AN ALGORITHM FOR A DOLLAR BILL RECOGNITION SYSTEM

by

Anupam Singh

Thesis submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Master of Science

in

Electrical Engineering

APPROVED:

Dr. M. Nadler, Chairman

Dr. R. W. Conners

Dr. R. W. Ehrich

March, 1988

Blacksburg, Virginia

AN ALGORITHM FOR A DOLLAR BILL RECOGNITION SYSTEM

by

Anupam Singh

Dr. M. Nadler, Chairman

Electrical Engineering

(ABSTRACT)

This paper presents an algorithm for a dollar bill recognition system. Although this thesis describes it in detail for the specific application of designing a dollar bill recognition system, the algorithm is quite general and can be applied to a variety of pattern recognition problems. The scheme operates on the image of a corner of the bill. Hough transform is used to find the edges and the corner point in the image. If there is any skew in the edges, it is corrected and a 256 x 256 pixel image is obtained. This image is then compressed to an 8 x 8 matrix, and features are extracted from a two dimensional Walsh Transform of this matrix. The process of feature selection is based upon the standard deviations of the Walsh coefficients. These features are then used by a Sequential Classifier for classifying the bill.

Acknowledgements

I would like to thank Dr. M. Nadler for his guidance and support. Working with him provided a valuable experience. I would also like to thank Dr. R. W. Ehrich and Dr. R. W. Conners for sitting on my committee.

I dedicate this thesis to my parents.

Table of Contents

List of Illustrations

List of Tables

1.0 INTRODUCTION

This thesis presents an algorithm for a dollar bill recognition system. The algorithm is quite general and can be applied to a variety of pattern recognition problems. However, the algorithm will be explained with the specific purpose of designing a dollar bill recognition system. A recent report by the Treasury Department indicates that the Government is considering the abolition of the one dollar bill and introducing a one dollar coin. One of the reasons cited in the report is that a blind person can easily distinguish a one dollar coin from the coins of other denominations. This thesis presents a scheme which can eventually lead to the fabrication of a low-cost, hand-held device for recognizing dollar bills. The goal is to enable blind people to recognize dollar bills. The constraints are that the device should use a scheme which requires minimal hardware for implementation so that the device is affordable and can be carried along by a person without any inconvenience. Also, the device should be reliable enough i.e., the misclassification rate should be zero (or negligible) and the rejection rate should be as low as possible.

Before proceeding further, however, a review of basic pattern recognition theory is in order.

1.1 A Brief Review

Pattern recognition is the process of assigning objects (patterns) to one of several possible categories (classes).

The first step in the design of a pattern recognition system involves a detailed analysis of the sample space. The performance of the system depends upon the diversity of the input patterns. A set of patterns is selected from the input space to form what is known as the design set. This is a very crucial step of the process. The more accurate the design set is in representing the sample space, the better is the system's performance. After the design set has been collected, measurements are made on the patterns in the set. From these measurements are derived certain features to obtain a feature set. Next, a subset of features that best discriminates among patterns of different classes is selected. Finally, a classifier is designed which takes an input pattern and assigns a class to it based on the features obtained in the previous step. The performance of the system can be adjudged by examining the results over a test set of patterns. Again, the test set should also be representative of the pattern space. There are various methods of selecting the design and test sets and the interested reader is referred to [1]. If the results are not acceptable, modifications can be made to any of these steps and the process repeated until acceptable results are achieved.

The system is implemented in a corresponding manner. In very crude terms, a typical pattern classification system has three distinct subsystems [Figure 1 on page 4]. The first step is the process of making measurements on input data (object). Next, the Feature Extractor operates on the measurements and extracts features which are then used by the Classifier to assign the object to a particular class.

However, the trend now is to treat the process as a continuous one with more interactions among the subsystems [Figure 2 on page 5]. Instead of the three subsystems working independently of one another, information from the later stages is fed back to the earlier stages in order to improve the performance of the recognition system.

1.2 The Problem of Dollar Bill Recognition

After reviewing the basics of pattern recognition, we are now ready to examine the problem of dollar bill recognition. As mentioned before, the goal was to come up with a scheme of classification that could be implemented easily without requiring extensive hardware. The device, after fabrication, should be small enough in size so that it can be held in the hand just like a walkman or a portable AM/FM radio. It was decided that the device should be able to recognize the seven most frequently used denominations, namely one, two, five, ten, twenty, fifty and hundred. The scheme, by itself, is capable of handling other denominations also. It was the inability of the author to collect bills of other denominations for building up the design and test sets that was the primary cause of this restriction (Moreover, none of the various tellers whom the author met had ever seen bills of a denomination other than the ones listed above).

Keeping these goals in mind, it was decided that the most efficient way to handle the problem would be to consider the corner of the bill. Using the image of the entire bill

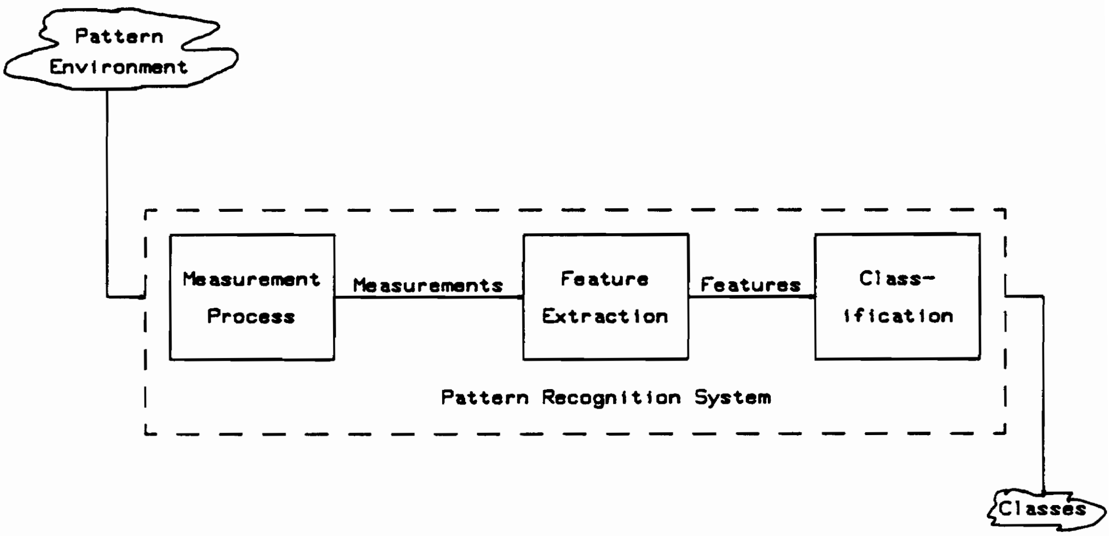


Figure 1. Pattern Classification System

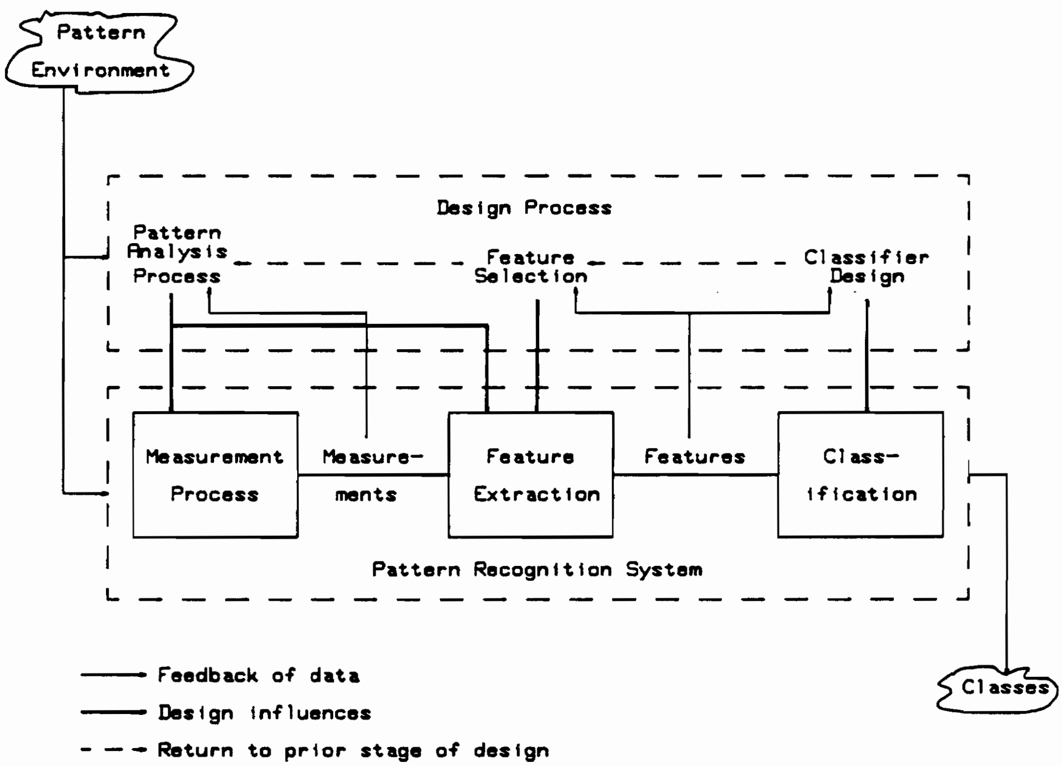


Figure 2. Pattern Classifier Design System

would provide much more information than would actually be needed, thereby requiring more hardware resources and increasing the cost and the size of the device. A scanning device would be mounted such that whenever a bill is inserted into the device, it would image a corner of the bill and store the data into memory. With this design in place, there are four different ways in which a bill can be inserted (Figure 3 on page 7 and Figure 4 on page 8). The corner imaged by the device depends on the way the bill is inserted. Thus, there are four corners for each denomination. Since the corners are different from one another, four classes are obtained for each denomination. Since seven denominations are being considered, we have 7 x 4 = 28 classes.

The design process began with the collection of a number of images of the corners of dollar bills. Utmost precaution was taken to ensure that the data base contained bills of all types - ranging from brand new bills to extremely worn -out bills for each class. The condition of each bill was meticulously recorded to enable a better understanding of the variations while processing the images. The next step was to choose a mask or window to standardize the image. This was necessitated by the observation that different bills have different margins from the edges (Figure A.1 on page 54). Moreover, the image itself is skewed with respect to the bill in certain cases (Figure A.2 on page 55).

Thus, it becomes imperative that the same region of the corner be compared for all the patterns. It was decided that a larger image (than the one eventually needed by the system) would be taken. From this image, the corner would be detected and a 256 x 256 pixel image would be obtained. This would ensure that the same region is considered for all the images. At this stage, it was decided to reduce the data to a matrix of 8 x 8 = 64 elements. The most important reason for doing so was that the mathematical transformations are very sensitive to small variations in the image [13]. Even if the image is

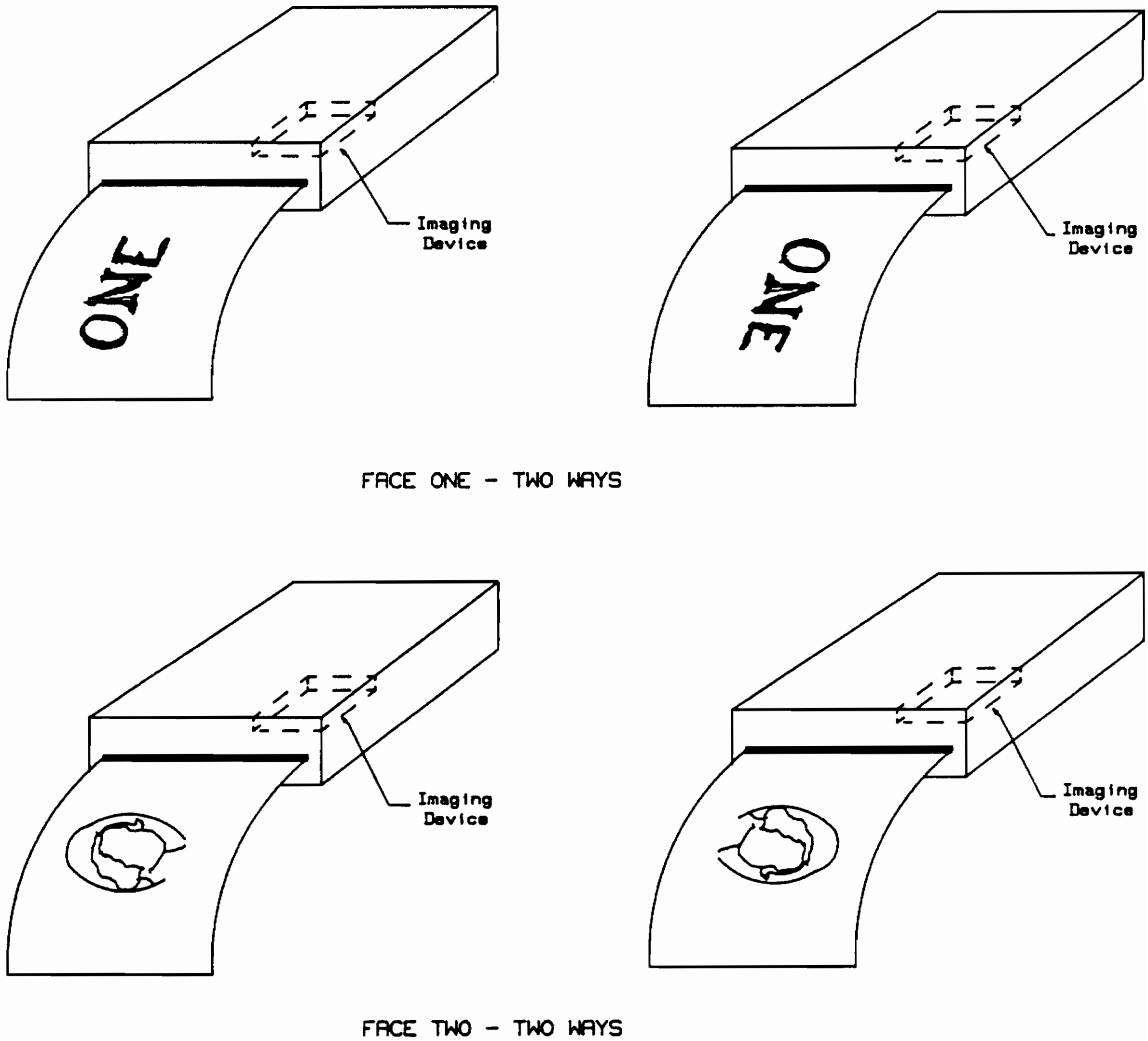


Figure 3. Inserting the bill into the device

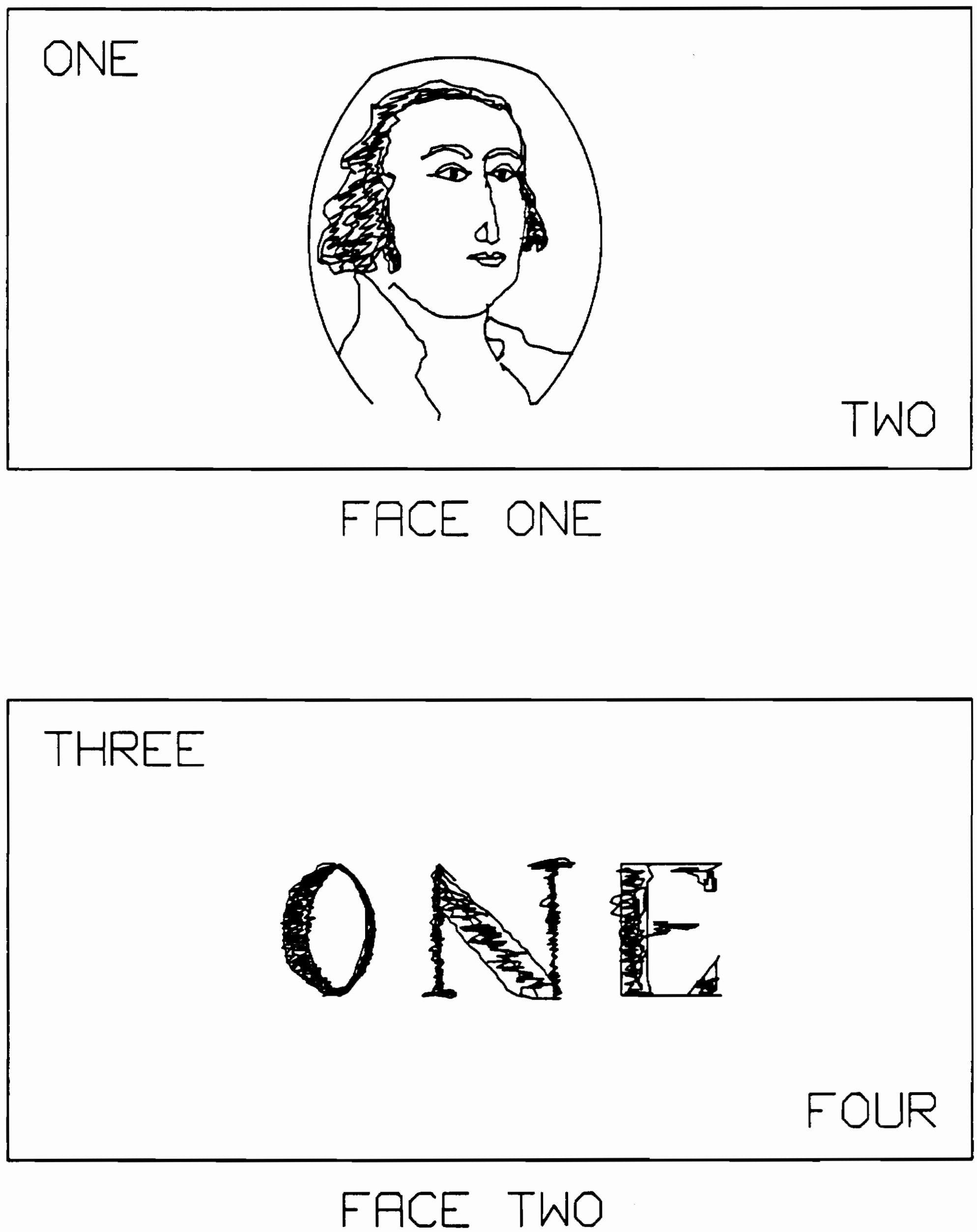


Figure 4. The four relevant corners of the bill

shifted by a pixel or two, there may be considerable differences in the values of the coefficients obtained after transformation. A consolidation of the larger image to a smaller image would make the system less susceptible to small shifts in the image and increase the reliability of the recognition scheme. Next, Walsh Transformation was applied to this matrix and another 8 x 8 matrix was obtained. The next step was to select appropriate features that would reliably discriminate amongst the different classes. Different methods were attempted and finally one was selected. After having obtained a feature set, different classifiers were tried on the design as well as the test set and the one that gave the best results was chosen.

1.3 Organization of The Thesis

Chapter 2 discusses the process of collecting the design set and the test set images. It also describes in detail the process of obtaining a standard 256 x 256 image and the techniques employed, namely edge and corner detection, estimation of the skew and skew correction. Chapter 3 discusses different transformations that have been used in pattern recognition and justifies the use of the Walsh Transform which was eventually chosen. The chapter also describes the various techniques that were employed in an attempt to overcome the variations due to the condition of the bills. The process of feature selection is discussed in detail in chapter 4. Chapter 5 presents a discussion on different classifiers. The various classifiers that were actually tried are also discussed here. The results obtained by the application of these schemes are presented in chapter 6. Chapter 7 suggests some modifications which can enhance the capabilities of this algorithm with reference to the problem of dollar bill recognition and also outlines a general scheme for implementing the algorithm to fabricate a hand-held, low-cost device for dollar-bill recognition.

2.0 IMAGE ACQUSITION AND STANDARDIZATION

This chapter presents the imaging process and details of the device used for taking the images. The processes of detecting the edges and the corner are also discussed in detail. The problems encountered during this process are enumerated. The problem of detecting the skew in the image and employing skew correction is also dealt with in this chapter.

2.1 Image Acquisition

Initially, a camera was used to take images of the corners of dollar bills. There were certain problems with this approach. First, the resolution was not very good and consequently, the images were not of a very good quality. Secondly, it was difficult to simulate the conditions that would exist when the device would finally be fabricated. After fabrication, the system would most likely have a fixed imaging device which would always have exactly the same distance from the inserted bill. It would also have one, or a particular number of threshold levels. With the camera, it was difficult to maintain the same aperture (as it had a continuously varying aperture control) which effectively meant that it was not possible to have the same thresholding as the images were collected over a long period of time. Thus, this approach was abandoned in favour of another scanning device - the Ricoh scanner. It has a resolution of 300 points per inch. It has 8 thresholding levels which can be set by the user before taking the image. The scanner digitizes the image and sends the compressed or uncompressed data to a file. For each scan, a file of 920 k bytes is generated. A program was written that would enable one to access any line of the entire image and display a particular portion of the image on the screen. To simulate the real - life situation, it was decided that a 40 byte by 300 line (equivalent to 320 x 300 pixels) segment would be extracted for each corner. This would ensure that the edge, corner and skew detection part of the scheme would have a sufficient margin for correcting the original image in order to obtain a standard 256 x 256 pixel image from the corner. All the images were taken with the threshold set to level three to ensure uniformity. After each scan, the image file generated by the scanner would be examined and a 40 byte by 300 line image around each corner would be extracted to create a pattern for further processing. Typically, each image from the scanner would accommodate 3 bills.

2.2 *Edge and Corner Detection*

Efficient corner detection is a very critical part of the entire process. There are several techniques of detecting edges and corners in a given image. A survey of the available techniques is neither possible nor warranted. Interested readers may refer to [2]. The technique that was chosen to detect the edges was a Hough transform technique. After detecting the two edges (which should ideally be perpendicular to each other), the corner of the image is detected by computing the intersection of the two edges. This technique was chosen as it was simple to implement.

Hough transforms are techniques in which points in the image are transformed into a parameter space [3]. The set of all the straight lines in the image can be viewed as constituting a two-parameter family. The parameter space can be described in a variety of ways e.g., slope-intercept or normal parametrization. The problem with the former is that both the slope and the intercept may become unbounded, thereby introducing implementation complexities. The latter was implemented as it does not present any such problem. The normal parameters, θ and ρ, of a line are defined as follows:

θ is the angle the normal to the straight line makes with the x axis and ρ is the algebraic distance of the line from the origin [Figure 5 on page 14].
The equation of a straight line in this notation is:

$$x^*\cos\alpha + y^*\sin\beta = \rho \tag{2.1}$$

If θ is restricted to the interval [0,pi], then every line in the x-y plane will have unique normal parameters. It can be demonstrated easily that every line in the image plane will correspond to a unique point in the parameter space. Thus, if one desires to find the straight line that fits a given set of points

$$\{(x_i , y_i)| \text{ i } = 1..n\} \tag{2.2}$$

in the image plane, then one should transform the points into corresponding figures in the parameter space. It can be readily seen that the figures corresponding to those points in the image plane that lie on a straight line will intersect at a point in the parameter space. Thus, the problem of finding colinear points in the image plane is now transformed into one of finding coincident points of intersection of their corresponding curves in the parameter space. Unfortunately, n points in the image plane will, in general, have n(n-1)/2 points of intersection which can be a prohibitively large number (in our case,

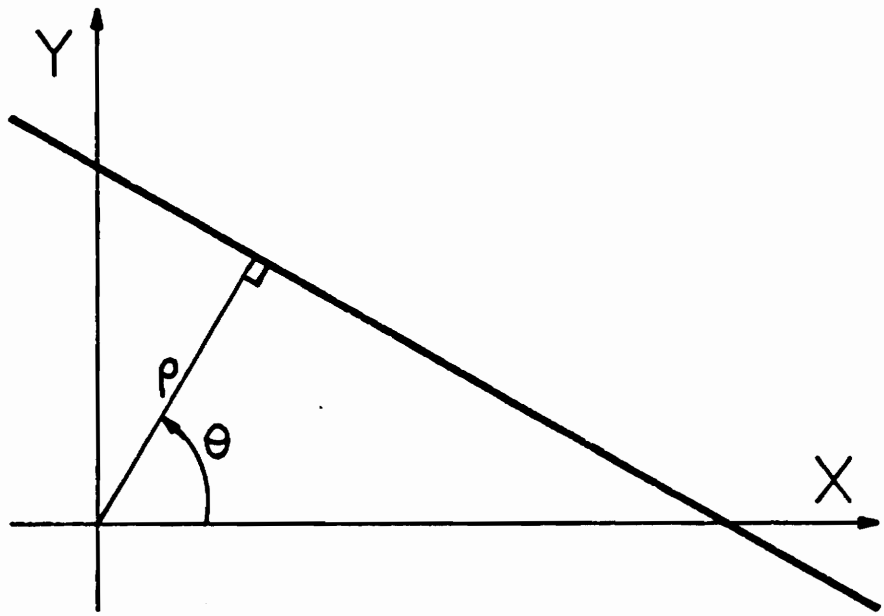


Figure 5. Parametric representation of a line

for example, n is 256 x 256 = 64k !). This can be overcome by quantizing the parameter space by constraining the normal parameters to certain levels. Also, as noted before, θ can be restricted to $[0,\pi]$ and ρ to $[-r,r]$ where r is the diagonal distance of the image plane (in case of a square image, like ours). The levels of quantization will, in general, depend upon the desired accuracy. The parameter space can now be treated as a two-dimensional array. Let there be p uniformly spaced values of θ in $[0,\pi]$ and q uniformly spaced values of ρ in $[-r,r]$. Then, for each image point, p ρ values are calculated, one corresponding to each of the p values θ can take. Since there are n image points, n x p computations are required. Thus, we have an o(n) algorithm (instead of the o(n^2) algorithm discussed above). This can be reduced further by observing the fact that we are detecting **edges** which have certain features:

- The two edges we are looking for have to lie within a certain distance from the corresponding boundaries of the image plane (which imposes limits on the values that ρ can take).

- The skew of the edges can lie only within a certain range (which further limits the values that θ can take).

The quantized region thus obtained represents a two dimensional array of accumulators. For each point (x_i, y_i) in the picture plane, the corresponding curve (as given by 2.1) is calculated and the cells in the quantized array along this curve are updated by incrementing the corresponding count by one. Thus, after the entire region of interest in the picture plane has been examined, each cell in the array records the total number of curves passing through it. At this stage, the array can be examined to find those cells that have high counts. If the count in a given cell (θ_i, ρ_i) is say, k, then exactly k points lie along the line whose normal parameters are (θ_i, ρ_i). The user can specify a threshold to find those cells whose count exceeds the threshold. The resulting cells correspond to a subset of image points that are nearly colinear.

Different values of p and q were tried and the results obtained for edges with different skews (generated by test programs) were examined. Finally, it was decided that ρ would be computed in steps of 2 and θ in steps of 0.22 degrees. To run the Hough Transform on all the points (320 x 300 = 9600 points) in the image would take a very long time (hours on an IBM PC). Hence, it was decided that a skeleton of the image would be obtained and this would be the input for the Hough transform. A point (x_i, y_i) would be called a skeletal point if:

- both (x_i, y_i) and $(x_i + 1,\ y_i)$ had a gray value of 1 (corresponding to a y edge point), or

- both (x_i, y_i) and $(x_i, y_i + 1)$ had a gray value of 1 (corresponding to an x edge point).

After obtaining the normal parameters for the x and y edges, this information was used to compute the point of intersection of the two edges, called (x0, y0).

2.3 Skew Estimation and Correction

As seen above, the Hough transform also provides an estimate of the angle which the edge makes with the x or y axis. This angle can be translated into the number of pixels by which the image has to be rotated in order to correct the skew. It may be noted here that both the x and y edges provide an independent estimate of the skew. Ideally, the skews should be such that the angle between the two edges is exactly 90 degrees. In practice, however, this is not always true. It was therefore decided that the skew of the image would be the mean of the two independent estimates. It was also decided that the system should be able to handle skews of less than 17 pixels in either direction. In other words, if the image is skewed by more than 3.57 degrees, the system would output an error message requesting the user to insert the bill again. The skew correction procedure was divided into two parts. First, the entire 256 x 256 image would be divided into a certain number of blocks depending upon the number of pixels by which the image was skewed. A skew of n pixels would mean that the image would be divided into n+1 blocks. For each row, the ith block from the corner would then be moved up or down, depending upon the direction of skew, by i-1 pixels. Next, a similar operation would be performed on each column of the image i.e., the ith block of pixels in each column of the image would be moved to the right or left by i-1 pixels. Once the skew in the image has been corrected by the above process, the desired 256 x 256 image is extracted from this image starting at the corner point (x0, y0). Figure A.3 on page 56 shows a computer image of a corner of a bill (320 x 300 pixels). Figure A.4 on page 57 shows the 256 x 256 pixel image obtained from it as a result of the processing described above.

After the above steps have been performed on the image, it is in the standard form.

3.0 TRANSFORMATION

The standard image obtained by the processes discussed in the previous chapter is 256 x 256 pixels. From the storage point of view, this would require 32 x 256 = 8k bytes (Since each pixel would have a value of 1 or 0, the pixel values for 8 successive bits in each row were packed together in one single byte, thereby requiring 32 bytes for each row of the image). As mentioned before, all the mathematical transformations are very sensitive to small shifts in the images. Even small variations in the locations of the edges or corners of the images would cause significant changes in the coefficients obtained after transformation. This could cause problems during feature selection and classification. If the data could be reduced without losing the information relevant to the process of recognition, we would be able to improve the reliability of the system significantly and make it less susceptible to the variations arising out of small shifts or skews. From the computational point of view, these data are quite voluminous and any mathematical operations on them would not only require too many operations (hence, too much time) but would also increase the cost and size of the device to implement the scheme. Thus, there is a need to reduce these data without sacrificing important information.

It was decided that the data should be reduced to an 8 x 8 matrix. Careful observation and some experimentation revealed that a simple averaging of the gray levels would serve the purpose. The 256 x 256 image was segmented into 8 x 8 = 64 blocks. Each block contained 32 x 32 = 1k pixels. The gray values of the 1k pixels were added together to obtain a number between 0 and 1024. This number was then divided by the final matrix size, i.e., 64 to obtain a value between 0 and 16 for each block. To verify that this 8 x 8 matrix still retained the useful information, a dithering technique was used to display these data. It was observed that enough information was resident in the 8 x 8 image to visually distinguish one class from another. Figure 6 on page 20 shows four patterns belonging to different classes. On the other hand, patterns belonging to the same class look very similar as demonstrated by Figure 7 on page 21.

Another important point is in order here. The ultimate goal of the system was to distinguish bills of one denomination from those of other denominations. The process of averaging mentioned above serves a dual purpose. It not only retains enough information to distinguish one class from another but also gets rid of the fine details in the image of the corner. These local patterns are not pertinent for achieving our objective. On the contrary, their presence may complicate the process of classification.

The 8 x 8 matrix is now ready for transformation. A review of the basics of transformation techniques and their relevance to pattern recognition is presented here.

3.1 A Brief History

The term *frequency* is applied to sinusoidal (periodic) functions that have uniformly spaced zero-crossings over an interval. The term *sequency* can be considered to be a generalization of frequency. Harmuth [4] introduced it to represent one-half the average number of zero-crossings per unit time. The term is especially useful in digital logic for it can be applied to distinguish functions that are not periodic and whose zero-crossings

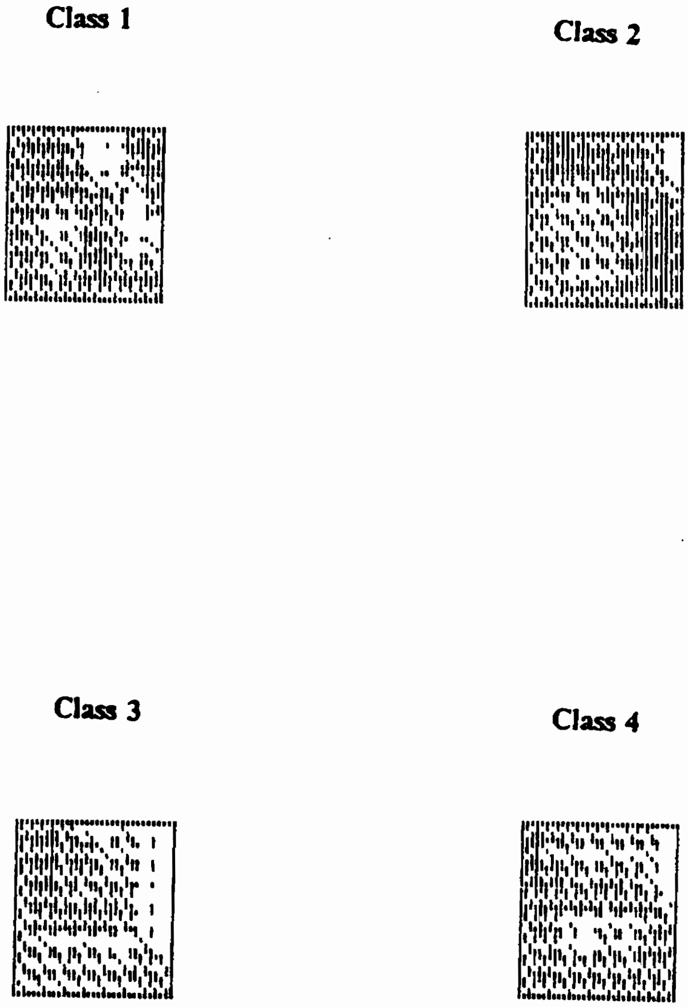
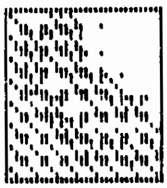
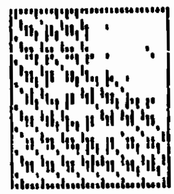


Figure 6. Patterns belonging to different classes.

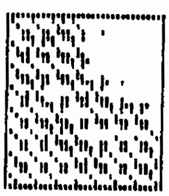
Pattern 1



Pattern 2



Pattern 3



Pattern 4

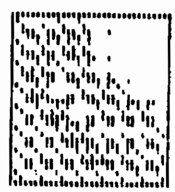


Figure 7. Patterns belonging to the same class.

are not uniformly spaced over an interval. This definition implies that sequency is to digital functions as frequency is to sinusoidal functions.

Rademacher functions are an incomplete set of orthonormal functions which were developed in 1922 [5]. The Rademacher function of order m, denoted by rad(m,t), is a train of rectangular pulses with $2_{(m-1)}$ cycles in the half-open interval [0,1), taking the values +1,-1. Rad(0,t) is the unit impulse. Rademacher functions are periodic with period 1, i.e.,

$$\mathrm{rad}(m,t + n \times 2^{1-m}) = \mathrm{rad}(m,t), \qquad m = 1,2,\ldots \qquad n = \pm 1, \pm 2, \ldots \quad (3.1)$$

Rademacher functions can be generated using the following recurrence relation:

$$\mathrm{rad}(m,t) = \mathrm{rad}(1,2^{m-1}t) \qquad (3.2)$$

where

$$\mathrm{rad}(1,t) = \begin{cases} 1, & t \in [0,1/2) \\ m1, & t \in [1/2,1) \end{cases}$$

The incomplete set of Rademacher functions was completed by Walsh in 1923 and it is now known as the set of Walsh functions. This set is generally classified into three groups. The distinction within these groups is due to the *order* in which the individual functions appear. The three different groups are :

1. **Sequency, or Walsh Ordering:** This is the ordering originally proposed by Walsh [6]. The sequency ordered Walsh functions are defined as:

$$S_w = \{wal_w(i,t), i = 0,1,\mathrm{N}-1\}$$

where

$$N = 2^n, \qquad n = 1,2,3,\ldots$$

The subscript w denotes Walsh ordering and i represents the i th member of S_w. The sequency s_i of $wal_w(i,t)$ is given as:

$$s_i = \begin{cases} 0, & i = 0 \\ i/2, & i \text{ even} \\ (i+1)/2, & i \text{ odd} \end{cases}$$

It is more instructive to represent these functions pictorially. The first 8 Walsh ordered functions (continuous and discrete) are shown in Figure 8 on page 25. It can be easily seen that the sequency of a Walsh function is greater that of the preceding one and has exactly one more zero-crossing in the open interval $t \in (0,1)$. This also explains why Walsh functions are also known as *Sequency ordering*.

2. **Dyadic, or Paley Ordering:** This type of ordering was intoduced by Paley [7]. The Paley ordered Walsh functions are defined as:

$$S_p = \{wal_p(i,t),\ i = 0,1,N-1\}$$

The subscript p denotes Paley ordering and i represents the i th member of S_p. The set S_p is related to the Walsh ordered set S_w by the relation:

$$wal_p(i,t) = wal_w[b(i),t]$$

where $b(i)$ represents the *Gray code-to-binary conversion* of i. The first 8 Paley ordered functions (continuous and discrete) are shown in Figure 9 on page 26.

3. **Natural, or Hadamard Ordering:** This set of Walsh functions is given by:

$$S_h = \{wal_h(i,t), i = 0,1,\mathrm{N} - 1\}$$

The subscript h denotes Hadamard ordering and i represents the i th member of S_h. The functions belonging to the set S_h are related to the Walsh ordered set by the relation:

$$wal_h(i,t) = wal_w[b < i > ,t]$$

where $< i >$ is obtained by the bit-reversal of i and, $b(< i >)$ is the *Gray code-to-binary conversion* of $< i >$. The first 8 Hadamard ordered functions (continuous and discrete) are shown in Figure 10 on page 27.

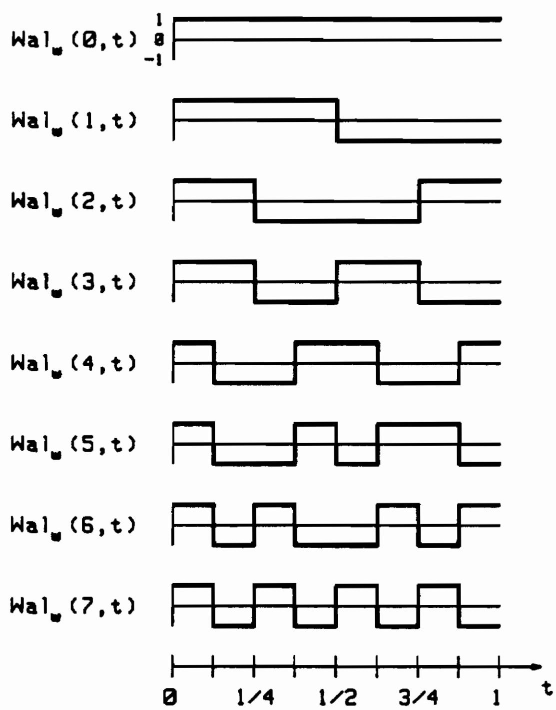
3.2 *Equalization and Normalization*

As mentioned before, the condition of the bills selected for imaging spanned a wide range. Consequently, there was an appreciable difference in some of the elements of the 8 x 8 matrices (henceforth referred to as coefficients) even though they belonged to the same class. To reduce such differences, different techniques were attempted.

Normalization is the process of dividing all the 64 coefficients by the value of the largest coefficient and then scaling up all the values to a desired range. In the case of normalization before taking the transform, this would mean that the values would be multiplied

$$H_w(3) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix}$$

Discrete

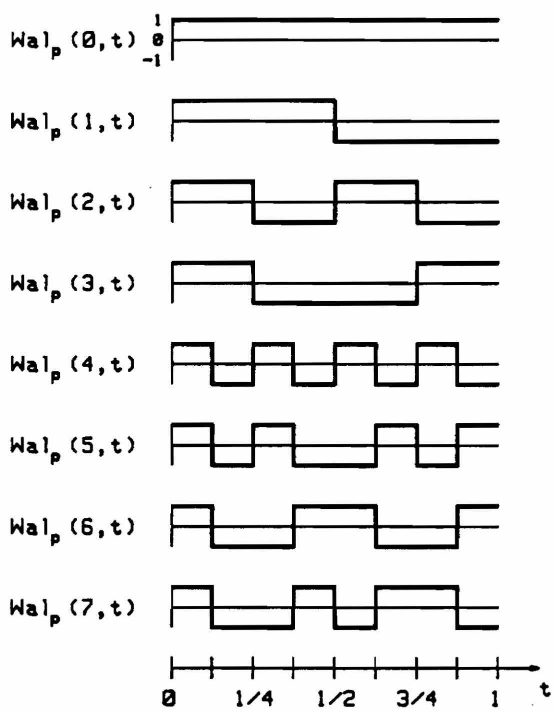


Continuous

Figure 8. The first 8 Walsh ordered Walsh Functions

$$H_p(3) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}$$

Discrete

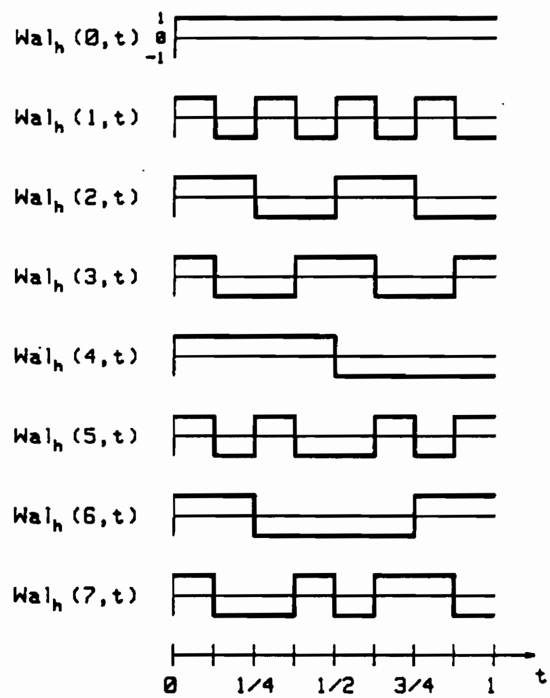


Continuous

Figure 9. The first 8 Paley ordered Walsh Functions

$$
H_h(3) = \begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\
1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\
1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\
1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\
1 & -1 & -1 & 1 & -1 & 1 & 1 & -1
\end{bmatrix}
$$

Discrete



Continuous

Figure 10. The first 8 Hadamard ordered Walsh Functions

by 16 to scale them up to the 0 to 16 range. Normalization after the transformation is done by dividing the coefficients by the (1,1) coefficient (which corresponds to the dc value and is always the largest among the 64 coefficients). The value is then scaled up by multiplying by 1000 (There is nothing sacred about choosing this factor; it was done to make the values consistent and facilitate comparison).

A pseudocode for equalization is given in Figure 11 on page 29. For each class, the maximum and the minimum of the 64 coefficients for each image were used to calculate the mean maximum and mean minimum values. Next, for each image of the class, the difference between its mean coefficient value (obtained simply as half of the difference between the maximum and minimum coefficient values for the image) and the overall mean coefficient value (half of the difference between the overall mean maximum and minimum values) is calculated and added to each of the 64 coefficients. This results in an additive equalization of the coefficients. Equalization can be done only before the transformation. The purpose of equalization is to try to bring the 64 coefficient values for each image of a class into the same range to compensate for the variations which arise due to the different conditions of the bills.

Different combinations of equalization and normalization were experimented with and the results analyzed. It was discovered that for combination of *normalization after transformation*, the values obtained for the *fom*'s were generally higher than those obtained by other combinations. Since the algorithm was based on the *fom* criterion, this combination was chosen. All the design and test images were processed through the aforementioned steps. For each class, a range for each of the 64 coefficients was obtained by calculating the mean and standard deviation and using two standard deviations

```
For each class
    begin
        Calculate mean maximum and mean minimum coeff
        Calculate the mean of these two (overall mean)
        For each pattern of the class
            begin
                Find the mean coeff value  (max+min)/2
                Compute the difference between this and overall mean
                Add this to all the 64 coeffs of the pattern
            end
    end
```

Figure 11. A pseudocode for equalization

on either side of the mean. These ranges were then used for selecting the features and the process is discussed in the next chapter.

4.0 FEATURE SELECTION

Feature selection is an important step in any pattern recognition system. A number of techniques have been developed over the years. Some of the more commonly used techniques are described briefly and their relative merits and disadvantages are also listed. This is followed by a description of the technique that was finally implemented.

Basically, there are two types of feature selection methods - transformation based and search based. Transformational techniques, as the name implies, transform the features from one domain into another one having a lower dimension. The reduction is achieved by forming linear combinations of features. Search techniques, on the other hand, do not transform the features from one domain into another but attempt to pick the best subset of features according to some criterion.

4.1 Transformation Based Methods

These methods generally require the computations of eigenvalues and basis functions. Since this usually requires extensive computations, these methods are more useful when the number of features is small. Two of the more popular transformation methods are the **Karhunen-Loeve (K-L) expansion** and **SELECT.**

4.1.1 K-L Expansion

This method is considered by many to be the most reliable for feature selection [9]. Originally proposed for representing a sample function of random processes, it was later modified by Fukunaga and Koontz for the feature selection problem [8].

Let $f_i(x_n)$ (i = 1,2,...,N) denote a set of discrete functions. Representing them as a column vector, we have:

$$F_i = \begin{vmatrix} f_i(x_1) \\ f_i(x_2) \\ \cdot \\ \cdot \\ \cdot \\ f_i(x_n) \end{vmatrix} \qquad (i = 1,2, \ldots, N)$$

The set of functions F_i can be expressed as a linear combination of a set of basis functions, $\phi_j(x_n)$, ($j = 1,2, \ldots, n$) as given by the following equation:

$$F_i = \sum_{j=1}^{n} \alpha_{ij}\phi_j$$

where ϕ_j is

$$\phi_i = \begin{vmatrix} \phi_j(x_1) \\ \phi_j(x_2) \\ \cdot \\ \cdot \\ \cdot \\ \phi_j(x_n) \end{vmatrix} \qquad (i = 1,2, \dots , N)$$

The basis functions can be obtained by solving

$$\mu_j \phi_j = S\phi_j \quad \text{and} \quad S = \underset{i}{E}\left[F_i F_i^t\right]$$

where E [ ] is the expected value over N samples. S is called the autocorrelation matrix, and μ_j and ϕ_j are the jth eigenvalue and eigenvector of S respectively. If all the basis functions are used to expand F_i , then there is no error. An error is introduced if the number of basis functions used for expansion is less than n. It can be shown that the basis functions can be selected in order of eigenvalues until the mean squared error becomes acceptably low [8]. As noted before, the **K-L expansion** was originally designed to guarantee the best set of features for reconstructing the original signal. However, it does not guarantee that it will be the best set for discrimination between classes. In order to overcome this problem, Fukunaga and Koontz describe a preprocessing technique that transforms the autocorrelation matrices so that

$$\mu_j^1 = 1 - \mu_j^2 \quad \text{and} \quad \phi_j^1 = \phi_j^2$$

Thus, the basis functions (or features) can be obtained by choosing ϕ_1^1 , ϕ_2^1, ... for class 1 and ϕ_n^2 , ϕ_{n-1}^2, ... for class 2 (For simplicity, only two classes were considered here). Young [9] shows that this set of features is best for discriminating between two classes.

However, there are certain problems with **K-L expansion.** The method is computationally very complex. Also, the method works well for two-class problems but gets complicated for multi-class applications. Moreover, the complexity of the method depends only on the number of original features, no matter how small the number of final features is.

4.1.2 SELECT Transformation

SELECT is a technique that generates an orthogonal set of features which best discriminates between different classes [10]. The features are first scaled and normalized and then rank ordered. The top ranked feature is accepted. Next, any correlation among the remaining features is removed and a new set of orthonormal features is thus obtained. Note that this set still has the same number of features. Reduction in the number of features can be achieved by the following method: Calculate the correlation coefficients between the most recently selected feature and each of the remaining unselected features. If any value is greater than a set threshold, then the particular unselected feature is ruled out permanently. The main problem with **SELECT** is that it is based on rank ordering. There could be cases where two lower ranked features have a combined discriminatory power greater than that of a single feature ranked higher than both of them. **SELECT** would still choose the higher ranked single feature. In general, the performance of **SELECT** is comparable to that of **K-L expansion** technique.

4.2 Search Based Methods

Search based strategies concentrate more on reducing the feature set size than on obtaining an optimal feature set. They are typically less computation intensive than transformational techniques. The final feature set can be formed either by adding (*bottom-up*) or deleting (*top-down*) one or more features to the current best set. The process terminates when the desired number of features is reached or when an acceptable error rate is achieved. Search strategies are basically of three types - **exhaustive, accelerated and stepwise.**

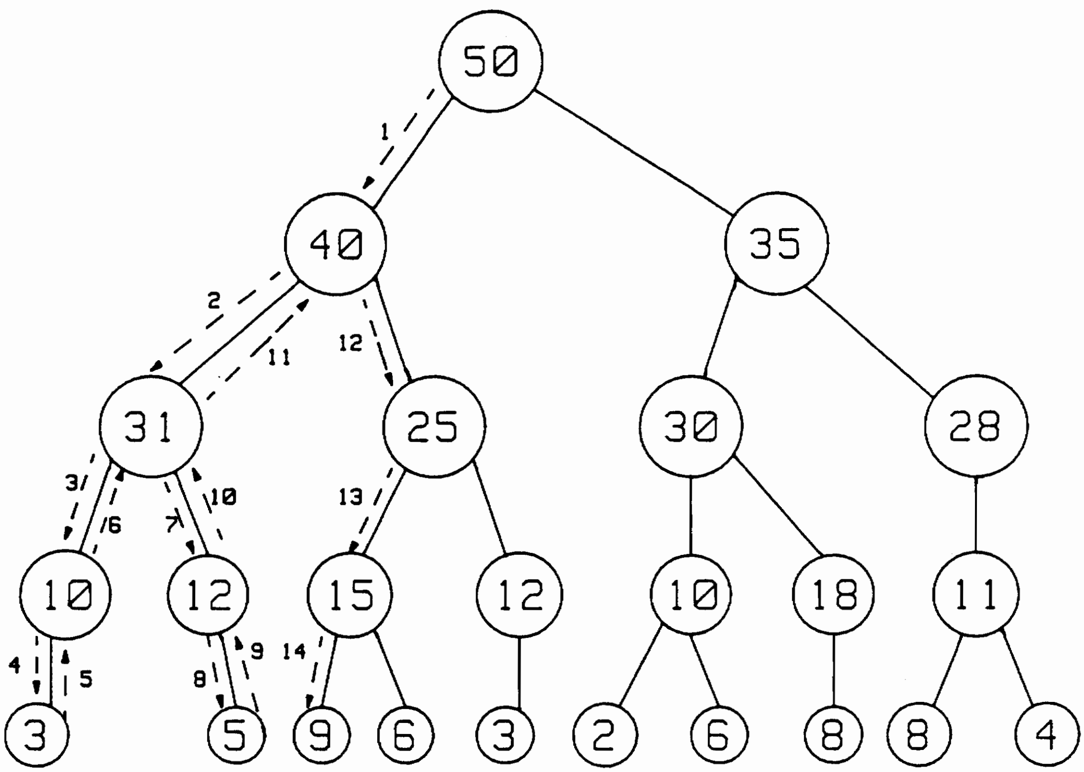
Exhaustive search tries all the possible combinations of features to select the best one. It is viable only when the number of features is quite small.

Accelerated search applies a branch and bound technique. It assumes that the criterion for selection varies monotonically (Most criteria, such as nearest neighbor, Bayesian and clustering possess this property). The technique is best explained with the help of a tree [Figure 12 on page 37]. The root node is the set of all features. Each leaf node represents a single feature. Each intermediate node represents a feature set having one feature less than its parent node. A tree for N features will have N-1 levels. The process begins by computing the classification at the root node and all its children. The search then proceeds along the child with the highest classification. At each level, the classification rates for all the nodes at that level are calculated and if the current node has a rate lower than that of any other in the same level, then the search backtracks and follows the next highest. The process continues until either a leaf node is reached, or an acceptable error rate is achieved or a certain predetermined depth of search is reached. This technique will always find the optimal set of features. However, even this method requires high computation times.

Stepwise methods are suboptimal and, in general, do not find an optimal feature set. A simple method of this category is to rank order the entire feature set and pick up the desired number of features starting from the top. This method is also known as the **n-best features** technique. There are certain inherent drawbacks in this method. In the event of two features having exactly the same classification rate, a scheme to select only one of these is required. Also, as discussed above, two lower ranked features may have a combined discriminatory power greater than that of a higher ranked feature. Kittler observes that this scheme, in general, does not lead to a good feature set [11]. He also suggests two sequential methods - *Sequential Forward Selection* **(SFS)** and *Sequential Backward Selection* **(SBS).**

SFS is a bottom-up procedure. Initially, the final feature set is empty. Let k be the number of features in the final set at any stage. The next feature added is the one for which the combined classification rate (in association with the k features already existing) is the highest. Conversely, **SBS** is a top-down procedure. Initially, all features are in the final feature set and at each step the worst feature is discarded. Computationally, **SFS** is less intensive as it will never require more than (n-1) dimensions for the classifier whereas **SBS** will involve n dimensions also. On the other hand, **SBS** also provides an estimation of the information lost as a particular feature is discarded. Both these methods are better than the **n best features** technique. Again, it has the drawback that once a feature is selected, it cannot be removed.

The **plus q - take away r** algorithm suggested by Kittler [11] tries to overcome this drawback. This involves the addition of q features (using **SFS**) and then the removal



Encircled numbers indicate a measure of Classification rate

Figure 12. Accelerated Search Tree

of r features one at a time (using **SBS**) at every stage. For q > r, it is a bottom-up approach and for q < r, it becomes a top-down approach. The algorithm now has the ability to remove any features from the final set. The problem is that as q increases, the complexity of the algorithm increases rapidly.

4.3 *The Proposed Feature Selector*

Before presenting the feature selector actually used, let us quickly review the problem. We have 64 coefficients (features) on the basis of which to classify an input pattern. Recall that we have 28 different classes. For each class, each of the 64 features has a certain range (caused by the variations discussed in section 1.3). Computations revealed that there was no single coefficient (henceforth, the words coefficients and features will be used interchangeably) for which the range of values for one class (say **a**) did not overlap with the range of values of at least one of the remaining 27 classes. In other words, there was no $[R_{i,j}]$ such that:

$$[R^{27}_{i,j}] \cap [R^{26}_{i,j}] \cap \cdots \cap [R^{1}_{i,j}] = \phi, \qquad i = 1,2,\ldots,8$$
$$j = 1,2,\ldots,8$$

where $[R^{a}_{i,j}]$ denotes the range of values for the i,j th coefficient for class a. This effectively meant that the feature selection procedure would have to identify combinations of features which would be able to dicriminate among the classes reliably. It should also be clear that the same set of features cannot be used to discriminate among all the classes. A feature that best discriminates between class **a** and class **b** (ranges wide apart) may provide no clu to whether the pattern in question belongs to class **a** or class **d** (ranges overlap completely).

Previous work on similar problems indicated that selecting features based on the standard deviations was a good choice. Huang and Chung used a similar technique for recognizing Chinese characters [12]. After computing the Walsh transform, for each group of characters, they chose the coefficient that had the *maximum separability power*. They then partitioned the group into subgroups choosing proper thresholds. If the subgroups overlapped, the process was repeated among the ones that overlapped. These features were used in an hierarchical manner for classifying the patterns. In this way, they were able to achieve a recognition rate of 99.5% (one failure out of 200). Wang and Shiau also concluded that ordering with respect to standard deviation is the best scheme for selecting features for recognizing Chinese characters [13]. Ahmed and Rao demonstrate that the variance criterion is effective in dimensionality reduction during feature selection [14].

Based on these results, it was decided that feature selection would be based on the standard deviation. Standard deviation is a measure of the spread of the values about the mean and thus gives a good measure of class separability also. Instead of using the standard deviations alone, the selection was based on the *figure-of-merit (fom)* criterion. A pseudocode for feature selection is given in Figure 13 on page 40. The *fom* of coefficient (i,j) between two classes **a** and **b** is defined as:

$$\mathrm{fom}_{\mathrm{a\,b}}(i,\ j) = 1 + \frac{\mathrm{d}_{\mathrm{a\,b}}(i,\ j)}{\sigma_{\mathrm{a}}(i,\ j) + \sigma_{\mathrm{b}}(i,\ j)}$$

where $\mathrm{d}_{\mathrm{a\,b}}(i,\ j)$ is the absolute difference between the two closest end-points of the ranges in question. If there is an overlap in the respective ranges, then the *fom* for that feature is set to zero.

```
For each class
    begin
        For each of the remaining 27 classes
            begin
                Calculate fom for all 64 coeffs
                Rank them in order of fom
                Pick up the first 8
            end
        set classlist to the 27 classes
        repeat
            Sort the features of classlist in order of the number of
                                                          occurrences
            With the top ranked feature do
                Include it in the final set of all classes in which it
                                                  appeared in the top 8
                remove all these classes from the classlist
            endwith
        until classlist is empty
    end
```

Figure 13. A pseudocode for feature selection

As mentioned in Section 3.2, a range of values was obtained for each of the 64 features for all the 28 classes. The ranges were calculated using two standard deviations around the respective means (If the patterns in the design set had coefficients having a normal distribution, this would ensure that 99.5% of the population would fall within this range. Later, the ranges were expanded by 10%, 20% and 30% to observe the effect on the selection and classification processes). For each class, the *fom*'s were calculated considering each of the remaining classes one by one. Next, for any class (say **a**), the *fom*'s that were greater than 2.0 were selected and ranked according to their values (Note that *fom*'s for all the 27 combinations with class **a** were considered). Next, the feature that occurred most often was placed in the final feature set. We now have a feature that can reliably discriminate between class **a** and all those classes for which this feature had a *fom* greater than 2.0. The features of the remaining classes are now considered and the feature that occurs most often is added to the final feature set. The process continues until we have found at least one feature for all the 27 combinations for class **a**. This process is repeated for all the 28 classes.

It was mentioned in the preceding paragraph that for each combination of classes, the features with a *fom* greater than 2.0 were considered. This was done to ensure that the features selected finally were able to discriminate reliably between the classes. If there is a combination of classes (say **a** and **b**) for which no feature meets this criterion, then the highest ranked feature should be included. Notice that if the *fom* is too small, then there is a possibility of misclassification. One way to avoid misclassification is to include the second highest ranked feature also (and more features, if deemed necessary) in the final feature list of either **a** or **b**.

5.0 CLASSIFICATION

The pattern classification problem has been treated extensively in the literature. A discussion of some of the popular techniques can be found in [1]. A number of interesting experiments and applications of classification procedures can be found in [15].

5.1 A Sequential Classifier

Recall the feature selection procedure employed for our problem. We obtained 28 sets of features - one for each class. The way the features were selected also dictated the choice of classifier. Since each feature in the feature set (for any class, say **a**) discriminates between class **a** and at least one other class, one possibility for the classifier design is a sequential design. The decision tree is given in Figure 14 on page 43 (a somewhat similar technique was employed by [12]). A pattern would be classified as belonging to class **a** if tests for **all** the features in the final set for class **a** would respond positively. Note that the classification would fail even if one test failed.

When the classifier was first tested on the design set, a modified version was implemented. As mentioned in the last paragraph of section 4.3, there exists a possibility of misclassification in the event of some feature in the final set having a low *fom*. To overcome this, the first implementation allowed the classification to continue even after

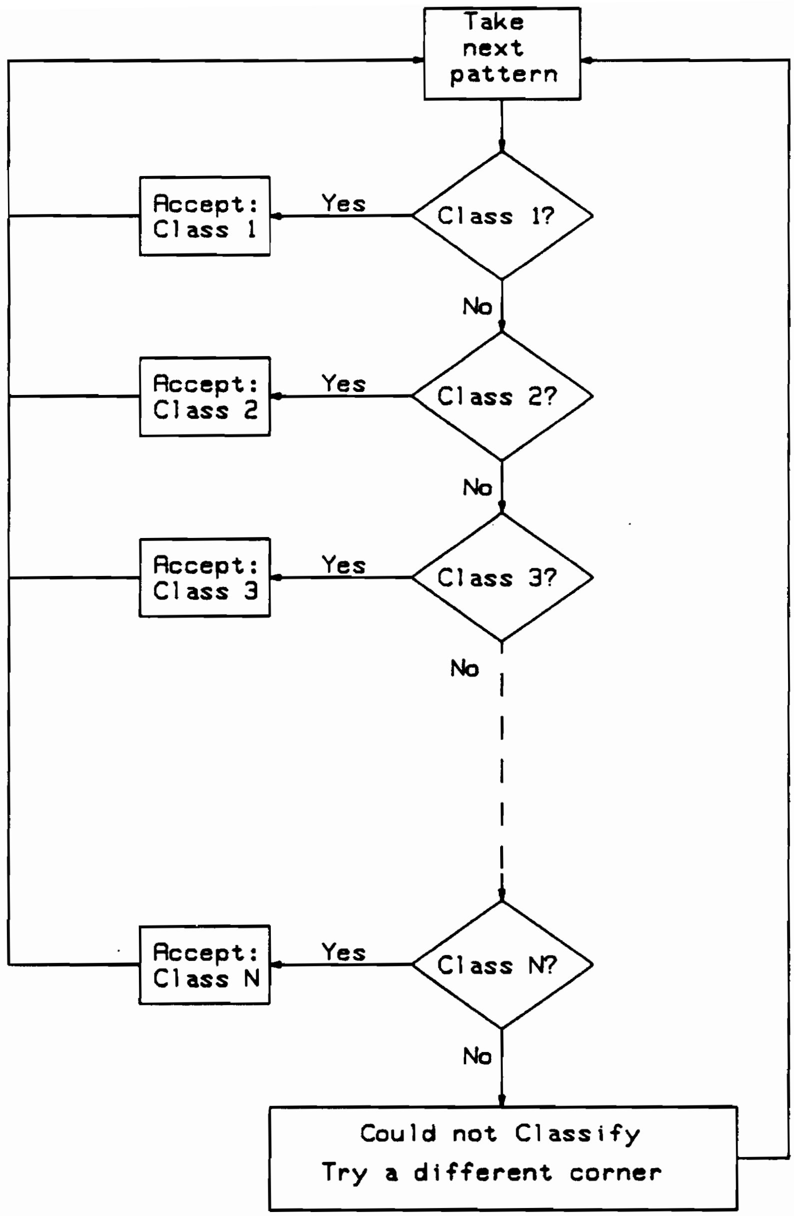


Figure 14. Decision tree for a sequential classifier

a classification was signalled. As a result, all misclassifications appeared as multiple classifications. After the entire design set had been classified, all the multiple classifications were inspected and an additional feature was added to the actual class - the feature selected was the one having the highest *fom* for the actual class and the misclassified class. After the first run, 3 patterns actually belonging to corner 1 of a 20 dollar bill were also classified as corner 2 of a 20 dollar bill [Figure 4 on page 8]. Thus, the feature that best discriminated between these two classes was added to the final feature list of corner 1 of 20 dollar bill. The next run did not report any misclassifications or multiple classifications. The classifier was now run on the test set and the results are mentioned in Chapter 6. Note that when the original classifier (without the extra feature) was run on the test set, all the misclassifications that occurred (3 in all) were of the same type - patterns actually belonging to corner 1 of 20 dollar bills were also classified as corner 2 of 20 dollar bills. But after the inclusion of the feature, **no misclassifications occurred.**

A few remarks about the classification scheme are in order here. The final feature set obtained for each class contained an average of three features. Note that while selecting the features, dimensionality reduction was the primary concern and an attempt was made to reduce the number of features in the final set. Consequently, a feature that would discriminate among a larger number of classes would be preferred over a feature that would distinguish among fewer classes (albeit more reliably). An attempt to select the best discriminating features for each combination presents certain problems. First, the number of features in the final feature set for each class becomes much higher (in some cases, more than 10) and the classifier becomes more complex. Secondly, the reject rate of the classifier increases substantially. This is an immediate consequence of the design of the classifier. The fewer the number of features in the final set, the higher is the misclassification rate; the higher the number of features in the final set, the higher the reject rate. Although one of the design objectives was to reduce the misclassification rate as much as possible, a very high reject rate is not permissible either. The scheme that was chosen not only provided a zero misclassification rate but also had a low reject rate (Refer to Chapter 6).

5.2 *A Hierarchical Classifier*

The sequential classifier was not the only classifier that was tried. A layered decision process is a powerful means of classification [16,17]. Nadler [18] proposes a hierarchical classifier which is briefly described below [Figure 15 on page 46]:

- At each level, the number of recognizers is the same as the number of classes.

- The decision process is constructed by the logical intersection of the classes $[D_n]$ accepted by the recognizers at level n with the result of the corresponding intersection at level $n-1$. Initially, at level zero, all the classes are accepted:

$$[D] = [D_n] \cap [D_{n-1}] \cap \cdots \cap [D_1]$$

- The decision process stops at level n if:

  the intersection contains one and only one accepted class signalling *recognition* (correct or incorrect), or

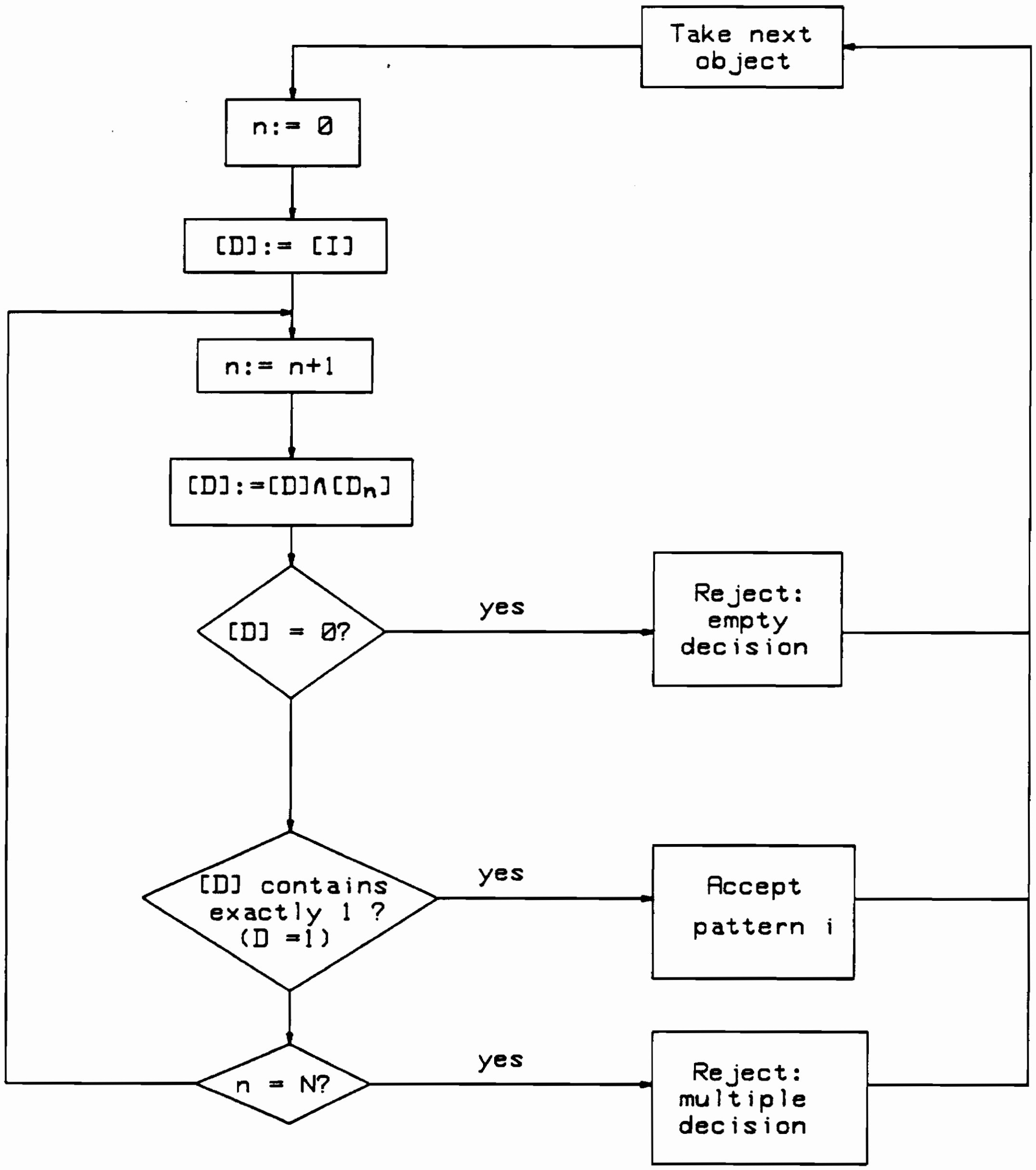


Figure 15. A Hierarchical classifier

the intersection contains no accepted classes i.e., it is empty. This signals a *reject* (empty decision).

- Otherwise, the decision process proceeds to level $n+1$ (if the intersection contains more than one accepted class).

- The process terminates at the highest level with a *reject* if the intersection still contains more than one accepted class.

When this scheme was implemented with the features selected as before, the reject rate was lowered, but there were some misclassifications too. The misclassifications occurred because the classifier rejected the actual (correct) class at the very first level but accepted some other class/classes (obviously incorrect) at this level. In subsequent levels, all except one among these were rejected resulting in a misclassification. All such cases would have resulted in a reject decision with the previous scheme. A simple way to avoid these misclassifications is to test for all the features in the final set of the class which is accepted by the classifier irrespective of the level at which the pattern is accepted. If any of the features do not satisfy the test, then the pattern will be rejected. Thus, the modified hierarchical classifier not only gave a zero misclassification rate but also lowered the reject rates compared to the sequantial classifier. The complete results are presented in Chapter 6.

It should be borne in mind that the ranges for the features were calculated using two standard deviations around the means (Section 3.3). In our recognition schemes we have employed very rigid criteria and that was one of the reasons that a zero misclassification rate could be achieved. However, it was also partially responsible for rejecting some patterns that would otherwise have been accepted.

6.0 RESULTS AND DISCUSSION

As mentioned in the last chapter, a number of schemes were employed for feature selection and classification. The results are summarized in

The first sequential scheme allowed the process to continue even after a classification (correct or incorrect) was signalled to check for any misclassification. As a result, all misclassifications appeared as multiple classifications. Specifically, 3 design set patterns corresponding to corner 1 of a 20 dollar bill were misclassified as corner 2 of a 20 dollar bill besides being correctly classiffied (as corner 1 of a 20 dollar bill). There was also one pattern from the test set that was similarly misclassified. It should be noted here that there were more rejects for patterns belonging to the test set than for patterns belonging to the design set. This is because the features are derived from the design set and hence the system is *tuned* to the patterns belonging to the design set. However, an interesting result was that there was no bill (even in the test set) for which all four patterns were rejected. This effectively means that there was no bill that could not be recognized by the system. The device will have a reject option and whenever the classifier rejects a pattern, the user will be prompted to try a different corner.

However, the misclassifications were removed entirely (from both the design set and the test set) after the inclusion of another feature. The process of inclusion of this feature and the reason for doing the same is explained in detail in section 5.2. When the ranges for all features were expanded (by 20%, 30% and 40%) to relax the hitherto rigid crite-

Table 1. Summary of Results

Classification Scheme	design Set			test Set		
	correctly classified	rejected	mis-classified	correctly classified	rejected	mis-classified
Sequential :						
1. Normal	315	18	3*	127	55	1*
2. Additional feature for 2011	315	18	-	127	55	-
3. All ranges expanded by 20%	327	6	-	139	43	-
4. All ranges expanded by 30%	329	4	-	148	34	-
Hierarchical :						
1. Normal	318	11	4+3*	139	31	12+1*
2. Additional feature for 2011	318	11	4	139	31	12
3. 10% expansion of ranges for features that caused the 4 misclassifications above	325	8	-	143	29	10
4. All ranges expanded by 20%	330	3	-	151	22	9
5. All ranges expanded by 30%	331	2	-	158	17	7
Modified Hierarchical :						
1. Normal	318	11	4+3*	139	43	1*
2. Additional feature for 2011	318	11	4	139	43	-
3. All ranges expanded by 10%	325	8	-	143	39	-
4. All ranges expanded by 20%	330	3	-	151	31	-
5. All ranges expanded by 30%	331	2	-	158	24	-
* denotes doubly classified						

ria, the classification rate improved for both the design and test sets. More importantly, no misclassifications were introduced.

When the hierarchical classifier was run, there were 4 patterns in the design set (corresponding to 4 different classes) that were misclassified besides the 3 patterns that were doubly classified (the latter were the same as the ones in scheme 1). Besides the inclusion of another feature for the latter category (as was done for scheme 1), the ranges for the 4 additional classes that were misclassified were increased by 10%. The result was that there were no misclassifications among the patterns belonging to the design set but there were still 10 misclassifications among the test set patterns. Moreover, the misclassified patterns did not belong to one or two classes; they belonged to 7 different classes. Even with the 20% and 30% range expansion, there was still a significant number of misclassifications. The reasons for these misclassifications are discussed in detail in Section 5.2. When the hierarchical classifier was modified as suggested in the same section, there were no misclassifications at all.

It should be noted here that the hierarchical classifiers gave significantly better reject rates as compared to the sequential classifiers. This is because the hierarchical classifier starts off with less stringent conditions initially and consequently, patterns that would normally be rejected by the sequential classifier are still retained by the classifier until a decision is reached at a higher level (see Section 5.2 for details). The complete results are presented in Table 1.

In conclusion, the *fom* based feature selection scheme and the modified hierarchical classifier gave very promising results. The misclassification rate was **zero** and for all those bills for which one or more corners were rejected, there was at least one corner (corresponding to the same bill) that was classified correctly by the above scheme. This implies that even if one or two corners of a particular bill are worn out or torn and are rejected by the classifier, the device can still recognize the bill if the other corners are in a good condition. Since the algorithm is very general, it can be applied to similar pattern recognition problems.

7.0 IMPLEMENTATION AND FUTURE SUGGESTIONS

The scheme can be easily implemented to fabricate a walkman - size, battery - powered device. The imaging device can be an optic RAM which serves the dual purpose of taking a binary image and storing it too. This image can then be processed to detect the edge and corner in the image and create a new image. The fast Walsh Transform can be implemented in hardware itself or can be computed by a microprocessor. The relevant features can be stored in the form of a look-up-table in a ROM and can be retrieved by the microprocessor for the purpose of classification. The result of the classification can be passed to the user through a voice-encoded message.

The major drawback of the current scheme is the process of edge and corner detection. The proposed Hough Transform technique requires a lot of time and does not detect the edges (and hence the corner) very accurately if they are not very sharp. A small error in the angle or intercept of an edge can cause a substantial error in the location of the corner. Since the location of the corner is very critical, some other technique that requires less time and is less susceptible to errors caused by the edges not being very sharp should be used. Also, the mechanical systems of the device should be reliable enough (from the point of view of the positioning of the corner under the imaging device).

BIBLIOGRAPHY

1. R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*, John Wiley and Sons, New York (1973).

2. J.R. Fram and E.S. Deutsch, "On the quantitative evaluation of edge detection schemes and their comparison with human performance," *IEEE Trans. on Computers*, vol. C-24, no. 6, pp. 616-628.

3. R.O. Duda and P.E. Hart, "Use of the Hough transform to detect lines and curves in pictures," *Comm. of the ACM*, vol. 15, no. 1, pp. 11-15, Jan. 1972.

4. H. Harmuth, *Transmission of Information by Orthogonal Functions*, Springer-Verlag, New York (1972).

5. H. Rademacher, "Einige Sätze von allegemeinen Orthogonalfunktionen," *Math. Annalen*, vol. 87, pp. 122-138, 1922.

6. J.L. Walsh, "A closed set of Orthogonal Functions," *American J. of Mathematics*, vol. 45, pp. 5-24, 1923.

7. R.E.A.C. Paley, "A remarkable series of Orthogonal Functions," *Proc. London Math. Soc.*, vol. 34, no. 2, pp. 241-279, 1932.

8. K. Fukunaga and W. Koontz, "Application of the Karhunen-Loeve Expansion to Feature Selection and Ordering," *IEEE Trans. on Computers*, vol. C-19, no. 4, pp. 311- 318, 1970.

9. T. Young, "Reliability of linear feature extractors," *IEEE Trans. on Computers*, vol. C-20, pp. 961-971, Sep. 1971.

10. B.R. Kowalski and C.F. Fender, "An Orthogonal Feature selection Method," *Pattern Recognition*, vol. 8, pp. 1-4, 1976.

11. J. Kittler, "Feature set search Algorithms," in *Pattern Recognition and Signal Processing*, ed. C.H. Chen, Sijthoff and Noordhoff, Holland, pp. 41-60, 1974.

12. J.S. Huang and M.L. Chung, "Separating similar complex Chinese characters by Walsh transform," *Pattern Recognition*, vol. 20, no. 4, pp. 425-428, 1987.

13. P.P. Wang and R.C.C. Shiau, "Machine recognition of printed Chinese characters via Transformation Algorithms," *Proc. of the 1972 Conf. on Decision and Control*, pp. 532-536, 1972.

14. N. Ahmed and K.R. Rao, *Orthogonal Transforms for Digital Signal Processing*, Springer-Verlag, New York (1975).

15. L.N. Kanal, ed., *Pattern Recognition*, Proc. of the IEEE Workshop on Pattern Recognition, Dorado, Puerto Rico, Oct. 1986.

16. L.A. Tel'ksnis, "Multilevel recognition system," *Avtomat. Telemekh*, vol. 26, pp. 2170-2175, Dec. 1965.

17. K.S. Fu, *Sequential Methods in Pattern Recognition and Machine Learning*, Academic Press, New York (1968).

18. M. Nadler, "Error and reject rates in a hierarchical pattern recognizer," *IEEE Trans. on Computers*, vol. C-20, no. 12, pp. 1598-1601, Dec. 1971.

Appendix A.

Figure A.1 Variations in the margins

Figure A.2 Skew in the edges

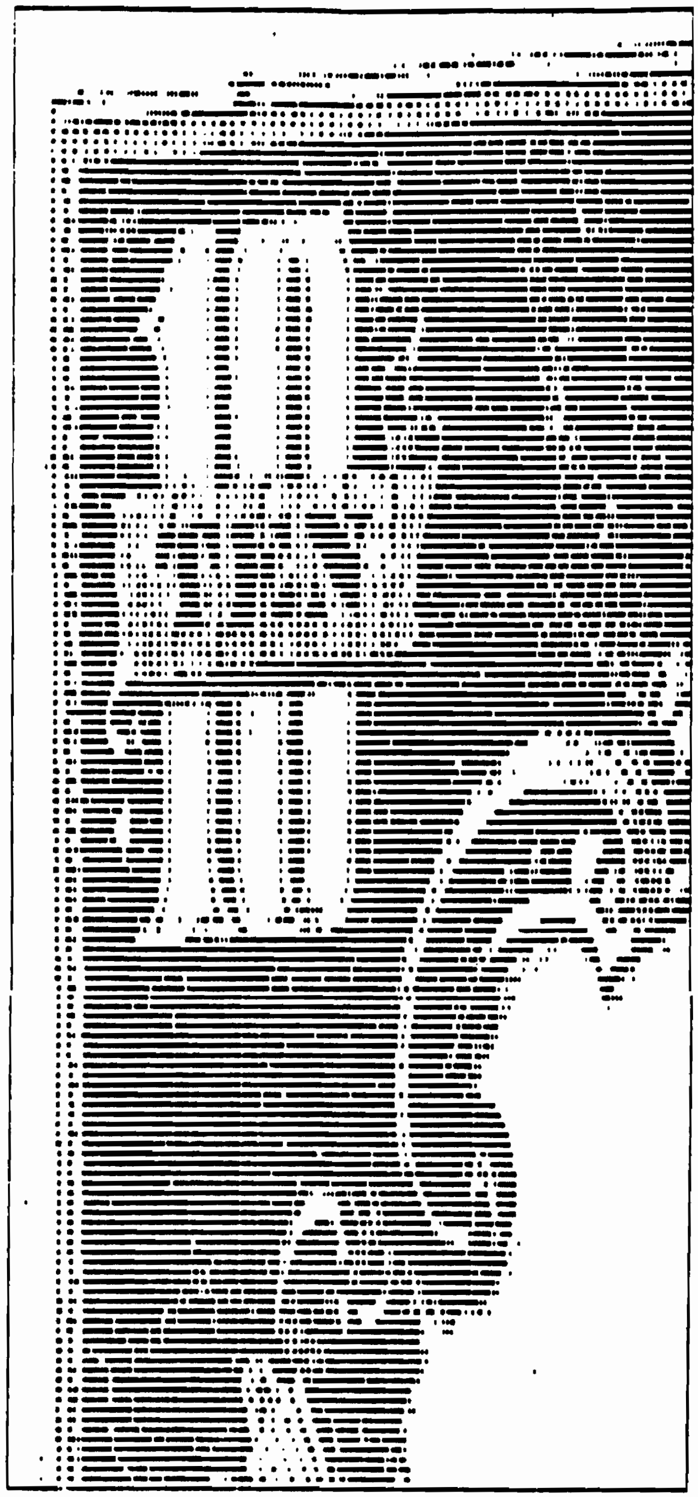


Figure A.3 The original image (320 x 300 pixels)

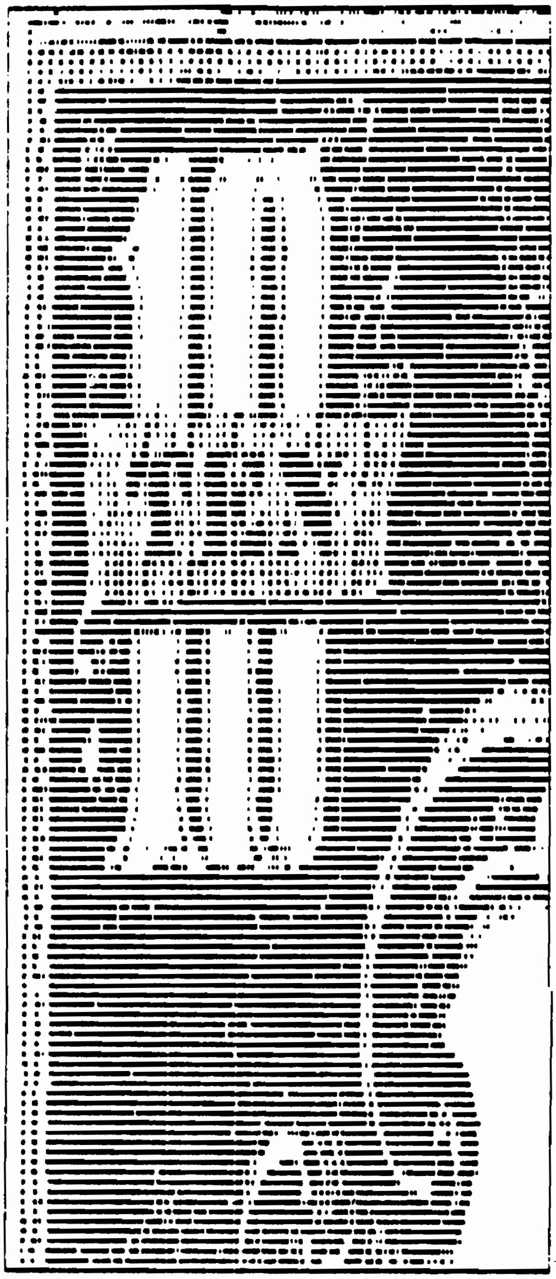


Figure A.4 Image after skew correction (256 x 256 pixels)

Vita

Anupam Singh was born in India in 1962. He completed his B.E. (Hons.) in Electrical and Electronics Engineering from Birla Institute of Technology and Science, Pilani, India in Dec. 1983. He was with the CMC. Ltd., N.Delhi, India from Feb. 1984 to July 1985. He is currently a graduate student at Virginia Polytechnic Institute and State University pursuing his M.S. in Electrical Engineering. His research interests include Pattern recognition, Image processing and Microprocessor based devices. He is a member of the IEEE Computer Society.

Anupam Singh