R3)))
      IF(R.GT.(1.0).OR.R.LT.(-1.0)) TYPE 1000
1000  FORMAT('  WRONG  R ')
C
C
      RETURN
      END

```

```

C      NAME :  TH1.FOR
C
C      This a Thresholding program for 128 by 128 images .
C
      LOGICAL*1 SP1(128,128) , HSP1(512) , NAME1(36)
      LOGICAL*1 NAME2(36)
C
C
111    TYPE 1000
1000   FORMAT(//,' Enter 1 to continue : ', $)
      ACCEPT *,K1
      IF(K1.NE.1) GO TO 2
      CALL GETFNM(' Enter input file : ','PIC',NAME1,NC)
      OPEN(UNIT=1,NAME=NAME1,TYPE='OLD',ACCESS='DIRECT',
1      FORM='UNFORMATTED',RECORDSIZE=128)
      CALL PICRD(1,SP1,1,1)
      CALL RDHDR(SP1,,,IDSTBK)
      CALL PICRD(1,SP1,IDSTBK,32)
C
C
      TYPE 2000
2000   FORMAT(' Enter the Threshold ( 0 to 255 ) : ', $)
      ACCEPT *,IT
C
C
      DO 1 I=1,128
      DO 1 J=1,128
      IS=SP1(I,J)
      IF(SP1(I,J).LT.0) IS=266-IS
      SP1(I,J)=0
      IF(IS.GE.IT) SP1(I,J)=-1
1      CONTINUE
C
C
      CALL GETFNM(' Enter output file : ','PIC',NAME2,NCHAR)
      OPEN(UNIT=2,NAME=NAME2,TYPE='NEW',ACCESS='DIRECT',
1      FORM='UNFORMATTED',RECORDSIZE=128)
      CALL NEWHDR(HSP1,,'S',9,,NAME2)
      CALL PICWT(2,HSP1,1,8)
      CALL PICWT(2,SP1,9,32)
      CLOSE(UNIT=1)
      CLOSE(UNIT=2)
      GO TO 111
2      CONTINUE
      STOP
      END

```

```

C      NAME :  VIP.FOR
C
C      This program performs the slicing of an image at the
C      different values of the Video Integrator Processor .
C
      LOGICAL*1 SP1(64,64) , SP2(64,64)
      LOGICAL*1 NAME1(36) , NAME2(36),HSP1(512),HSP2(512)
C
111    CONTINUE
      TYPE 1000
1000   FORMAT(' Enter 1 to continue : ', $)
      ACCEPT * , KK
      IF(KK.NE.1) GO TO 999
C
      CALL GETFNM(' Enter input file : ', 'PIC', NAME1, NC)
      OPEN(UNIT=1, NAME=NAME1, TYPE='OLD', ACCESS='DIRECT',
2      FORM='UNFORMATTED', RECORDSIZE=128)
      CALL PICRD(1, SP1, 1, 1)
      CALL RDHDR(SP1, , IDSTBK)
      CALL PICRD(1, SP1, IDSTBK, 8)
      CLOSE(UNIT=1)
C
      DO 1 I=1, 64
      DO 1 J=1, 64
      IX=SP1(I, J)
      IF(SP1(I, J).LT.0) IX=256+SP1(I, J)
      SP1(I, J)=IX/4
1      CONTINUE
C
      TYPE 2000
2000   FORMAT(' Enter 5 Threshold-values : ', $, $, $, $, $)
      ACCEPT * , TH1, TH2, TH3, TH4, TH5
C
      DO 2 I=1, 64
      DO 2 J=1, 64
      K=SP1(I, J)
      IF((K.GE.TH5).OR.(K.GE.TH2.AND.K.LT.TH3)) SP2(I, J)=0
      IF((K.GE.TH4.AND.K.LT.TH5).OR.(K.GE.TH1.AND.K.LT.TH2
2      )) SP2(I, J)=-1
      IF((K.LT.TH1).OR.(K.GE.TH3.AND.K.LT.TH4)) SP2(I, J)
2      =127
2      CONTINUE
C
      CALL GETFNM(' Enter Output file : ', 'PIC', NAME2, NC)
      OPEN(UNIT=2, NAME=NAME2, TYPE='NEW', ACCESS='DIRECT',
1      FORM='UNFORMATTED', RECORDSIZE=128)
      CALL NEWHDR(HSP2, , 'V', 9, , NAME2)
      CALL PICWT(2, HSP2, 1, 8)
      CALL PICWT(2, SP2, 9, 8)
      CLOSE(UNIT=2)
999    CONTINUE
      STOP
      END

```

```

C      NAME : FIG.FOR
C
C      This program generates cells for Tornado Simulation.
C
C      LOGICAL*1 SP1(64,64)
C      LOGICAL*1 NAME2(32) , HSP1(512)
C
111    CONTINUE
      TYPE 1000
1000   FORMAT(' Enter 1 to continue : ', $)
      ACCEPT * , K
      IF(K.NE.1) GO TO 999
      VALMAX=0.
      VALMIN=1.
      PI=3.14159
      PHI0=1.759
      D=127.0
C
      TYPE 2000
2000   FORMAT(' Enter Center <IC,JC> , Width and length
2      Cell ')
      TYPE 3000
3000   FORMAT(' <A,B> , Shift Angle <AL> : ', $, $, $, $, $)
      ACCEPT * , ICENT , JCENT , A , B , AL
C
C
      TYPE 4000
4000   FORMAT(' Enter max and min Intinsity : ', $, $)
      ACCEPT * , MAXIN , MININ
      SINAL=SIN(AL)
      COSAL=COS(AL)
C
      DO 1 I=1,64
      DO 1 J=1,64
      X=I-ICENT
      Y=J-JCENT
      X=X*COSAL-Y*SINAL
      Y=X*SINAL+Y*COSAL
      PHI=0.0
      IF(X.NE.0.0) PHI=ATAN(Y/X)
      A1=SQRT(X**2+Y**2)
      A2=(A-B)*EXP(-ABS(PHI)/PHI0)
      A3=(PI-ABS(PHI))/PI
      D1=(A1+A2*A3)**2
      D1=D1/3000
      IF(D1.LT.0.0000001) D1=0.0
      VAL=EXP(-D1)
      IF(VAL.GT.VALMAX) VALMAX=VAL
      IF(VAL.LT.VALMIN) VALMIN=VAL
      II=D*VAL
      SP1(I,J)=II
1      CONTINUE

```

C

```

DO 2 I=1,64
DO 2 J=1,64
X=SP1(I,J)/D
X=(MAXIN-MININ)/(VALMAX-VALMIN)*(X-VALMAX)+MAXIN
II=4.0*X
IF(II.GT.127) II=II-256
SP1(I,J)=II
CONTINUE

```

2

C

C

```

CALL GETFNM(' Enter Output file : ','PIC',NAME2,NC)
OPEN(UNIT=2,NAME=NAME2,TYPE='NEW',ACCESS='DIRECT',
2 FORM='UNFORMATTED',RECORDSIZE=128)
CALL NEWHDR(HSP1,,'V',9,,NAME2)
CALL PICWT(2,HSP1,1,8)
CALL PICWT(2,SP1,9,8)
CLOSE(UNIT=2)

```

C

C

```

GO TO 111

```

C

C

```

999 CONTINUE
STOP
END

```

```

C      NAME :  FD.FOR
C
C      This program tests the Fourier Descriptors
C
C      REAL SP2(256) , XMAG(256)
C
C
111    CONTINUE
      TYPE 2000
2000   FORMAT('  ENTER 1 TO CONTINUE  : ', $)
      ACCEPT * , KK
      IF(KK.NE.1) GO TO 999
C
C
      TYPE 1000
1000   FORMAT('  ENTER FREQUENCY : ', $)
      ACCEPT * , FRE
C
C
      DO 1 I=1,256
      AR=6.28/255*FLOAT(I-1)*FRE
      SP1(I)=SIN(AR)
      SP2(I)=SP1(I)
      SP1(I)=((-1)**I)*SP1(I)
1      CONTINUE
      CALL FFT(SP1,256,1.0)
      DO 2 I=1,256
      XMAG(I)=CABS(SP1(I))
2      CONTINUE
      SPMIN=SP2(1)
      XMAMIN=XMAG(1)
      SPMAX=SP2(1)
      XMAMAX=XMAG(1)
      DO 3 I=1,256
      IF(SP2(I).LT.SPMIN) SPMIN=SP2(I)
      IF(SP2(I).GT.SPMAX) SPMAX=SP2(I)
      IF(XMAG(I).LT.XMAMIN) XMAMIN=XMAG(I)
      IF(XMAG(I).GT.XMAMAX) XMAMAX=XMAG(I)
3      CONTINUE
C
      TYPE * , SPMIN , SPMAX
      TYPE * , XMAMIN , XMAMAX
C
C
      DO 4 I=1,256
      SP2(I)=(SP2(I)-SPMIN)/(SPMAX-SPMIN)
      XMAG(I)=(XMAG(I)-XMAMIN)/(XMAMAX-XMAMIN)
4      CONTINUE
C
C
C

```

```

        KOUT=3
        IIN=8
        IOUT=3
        DO 6 I=1,256
        NB=50*SP2(I)+1
6      CONTINUE
C
C
        DO 5 I=1,256
        NB=50*XMAG(I)+1
        WRITE(KOUT,100) I , SP2(I) , XMAG(I),( IIN , J=1,NB)
100    FORMAT(1X,I4,2X,F5.3,3X,F5.3,5X,51I1)
5      CONTINUE
        GO TO 111
999    CONTINUE
C
C
        STOP
        END
C
C
C      This SUBROUTINE takes the One-Dimensional FFT of 'N'
C      elements located in 'F' . If 'S'=1.0 , the forward
C      transform is taken . If 'S'=-1.0, the inverse trans-
C      form is taken . 'N' is an Integer power of 2 . The
C      elements of 'F' are divided by F(0) .
C
C*****C
      SUBROUTINE FFT(F,N,S)
C*****C
C
      COMPLEX*8 F(N),U,W,T
      LN=IFIX(ALOG(FLOAT(N))/ALOG(2.0))
      NV2=N/2
      NM1=N-1
      J=1
C
C
      DO 10 I=1,NM1
        IF(I.GE.J) GO TO 1
        T=F(J)
        F(J)=F(I)
        F(I)=T
1      K=NV2
2      IF(K.GE.J) GO TO 3
        J=J-K
        K=K/2
        GO TO 2
3      J=J+K
10     CONTINUE
C
C

```

```
DO 40 L=1, LN
  LE=2**L
  LE1=LE/2
  U=(1.0,0.0)
  W=CMPLX(COS(3.141593/LE1),-S*SIN(3.141593/LE1))
  DO 30 J=1, LE1
    DO 20 I=J, N, LE
      IP=I+LE1
      T=F(IP)*U
      F(IP)=F(I)-T
      F(I)=F(I)+T
20    CONTINUE
      U=U*W
30    CONTINUE
40  CONTINUE
  RETURN
END
```

APPENDIX C
INPUT/OUTPUT

```

C      NAME :  RGBPIC.FOR
C
C      This program generates coded color pictures      form
C      monochrome cloud pictures.
C
C      LOGICAL*1 SP1(256,64)
C      LOGICAL*1 NAME1(36) , NAME2(36) ,HSP1(512),HSP2(512)
C
111    CONTINUE
      TYPE 1001
1001   FORMAT('  Enter 1 to continue : ', $)
      ACCEPT * , KK
      IF(KK.NE.1) GO TO 999
C
C
      CALL GETFNM('  Enter input file : ', 'PIC', NAME1, NC)
      OPEN(UNIT=1, NAME=NAME1, TYPE='OLD', ACCESS='DIRECT',
2      FORM='UNFORMATTED', RECORDSIZE=128)
      CALL PICRD(1, SP1, 1, 1)
      CALL RDHDR(SP1, , , IDSTBK)
C
C
      T1=20
      T2=30
      T3=35
      T4=40
      T5=45
      T6=50
      T7=55
C
C
      TYPE 2000
2000   FORMAT('  Enter 0 for R , 1 for G , 2 for B : ', $)
      ACCEPT * , IC
C
C
      CALL GETFNM('  Enter output file : ', 'PIC', NAME2, NC)
      OPEN(UNIT=2, NAME=NAME2, TYPE='NEW', ACCESS='DIRECT',
1      FORM='UNFORMATTED', RECORDSIZE=128)
      CALL NEWHDR(HSP2, , 'M', 9, , NAME2)
      CALL PICWT(2, HSP2, 1, 8)
C
C
      DO 99 K=1, 4
      IS=9+32*(K-1)
      CALL PICRD(1, SP1, IS, 32)
C
C
      IF(IC.EQ.0) GO TO 11
      IF(IC.EQ.1) GO TO 22
      IF(IC.EQ.2) GO TO 33
      GO TO 111

```

```

C
11  CONTINUE
    DO 1 I=1,256
    DO 1 J=1,64
    IX=SP1(I,J)
    IF(SP1(I,J).LT.0) IX=256+SP1(I,J)
    IX=IX/4
    SP1(I,J)=0
    IF(IX.GE.T4) SP1(I,J)=-1
1   CONTINUE
C
C
22  CONTINUE
    DO 2 I=1,256
    DO 2 J=1,64
    IX=SP1(I,J)
    IF(SP1(I,J).LT.0) IX=256+SP1(I,J)
    IX=IX/4
    SP1(I,J)=0
    IF(IX.GE.T2.AND.IX.LT.T4) SP1(I,J)=-1
    IF(IX.GE.T6) SP1(I,J)=-1
2   CONTINUE
C
C
33  CONTINUE
    DO 3 I=1,256
    DO 3 J=1,64
    IX=SP1(I,J)
    IF(SP1(I,J).LT.0) IX=256+SP1(I,J)
    IX=IX/4
    SP1(I,J)=0
    IF(IX.GE.T1.AND.IX.LT.T2) SP1(I,J)=-1
    IF(IX.GE.T3.AND.IX.LT.T4) SP1(I,J)=-1
    IF(IX.GE.T5.AND.IX.LT.T6) SP1(I,J)=-1
    IF(IX.GE.T7) SP1(I,J)=-1
3   CONTINUE
C
C
    CALL PICWT(2,SP1,IS,32)
C
C
99  CONTINUE
    CLOSE(UNIT=1)
    CLOSE(UNIT=2)
    GO TO 111
999 CONTINUE
    STOP
    END

```

```

C      NAME : QMDISE.FOR
C
C      This program displays arrows between ( I,J ) and
C      ( IOP,JOP ) from a file XXXXX.DAT, the number of er-
C      rows has to be specified at the beginning of the
C      data file .
C
C      CHARACTER*36 NAME1
C      INTEGER I(3000) , J(3000)
C      REAL XIOP(3000) , YJOP(3000)
C      REAL MAG
C
C
C      111      CONTINUE
C              TYPE 1000
C              TYPE 1001
C      1000      FORMAT(' Enter 1 to continue, -1 to draw lines : ')
C      1001      FORMAT(' Enter 7 to draw a square : ', $)
C              ACCEPT * , KK
C              IF(KK.EQ.-1) GO TO 110
C              IF(KK.EQ.7) GO TO 88
C              IF(KK.NE.1) GO TO 999
C              TYPE 400
C      400      FORMAT(' Enter 1 to initialize the sceen : ', $)
C              ACCEPT * , KK
C
C              IF(KK.EQ.1) CALL INIT
C              TYPE 410
C      410      FORMAT(' Enter Xorigin and Yorigin : ', $, $)
C              ACCEPT * , IXO , JYO
C
C              CALL GETFNM(' Enter input datafile', 'DAT', NAME1, NC)
C              OPEN(UNIT=1, NAME=NAME1, TYPE='OLD', ACCESS='SEQUENTIAL
2      ', FORM='FORMATTED')
C
C
C              TYPE 333
C      333      FORMAT(' ENTER MAGNIFICATION ( 1 OR HIGHER ) : ', $)
C              ACCEPT * , MAG
C              READ(1, *) NBMES
C              READ(1, *) ((I(K), J(K), XIOP(K), YJOP(K)) , K=1, NBMES)
C
C
C              SX=0.0
C              SY=0.0
C              SI=0.0
C              SJ=0.0
C              DO 10 N=1, NBMES
C              SX=SX+XIOP(N)
C              SY=SY+YJOP(N)
C              SI=SI+I(N)
C              SJ=SJ+J(N)
C      10      CONTINUE

```

```

C
DO 20 N=1,NBMES
XIOP(N)=XIOP(N)-(SX-SI)/FLOAT(NBMES)
YJOP(N)=YJOP(N)-(SY-SJ)/FLOAT(NBMES)
20 CONTINUE
C
C      Scaling
C
MAG=8.0*MAG
DO 1 K=1,NBMES
IXS=MAG*J(K)-1
IYS=MAG*(64-I(K))-1
IXEND=MAG*YJOP(K)-1
IYEND=MAG*(64-XIOP(K))-1
C
IXS=IXS+IXO
IYS=IYS+JYO
IXEND=IXEND+IXO
IYEND=IYEND+JYO
CALL DLINE(IXS,IYS,IXEND,IYEND)
C
1 CONTINUE
C
CLOSE(UNIT=1)
GO TO 111
110 TYPE 112
112 FORMAT(' Enter IXS,IYS,IXEND,and IYEND :',$,$,$,)$)
ACCEPT *, IXS , IYS , IXEND , IYEND
DO 11 K=1,3
KK=K-1
C
C      Draw
C
CALL VECTOR(KK,IXS,IYS,IXEND,IYEND,255,-1)
11 CONTINUE
C
C
GO TO 111
C
C
88 CONTINUE
DO 77 K=1,3
KK=K-1
CALL VECTOR(KK,0,0,0,511,255,-1)
CALL VECTOR(KK,0,511,511,511,255,-1)
CALL VECTOR(KK,511,511,511,0,255,-1)
CALL VECTOR(KK,511,0,0,0,255,-1)
77 CONTINUE
GO TO 111
C
C
999 CONTINUE
STOP
END

```

```

C      This program initialises the screen to display TREC-
C      Vectors .

```

```

C*****C

```

```

      SUBROUTINE INIT

```

```

C*****C

```

```

      CALL IP5INI
      CALL SETUP
      CALL ZEROMS(7)
      RETURN
      END

```

```

C

```

```

C

```

```

C      This SUBROUTINE displays arrows from TREC-Vectors .

```

```

C

```

```

C*****C

```

```

      SUBROUTINE DLINE(IXS,IYS,IXEND,IYEND)

```

```

C*****C

```

```

C

```

```

      IBPM=-1
      DO 1 K=1,3

```

```

C

```

```

C

```

```

      Draw

```

```

C

```

```

      KK=K-1

```

```

      CALL VECTOR (KK,IXS,IYS,IXEND,IYEND,255,IBPM)

```

```

      CALL VECTOR (KK,IXS,IYS,IXS-1,IYS-1,255,IBPM)

```

```

      CALL VECTOR (KK,IXS-1,IYS-1,IXS+1,IYS-1,255,IBPM)

```

```

      CALL VECTOR (KK,IXS+1,IYS-1,IXS+1,IYS+1,255,IBPM)

```

```

      CALL VECTOR (KK,IXS+1,IYS+1,IXS-1,IYS+1,255,IBPM)

```

```

      CALL VECTOR (KK,IXS-1,IYS+1,IXS-1,IYS-1,255,IBPM)

```

```

1  CONTINUE

```

```

      RETURN

```

```

      END

```

```

C      NAME : NOAAO.FOR
C
C      This program formats the NOAA data
C
C      INTEGER*2 PPSW(762)
C      INTEGER*2 PPV(762)
C      INTEGER*2 GVAL(762)
C      INTEGER*2 INTVAL(16)
C      INTEGER*2 TEMP(2332)
C      INTEGER*2 HEAD(42)
C      LOGICAL*1 NOAA(2332)
C
C
C      TYPE 1000
1000    FORMAT(' Enter First and Last Record : ',$, $)
C      ACCEPT * , IBEGIN , IEND
C      TYPE 1500
1500    FORMAT(' Enter Logic Unit ( 3 or 5 ) : ', $)
C      ACCEPT * , KK
C
C
C      CALL TAPIO(1,KEY)
C      CALL REWIND
C
C
C      DO 99 II=IBEGIN , IEND
C      LREC=2332
C      CALL GETDAT(NOAA,LREC)
C      IF(KEY.EQ.-10) GO TO 45
C      IF(KEY.EQ.-11) GO TO 45
C      WRITE(KK,1004)
1004    FORMAT(' HHMMSS DAZ AZM ELEV PPI RAN GS NS S AS CS ',
1          '1      'JUL CV PRF AB',
2          '2      ' HG AGC DAZ DVT DS EIB PRT IS M MTI CM SN')
C      DO 10 I=1,2332
10        TEMP(I)=NOAA(I)
C      DO 20 I=1,15
20        HEAD(I)=TEMP(I+1)
C      HEAD(7)=HEAD(7)/4
C      HEAD(15)=HEAD(14)
C      HEAD(14)=HEAD(14)/128
C      DO 40 I=16,22
40        HEAD(I)=TEMP(I)
C      HEAD(21)=MOD(HEAD(21),16)
C      HEAD(22)=MOD(HEAD(22),16)
C      HEAD(23)=(TEMP(22)/16)*4+TEMP(21)/16
C      HEAD(24)=MOD(TEMP(23),4)
C      HEAD(25)=MOD(TEMP(24),16)
C      HEAD(26)=MOD(TEMP(25),16)
C      HEAD(27)=TEMP(23)/4
C      HEAD(28)=(TEMP(25)/16)*4+TEMP(24)/16
C      HEAD(29)=MOD((TEMP(26)/8),2)

```

```

HEAD(30)=MOD((TEMP(26)/4),2)
HEAD(31)=MOD((TEMP(26)/2),2)
HEAD(32)=MOD(TEMP(26),2)
HEAD(33)=MOD(TEMP(27),2)
HEAD(34)=MOD((TEMP(27)/2),2)
HEAD(35)=MOD((TEMP(27)/4),2)
HEAD(36)=MOD((TEMP(27)/8),2)
HEAD(37)=TEMP(27)/16
HEAD(38)=MOD(TEMP(28),8)
HEAD(39)=MOD((TEMP(28)/8),2)
HEAD(40)=TEMP(28)/16
HEAD(41)=MOD(TEMP(29),8)
HEAD(42)=TEMP(29)/8
C WRITE(KK,200)HEAD
200 FORMAT(1X,6I1,2X,I1,2X,2(1X,3I1),2X,I1,2X,3I1,1X,I1,
1 2X,I1,2X,I1,1X,
2 2I1,1X,I1,2X,3I1,1X,2I1,3(2X,I1),4(3X,I1),2X,I1,3X,
3 I1,3X,I1,1X,I1,2(2X,I1),3X,I1)
DO 60 I=1,16
INTVAL(I)=TEMP(I+30)
60 CONTINUE
C WRITE(KK,300)
300 FORMAT('0INTEGRATOR VALUES (AGC)')
C WRITE(KK,400)INTVAL
400 FORMAT(1X,16I4)
DO 80 I=1,762
GVAL(I)=TEMP(I+46)
80 CONTINUE
C WRITE(KK,1504)
1504 FORMAT('0INTEGRATOR VALUES FOR 762 GATES')
C WRITE(KK,1604)GVAL
DO 100 I=1,762
PPV(I)=TEMP(I+808)
100 CONTINUE
DO 120 I=1,762
IF(PPV(I).GT.32)PPV(I)=- (64-PPV(I))
120 CONTINUE
C WRITE(KK,1704)
1704 FORMAT('0PULSE PAIR VELOCITY VALUES FOR 762 GATES')
C WRITE(KK,1604)PPV
1604 FORMAT(1X,32I4)
DO 140 I=1,762
PPSW(I)=TEMP(I+1570)
140 CONTINUE
C WRITE(KK,1804)
1804 FORMAT('0PULSE PAIR SPECTRAL WIDTHS FOR 762 GATES')
C WRITE(KK,1604)PPSW
C
C WRITE(KK,201) II , HEAD(24) , HEAD(25) , HEAD(26) ,
2 HEAD(1),HEAD(2),HEAD(3),HEAD(4),HEAD(5),HEAD(6) ,
3 HEAD(7) , HEAD(8) , HEAD(9) , HEAD(10) , HEAD(11) ,
4 HEAD(12) , HEAD(13)

```

```
C
201  FORMAT(10X,I5,2X,3I1,2X,6I1,2X,I1,2X,3I1,2X,3I1)
      TYPE * , II
99   CONTINUE
45   CONTINUE
      ITOTAL=IEND-IBEGIN+1
      TYPE 1001 , ITOTAL
1001 FORMAT('  Number of records read = ',I5)
C
C
      CALL REWIND
      STOP
      END
```

```

C
C      NAME :   IMF6.FOR
C
C      This program extracts 64 by 64 images from 'AR 17XX '
C      tapes . The intensity image is the average of 4 nei-
C      ghbors .
C
C      LOGICAL*1 SP1(64,64)
C      LOGICAL*1 BUFF(2332) , HSP1(512) , NAME2(36)
C
C      CALL TAPIO(1,KEY)
C
C      IPOS=1
111    CONTINUE
      TYPE 100
100    FORMAT('  Enter 1 to continue  : ', $)
      ACCEPT * , KK
      IF(KK.NE.1) GO TO 999
      TYPE 1000
      TYPE 1001
1000   FORMAT('  Enter 1 if no advance ,')
1001   FORMAT('  Enter Beginning record ( otherwise ): ', $)
      ACCEPT * , IADV
      TYPE 2000
2000   FORMAT('  Enter 1 for non-increasing,2 otherwise:', $)
      ACCEPT * , KI
C
C      CALL GETFNM('  Enter Output file : ','PIC',NAME2,NC)
C
C      ID=IADV-IPOS
C      IPOS=IADV
C      CALL SPAREC(ID)
C      DO 1 II=1,64
C      I=II
C      IF(KI.NE.1) I=65-II
C
C      CALL GETDAT(BUFF,2332)
C
C      IPOS=IPOS+1
C      IF(KEY.EQ.-10) GO TO 40
C      IF(KEY.EQ.-11) GO TO 40
C          DO 1 J=1,64
C              IX=2*(BUFF(111+2*J-1)+BUFF(111+2*J))
C              IF(IX.GE.128) IX=IX-256
C              SP1(65-J,I)=IX
1      CONTINUE
      DO 2 I=1,64
      SP1(1,I)=-1
      SP1(64,I)=-1
      SP1(I,1)=-1
      SP1(I,64)=-1
2      CONTINUE
      GO TO 998

```

```

C
  40      TYPE 3000
 3000    FORMAT('   End of file or end of record  !  ')
  998    CONTINUE
C
C
      OPEN(UNIT=2,NAME=NAME2,TYPE='NEW',ACCESS='DIRECT',
1  FORM='UNFORMATTED',RECORDSIZE=128)
      CALL NEWHDR(HSP1,, 'V',9,,NAME2)
      CALL PICWT(2,HSP1,1,8)
      CALL PICWT(2,SP1,9,8)
      CLOSE(UNIT=2)
      GO TO 111
  999    CONTINUE
C
C
      WRITE(5,4000) IPOS
 4000    FORMAT(' The position of the tape is : ',I6)
      TYPE 5000
 5000    FORMAT(' Enter desirable position : ', $)
      ACCEPT * , IR
      CALL SPAREC(IR-IPOS)
      STOP
      END

```

```

C
C      NAME :  MER2.FOR
C
C      This program merges 64 by 64 images  into 256 by 256
C      images .
C
      LOGICAL*1 SP1(256,16)
      LOGICAL*1 SP2(64,64) ,HSP1(512) , HSP2(512)
      LOGICAL*1 NAME1(36) , NAME2(36)
C
C
111      CONTINUE
      TYPE 1000
1000     FORMAT('  ENTER 1 TO CONTINUE : ','$)
      ACCEPT * , KI
      IF(KI.NE.1) GO TO 999
      TYPE 1500
1500     FORMAT('  ENTER ROW AND COLUMN # ','$)
      ACCEPT * , NR , NC
C
C
      CALL GETFNM('  ENTER A 64 IMAGE : ','PIC',NAME2,NC)
      OPEN(UNIT=1,NAME=NAME2,TYPE='OLD',ACCESS='DIRECT',
1  FORM='UNFORMATTED',RECORDSIZE=128)
      CALL PICRD(1,SP2,1,1)
      CALL RDHDR(SP2,,,IDSTBK)
      CALL PICRD(1,SP2,IDSTBK,8)
      CLOSE(UNIT=1)
      TYPE 2000
C
C
2000     FORMAT('  ENTER 1 IF YOU NEED A NEW 256 IMAGE : ','$)
      ACCEPT * , KJ
      IF(KJ.NE.1) GO TO 222
C
      CALL GETFNM('  ENTER NEW 256 IMAGE FILE : ',
2  'PIC',NAME1,NC)
      OPEN(UNIT=1,NAME=NAME1,TYPE='NEW',ACCESS='DIRECT',
1  FORM='UNFORMATTED',RECORDSIZE=128)
      CALL NEWHDR(SP1,,'M',9,,NAME1)
      CALL PICWT(1,SP1,1,8)
C
C
      DO 1 I=1,256
      DO 1 J=1,16
      SP1(I,J)=-1
1  CONTINUE
C
      DO 22 K=1,16
      IDSTBK=8*(K-1)+9
      CALL PICWT(1,SP1,IDSTBK,8)
22  CONTINUE

```

```
C          GO TO 333
222        CONTINUE
C
          CALL GETFNM('  ENTER OLD 256 IMAGE FILE : ',
1          OPEN(UNIT=1,NAME=NAME1,TYPE='OLD',ACCESS='DIRECT',
          FORM='UNFORMATTED',RECORDSIZE=128)
          CALL PICRD(1,SP1,1,1)
          CALL RDHDR(SP1,,,IDSTBK)
333        CONTINUE
C
          DO 3 K=1,4
          IDSTBK=32*(NR-1)+9+8*(K-1)
          CALL PICRD(1,SP1,IDSTBK,8)
          DO 2 I=1,64
          DO 2 J=1,16
          II=I+64*(NC-1)
          JJ=J+16*(K-1)
          SP1(II,J)=SP2(I,JJ)
2          CONTINUE
          CALL PICWT(1,SP1,IDSTBK,8)
3          CONTINUE
C
          CLOSE(UNIT=1)
          GO TO 111
999        CONTINUE
          STOP
          END
```

```

C
C      NAME :  MER4.FOR
C
C      This program retrieves a 64 by 64 image from a 256
C      by 256 image .
C
C      LOGICAL*1 SP1(256,64)
C      LOGICAL*1 SP2(64,64) ,HSP1(512) , HSP2(512)
C      LOGICAL*1 NAME1(36) , NAME2(36)
C
111      CONTINUE
        TYPE 1000
1000      FORMAT(' ENTER 1 TO CONTINUE : ', $)
        ACCEPT * , KI
        IF(KI.NE.1) GO TO 999
        TYPE 1500
1500      FORMAT(' ENTER ROW AND COLUMN # ', $)
        ACCEPT * , NR , NC
C
        CALL GETFNM(' ENTER A NEW 64 IMAGE : ',
2      'PIC',NAME2,NC)
        OPEN(UNIT=2,NAME=NAME2,TYPE='NEW',ACCESS='DIRECT',
1      FORM='UNFORMATTED',RECORDSIZE=128)
C
        CALL NEWHDR(SP2,, 'V',9,,NAME2)
        CALL PICWT(2,SP2,1,8)
C
        CALL GETFNM(' ENTER OLD 256 IMAGE FILE : ',
2      'PIC',NAME1,NC)
        OPEN(UNIT=1,NAME=NAME1,TYPE='OLD',ACCESS='DIRECT',
1      FORM='UNFORMATTED',RECORDSIZE=128)
        CALL PICRD(1,SP1,1,1)
        CALL RDHDR(SP1,,,IDSTBK)
        IDSTBK=32*(NR-1)+9
        CALL PICRD(1,SP1,IDSTBK,32)
C
C
        DO 2 I=1,64
        DO 2 J=1,64
        II=I+64*(NC-1)
        SP2(I,J)=SP1(II,J)
2      CONTINUE
C
C
        CALL PICWT(2,SP2,9,8)
C

```

```
CLOSE(UNIT=1)  
CLOSE(UNIT=2)  
GO TO 111  
999 CONTINUE  
STOP  
END
```

VITA

The author received his primary, secondary, and college education - up to Master's level - in French schools. Notably he received a statewide Achievement Award in both his primary and secondary graduation, and obtained his "Ingenieur Principal" diploma from "l' Ecole Nationale d'Ingenieur" with a nationwide Presidential Award.

Mr. Abidi joined the Image and Pattern Analysis Laboratory at The University of Tennessee, Knoxville to finish working toward a Ph.D. degree in Image Processing. His graduation is expected by late 1985. In his Master's and Ph.D. research, the author got involved in subjects such as: Digital Communication, Microprocessors and Microprocessor-based-systems architecture, Pattern Recognition, Image analysis, Motion measurement and interpretation in both weather forecasting and robotics, and VLSI and circuit design. He is a frequent contributor to the literature in these fields with either technical reports or papers.

Mr. Abidi is a member of numerous scientific, professional and honorary societies, including the Institute of Electrical and Electronics Engineers, the IEEE computer Society, Tau Beta Pi, Phi Kappa Nu, Phi Kappa Phi, and Eta Kappa Nu.