1998

Deviating Angular Feature for Image Recognition System Using the Improved Neural Network Classifier.

Adisak Srinakarin
Louisiana State University and Agricultural & Mechanical College

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

DEVIATING ANGULAR FEATURE FOR IMAGE RECOGNITION SYSTEM USING THE IMPROVED NEURAL NETWORK CLASSIFIER

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical Collage
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The Department of Computer Science

by
Adisak Srinakarin
B.E., Chulalongkorn University, Thailand, 1983
M.SC., Southern University, 1988
December 1998

To My Parents

Acknowledgments

First of all, I would like to thank my major professor, Prof. J. Bush Jones for his great kindness, encouragements, and advices. I also thank Prof. Subhash C. Kak, for his invaluable guidances in research direction. I am indeed grateful to both of them for their dedication and commitment to research.

I would like to thank the members of my doctoral committee, Prof. Doris Carver, Prof. Donald H. Kraft, and Prof. James J. Madden for their comments and suggestions.

I thank the department of Computer Science for providing me the assistantship for five years. To all students volunteers for their donations of handwritten numerals used in this research.

I also would like to express my sincere appreciation to all teachers and professors who taught me in every schools that I have attended

Special thanks to my wonderful friends and colleagues: Elias Khalaf, Dr. Deky Gouw, Dr. Choong-Gyoo Lim, Yoo-Han Hwang, and Kasidit Chanchio for their helps and joys.

Most of all, I am very grateful to my parents for giving me their supports, teachings, hopes, and dreams. Without their loves and guidances, my educational career could not have been this far.

Contents

List of Figures

Abstract

The ability to reconize images makes it possible to abstractly conceptualize the world. Many in the field of machine learning have attempted to invent an image recognition system with the recognition capabilities of a human. This dissertation presents a method of modifications to existent image recognition systems, which greatly improves the efficiency of existing data imaging methods.

This modification, the Deviating Angular Feature(DAF), has two obvious applications: 1) the recognition of handwritten numerals; and 2) the automatic identification of aircraft. Modifications of feature extraction and classification processes of current image recognition systems can leads to the systemic enhancement of data imaging. This research proposes a customized blend of image curvature extraction algorithms and the neural network classifiers trained by the Epoch Gradual Increase in Accuracy(EGIA) training algorithm. Using the DAF, the recognition of handwritten numerals and the automatic identification of aircraft have been improved.

According to the preliminary results, the recognition system achieved an accuracy rate of 98.7% when applied to handwritten numeral recognition. When applied to automatic aircraft identification, the system achieved a 100% rate of recognition. The novel design of the prototype is quite flexible; thus, the system is easy to maintain, modify, and distribute.

Chapter 1
Introduction

The ability to recognize images makes it possible to abstractly conceptualize the world. This ability is the result of intelligent actions from our complicated brain functions, which are able to process large amounts of information from the surrounding dynamic environments and then respond properly to the demanding events. The human image recognition system is still largely unknown, even though fundamental mechanisms of the system have recently been revealed. Many in the field of machine learning have attempted to invent an image recognition system with the recognition capabilities of human brain. This dissertation will propose a way to modify existing image recognition systems, which could greatly improve the efficiency of existing data imaging methods. The work presented here is a result of study of the human image recognition system and its possible applications to improved mechanical image recognition.

The image recognition system usually consists of three processing steps: preprocessing, feature extraction and classification(see figure 1.1). Any improvement in understanding one or more of these operational processes can directly enhance the performance of the entire recognition system. Two image recognition applications are very important and widely used: 1) handwritten numeral recognition system; and 2) automatic aircraft identification system. These two applications are the main focus of our research.

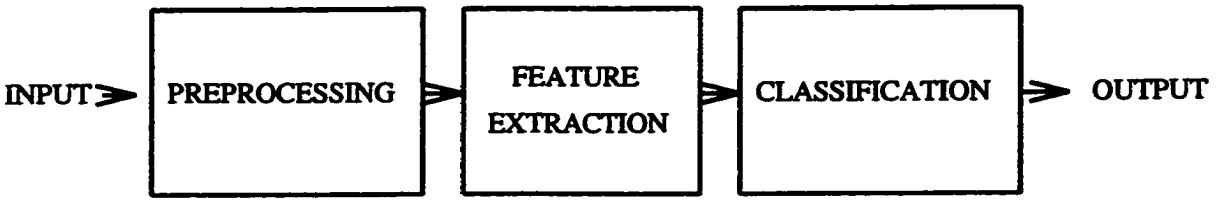


Figure 1.1: The three operational stages of image recognition system.

Generally speaking, there are two models of handwritten numeral recognition applications: the online handwritten numeral recognition system receiving numeral curvatures directly from a pressure-sensitive writing pad, and the offline handwritten numeral recognition system processing the image files of handwritten numerals. The handwritten numeral recognition system under study here is the offline system. The problems of the offline system are more challenging than the problems of online system, since the online numeral recognition system directly interacts with its user through the writing pad; this interaction causes it to process the sequence of curvature coordinates and transform them to the unique features of the written numbers. These unique features make the classification process quite simple, although the system does not require very high accuracy in recognition, since the user can neatly rewrite the same number in order to get the correct recognition from the system. Unlike the online recognition system, building the offline handwritten numeral recognition system is more problematic, since the numeral's feature cannot be easily generated from its image, which contains no information about pen movement and the beginning or ending of the image's curvature. Furthermore, the accuracy of an offline recognition system is very important to the system design considerations because the offline system usually processes the data batch with fewer human interventions. Therefore

any technique which can improve the performance of an offline handwritten numeral recognition system is extremely valuable. In order to improve the performance of an offline handwritten numeral recognition, we will propose a sophisticated technique for feature extraction and classification to be combined with one of the more effective designs so that it can recognize numeric images which have usually been imprecise and complex at the higher accuracy rate.

The second application, the automatic aircraft identification system, is also an interesting recognition system that has been studied by many researchers and for which a number of workable designs have been introduced[3, 8, 18]. Unlike the technical difficulties posed by the handwritten numeral recognition, the biggest problems of automatic aircraft identification system is the variety of rotation variance in images. To improve the performance of the identification system, we have designed new techniques for feature extraction which can generate the signatures of aircraft images which are quite invariant.

This research will concentrate on improvements to the feature extraction process and the classification process. The justification for this focus is that these two processes of the image recognition system capture the common characteristics of the image patterns and utilize them for recognition. The proposed techniques presented in these processes should be robust, efficient and capable of generalization at high accuracy.

Despite the development of handwritten numeral recognition systems and automatic aircraft identification techniques in the commercial and military sectors[2, 6, 15, 19, 21, 39], further improvements and simplifications for these two types of image

recognition systems can provide us with recognition systems that perform at higher accuracy with lower cost of implementation and a faster response. We will combine the original techniques for image curvature extraction with a new feature representation which can be used to enhance the system performances. In addition, we will propose an training algorithm for multilayer feedforward neural network, which should both speed up the training process, and solve the incomplete learning problems which sometimes occurs in the neural network classifiers.

1.1 The Objective

As the result of investigations of previous designs of handwritten numeral recognition systems, automatic aircraft identification system and neural network training algorithms for multilayer feedforward neural networks, we will first present new techniques to enhance the quality of the feature extraction process with a new feature representation and original algorithms for image curvature extraction. Then we will introduce new training algorithm to speed up the training process and increase the performance of neural network classifiers. Finally, we will demonstrate that incorporating these innovative approaches into the feature extraction process and classification process can enhance the performance of the handwritten numeral recognition system and the automatic aircraft identification system.

1.2 The Approaches

Since our objective is to design the highest performance image recognition systems possible and to test the systems for the comparison results, we divided our works for this research goal into three parts. The presentation of each part contains the informa-

tion of practical solutions, testing and analysises which support our new approaches described briefly in the following.

1.2.1 The Improvement of Feature Extraction Process

Feature extraction techniques and feature representations are very crucial to the image recognition systems. In facts, the feature quality is one of the direct factors which determine the system performance, since the image feature extraction process provides the data from which the classification process performs the image recognition. There is strong evidence to indicate that the image captured by human eyes is processed and encoded into some forms of complex information signals before the encoded information is transmitted to the brain cortex (where human recognition is believed to evolve). A good feature representation should possess the high qualities of data compression, invariance and fidelity [22]. Data compression of a feature indicates that the amount of information constructing the feature must be more concise than the raw image data; the invariance quality of a feature indicates that the conceptual signature of the image should not subject to change due to the imprecise appearance and the vagueness of images; and the fidelity quality of a feature indicates that the valuable information from the original image data must be maintained and enhanced in the feature. Frequently some other qualities of a feature representation, such as simplicity and flexibility, are also considered to be important in the design of any good feature representations because these qualities also determine the efficiency and complexity of the feature extraction algorithms. In the current research, the original *Deviating Angular Feature* and its image curvature extraction algorithms (which have the above qualities) will be described. The significances of these approaches to

the two image recognition applications can obviously be shown by the performance comparison tests we conduct on our challenging database.

1.2.2 The Improvement of Classification Process

It has been more than four decades since the first neural network, the Mark I Perceptron, was successfully built for a pattern recognition systems[10]. Several new models of neural networks and training algorithms were invented later to help solve the problems of image classification. Currently some advanced neural network classifiers can be trained from the train image data and will produce a very high accuracy of generalization of the unseen test images. Among many neural network structures invented, a committee of multilayer feedforward neural networks is frequently used in many image recognition systems, since it has a high capacity of information storage[13], and yet the structure of neural network committee allows both training and testing algorithms to be executed in parallel effortlessly. For this reason, the committee of multilayer feedforward neural networks is used as the classifiers in our experimental systems in which the classifier is trained by our new training algorithm, the EGIA which can train the neural network classifier much faster than *Backpropagation with momentum weight change* and *Gradual Increase in Accuracy*[20]. The new training algorithm also guarantees a complete learning of the neural networks for a large number of the train data with high dimension inputs.

Since the EGIA training algorithm provides advantages over some standard training algorithms, we test a neural network on two types of problems using 1) the Backpropagation with momentum weight change training algorithm; 2) the Gradual Increase in Accuracy training algorithm; and 3) Epoch Gradual Increase in Accuracy

training algorithm in order to get the comparison test results. The test results we obtain substantiate our claims of the advantages from Epoch Gradual Increase in Accuracy training algorithm.

1.2.3 The Experimental Prototypes

The studies in obtaining the highly accurate handwritten numeral recognition systems and automatic aircraft identification systems have been the focus of many researchers for decades[31]. Despite the long history of this type of research, the impetus to perfect the qualities of these two image recognition systems remains because these two automatic systems are vital to the information industry, in which millions of checks, credit card receipts, post office mails and satellite images are processed everydays.

In this study, we first examine a prototype of the handwritten numeral recognition systems in which the feature extraction unit generates deviating angular features and the Kirsch feature from segmented handwritten numeral images. The classification unit of the system is a committee of neural networks trained by Epoch Gradual Increase in Accuracy training algorithm.

Our second system prototype is the automatic aircraft identification system; the current project, strives to perfect its qualities by utilizing only the deviating angular feature and the new image curvature extraction algorithm for the feature extraction process. A committee of neural network classifiers trained by Epoch Gradual Increase in Accuracy for the classification process of the second prototype system

1.3 Layout

This dissertation is organized in the way that the new ideas and algorithms will be firstly described then we present the two complete image recognition systems using the new approaches.

Chapter 2 reviews the related works that paved way to our new approaches for feature extraction and neural network classifier.

Chapter 3 fully defines the deviating angular feature and its theories.

Chapter 4 explains two original algorithms for extracting the image curvatures: the first algorithm is for handwritten numeral images, and the second is for aircraft silhouettes. The pseudo codes for these algorithm are also presented in this chapter.

Chapter 5 presents the new training algorithm for multilayer feedforward neural networks called *Epoch Gradual Increase in Accuracy*, the performance comparison tests and statistics.

Chapter 6 introduces the new design of handwritten numeral recognition system using *Deviating Angular Feature* and *Epoch Gradual Increase in Accuracy* training algorithm for neural network classifier. The experimental results from our realistic image database are also shown.

Chapter 7 introduces the new design of automatic aircraft identification system. The experimental results are also reported.

Chapter 8 concludes our works of this thesis.

Chapter 2
The Related Works

An automatic image recognition system is a mixture of specialization and adaptive mechanisms capable of identifying complex and imprecise representation of image objects. This computing problem has been challenging to scientists for decades. The work of many scientists since the dawn of computer era have paved the way for new research to improve or correct the problems of the previous techniques. This chapter describes the relevant research in feature representation, feature extraction, and neural network classification. These works have served as the inspiration for new techniques which can be used in offline automatic image recognition systems.

2.1 The Curvature Features

The fundamental signatures of image objects are the curvature profiles', the features derived from the image curvatures usually inherit the unique information of the images which can be used in a recognition system. Consequently numerous research describes feature representations and the feature extraction techniques for obtaining the image curvatures. In 1991 a distance curvature feature for online character recognition system was introduced[12]. In this system, the features from the output of electronic pen were divided into three qualitative sequences: 1) the distance from horizontal reference; 2) the distance from vertical reference; and 3) the vertical distance from the starting point(see figure 2.1).

Information about image curvatures is a valuable feature that can be used effectively in character recognition. However, this feature had disadvantages for our

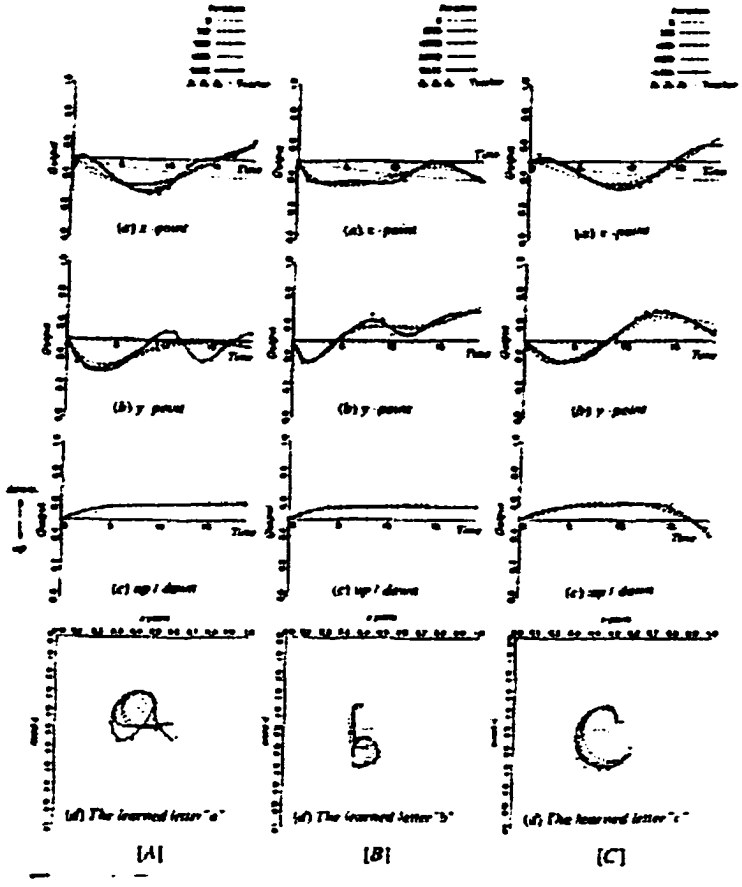


Figure 2.1: Curvature Distances

problems, since it can be extracted only from the electronic pen(online) and it can be highly sensitive to the transition, rotation, and scaling variances. The feature representation and feature extraction method from this experiment could not diminish the nuances of information about the images of the same category; therefore, the system classifier had to work very hard to map the abstracts of the images and the categories they belonged to. The burden this posed to the classifier was so heavy that the performance the overall recognition system was compromised.

The feature representing the character curvature can be something other than sequence of distances between pixel points and a reference point(for instance, a sequence of slopes at pixel points along the image curvature). In an attempt to create a feature having translation invariances(see figure 2.2 and figure 2.3) C. C. Tappert introduced a feature representation in 1982 which consists of a pair of slope and vertical distance over a pixel point [34]. With this approach, the scaling invariances of feature could additionally come from the character normalization. This feature was used in the preliminary experiment successfully and yielded over 95 % accuracy in generalization.

Tappert's approach of using the slope or angle sequence for feature representation illustrates that angular qualitatives can be an significant feature for image objects. Even though this feature representation was not invariant in scaling and translation in itself(this system depended on the normalizing process in solving the data invariances), obviously the feature is not invariant in rotation; Tappert's work inspires us to look closer in new design of angular features.

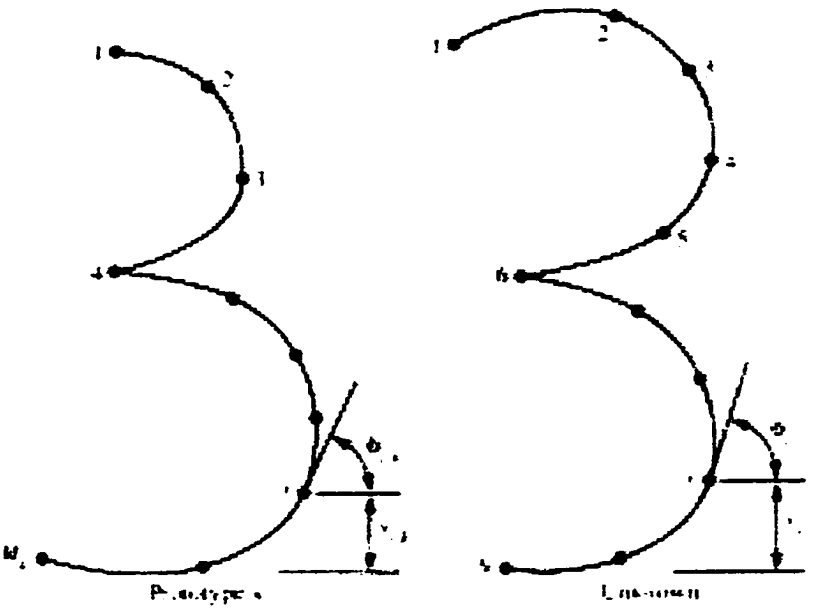


Figure 2.2: Tappert's Curvatures

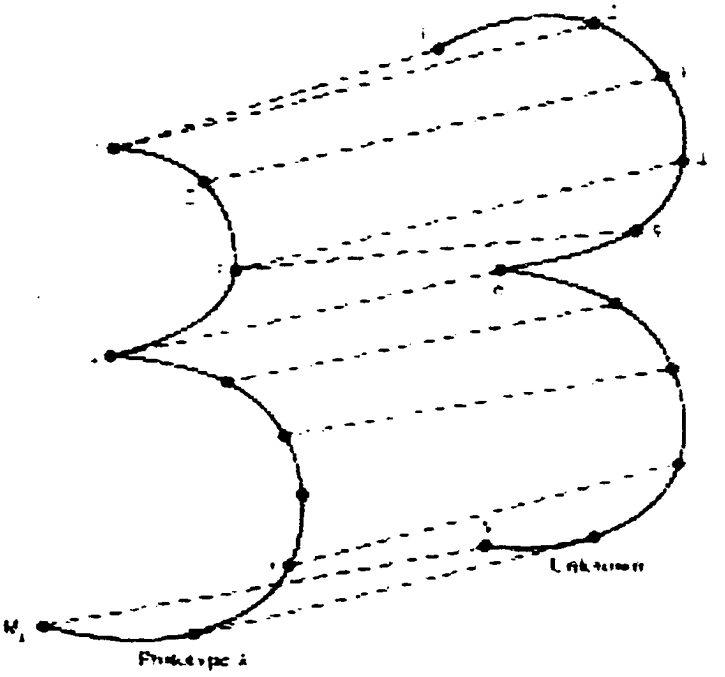


Figure 2.3: Tappert's Curvature

Continuing in this direction, a research work recently presented by Sung-Bai Cho introduces the idea of using directional filters, known as a Kirsch mask, to extract the features of binary bitmap images[6]. The results from feature extraction are the rational numbers which indicate the degree of the image curvature slope at particular pixel points. Figure 2.5 shows the Kirsch masks used by Cho in extracting four directional features: 1) horizontal feature; 2) vertical feature; 3) right-diagonal feature; and 4) left-diagonal feature. The features can be produced through applying the following functions to each pixels on the binary bitmap image.

$$G(i,j)_h = \max(|5S_0 - 3T_0|, |5S_4 - 3T_4|) \quad (2.1)$$

$$G(i,j)_v = \max(|5S_2 - 3T_2|, |5S_6 - 3T_6|) \quad (2.2)$$

$$G(i,j)_r = \max(|5S_1 - 3T_1|, |5S_5 - 3T_5|) \quad (2.3)$$

$$G(i,j)_l = \max(|5S_3 - 3T_3|, |5S_7 - 3T_7|) \quad (2.4)$$

where $G(i,j)$ is the feature value of image pixel(i,j), S_k and T_k are the neighbor pixel around pixel(i,j) defined as

$$S_k = A_k + A_{k+1} + A_{k+2} \quad (2.5)$$

$$T_k = A_{k+3} + A_{k+4} + A_{k+5} + A_{k+6} + A_{k+7} \quad (2.6)$$

A_k, A_{k+1},..., A_{k+7} are defined systematically in figure 2.4. The feature extraction using Kirsch mask on a handwritten numeral is illustrated in figure2.6.

Kirsch feature extraction is clearly an efficient technique; it can also extract the inner and outer curvature features of image objects. However this approach has some drawbacks; it is not invariant in translation, scaling, or rotation. The first

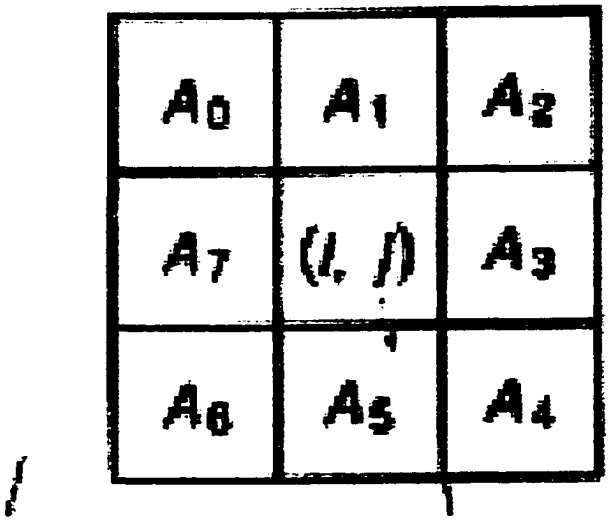


Figure 2.4: Kirsch Filter

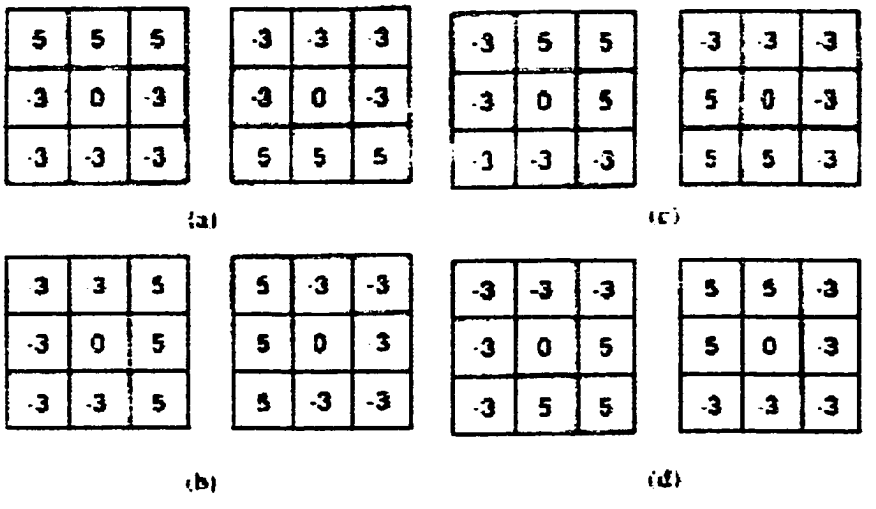


Figure 2.5: All Kirsch Filters used in Cho's experiment

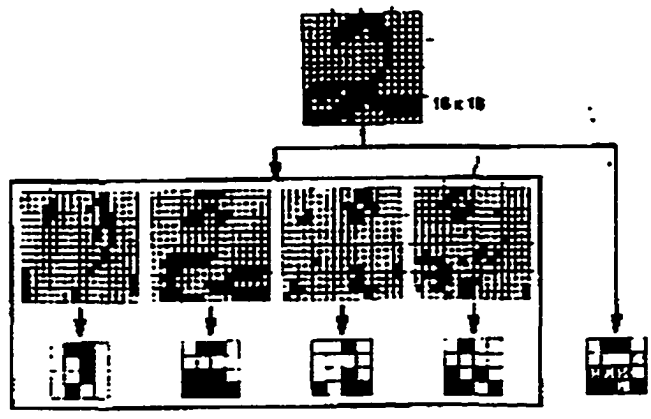


Figure 2.6: The example of Kirsch feature of an image

two invariances are solved by normalizing and centerizing the image, but so far no effective technique is available to fix the problem of rotation variance for Kirsch feature extraction. In 1997, Cho Sugn-Bae published his work using four directional Kirsch features in handwritten numeral recognition system utilizing a neural network classifier; the system achieves a 97 % recognition rate, despite the fact that he does not use the most advanced training algorithm for neural network classifier.

When the image objects are silhouettes, the distance from a unique reference point in the image frame to points on the image curvature is considered to be a good feature. A research group of Dae-Young Kim, Sung-Il Chien, and Hyun Son[18] invented a feature which is a sequence of distances between the image centroid and pixel points on the curvature of aircraft silhouette(see figure 2.7, figure 2.8, and figure 2.9). The group claims that the feature is translation-invariant, scaling-invariant(after normalization), and rotation-invariant(if the sequence starts from the biggest distance). We think that the claims for translation-invariance and scaling-invariance are reasonable; however the rotation-invariance of the feature is doubtful, since the uniform shapes of aircrafts

may have more than one largest distance from the centroid to pixel points on the image curvature. Despite its feature weakness in rotation-invariance, this feature representation is unique, and the work inspired the design of the feature which may be completely invariant in translation, scaling, and rotation. In 1991, the prototype system for aircraft identification introduced by this research group achieved a 97 % recognition rate.

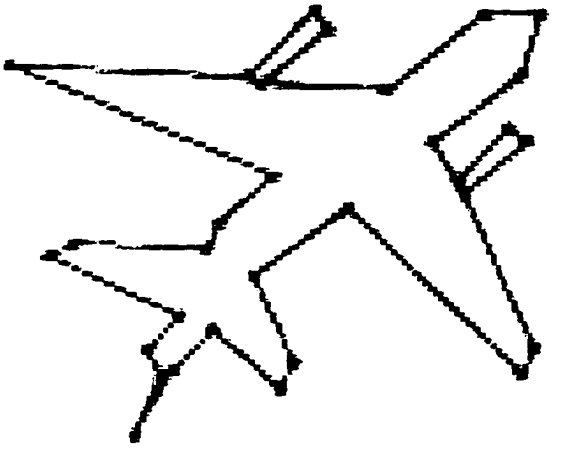


Figure 2.7: Aircraft Curvature

2.2 Classifications

Insofar as neural networks are concerned, multilayer feedforward neural networks have been used successfully in function approximation and classification problems. The standard training algorithm for this type of network structure is backpropagation with a momentum weight change. Despite successes in solving several design problems, the backpropagation with a momentum weight change has serious disadvantages in slow training. For instance, in consideration of the high dimensional classification problem,

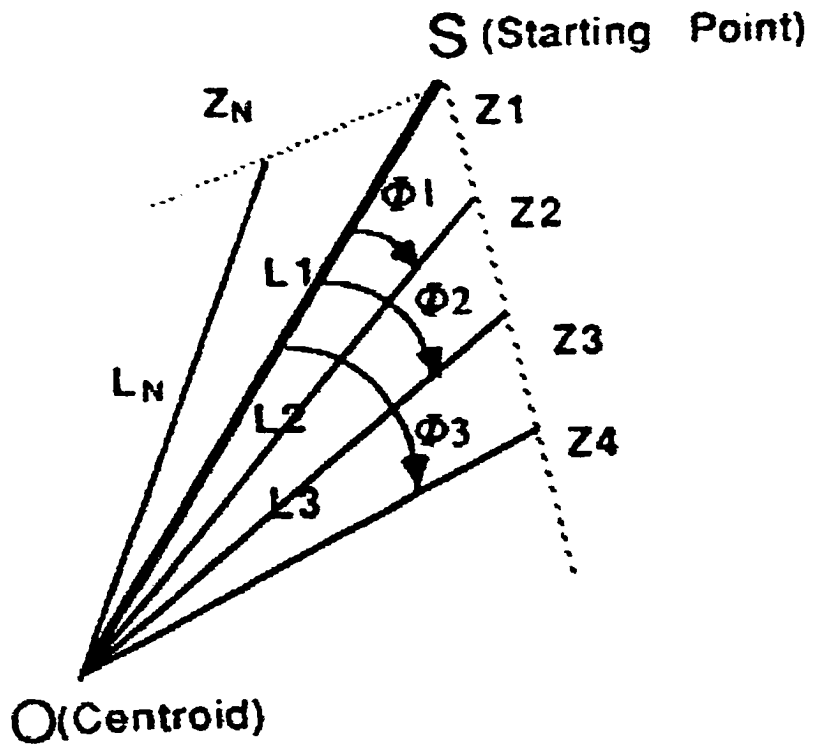


Figure 2.8: Composite of distances and angles for features

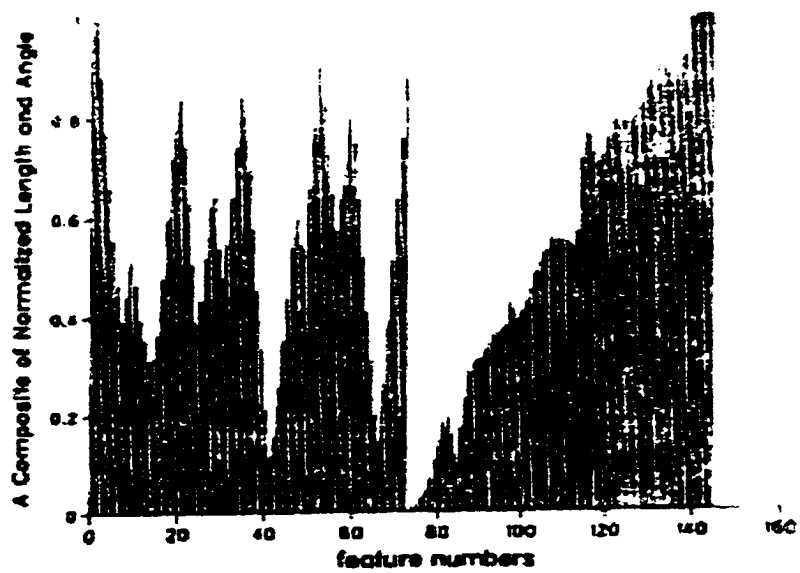


Figure 2.9: Feature sequences

Problem (network size)	BP	BPGIA	BP + Momentum	BPGIA + Momentum
1's complement (4x6x4)	6.2 (256874)	2.74 (104525)	2.0 (64936)	1.0 (30006)
Rotate register (3x6x3)	5.7 (72477)	1.5 (18727)	1.7 (15927)	1.0 (9613)
3 to 8 decoder (3x8x8)	5.3 (347587)	3.9 (252082)	1.2 (61366)	1.0 (51677)
EXOR (2x2x1)	2.8 (161396)	1.9 (107634)	1.6 (80321)	1.0 (48032)
Differentiation between a square,circle and triangle (16x16x1)	4.2 (34417)	3.1 (18235)	5.7 (36594)	1.0 (6024)

Figure 2.10: The experiment results

the backpropagation sometimes demonstrates incomplete learning, which happens when a neural network resists to learn few train patterns; this results in an extremely slow training process. This drawback in classification is very serious, because it guarantees the failure of any test patterns that are similar to the unlearned train patterns; this in turn reduces the generalization performance of the neural network classifier.

One practical solution to these problems was proposed by Ravi Kothari, Powsiri Klinkrachorn and Roy S. Nutter[20], in a technique called Gradual Increase in Accuracy(GIA). GIA is basically the rule which causes the backpropagation training algorithm to train the neural network using only one train pattern in every epoch. Each train patterns is used for training until the neural network reaches an output within $\pm\epsilon$(ϵ is initially large) of the target. After outputs from all train patterns are within $\pm\epsilon$ of their targets, ϵ is gradually decreased and the training process restarts until all outputs of train patterns are eventually within the updated $\pm\epsilon$. The whole process of training is repeated until ϵ is equal to a very small percentage of target outputs.

The experimental results using this new approach on four classification problems are shown on the table in figure2.10. The learning rate was set to 0.3 and the momentum was set to 0.9(3 times of the learning rate) for all experiments except the EXOR problem where the momentum rate was set to 0.5.

The approach of using GIA to accelerate the backpropagation training algorithm with momentum weight change is effective and has less overhead because it requires no additional complex computations which usually require significant computing time

to select of train patterns. Although the speed improvement of GIA can be up to five times faster than the backpropagation training algorithm, training one pattern repeatedly until its output is in the range of $\pm\epsilon$ of the target can cause unlearning of previously learned patterns. This in turn results in temporal instability in the training process. The solution for this problem will soon be introduced in this dissertation.

2.3 Summary

Significant in the field of feature representation indicates that angular features of the image curvature are translation and scaling invariances and can be used in many image recognition systems successfully. Additionally, the GIA approach suggests that the backpropagation training algorithm can be accelerated by the selective training on a single train pattern which has more error than the preset limit.

Chapter 3
Deviating Angular Feature

This chapter will discuss the representation of the deviating angular feature and the theories which explain its properties. In general, the deviating angular feature is a sequence of angles ranging between $-\pi$ and $+\pi$. It has the outstanding properties of translation invariance, rotation invariance, and scaling invariance. With appropriate techniques for image curvature extraction, the deviating angular feature can significantly improve the performance of many image recognition applications including the handwritten numeral recognition system and the automatic aircraft identification system.

To capture the hierarchical structure of deviating angular feature, we will first describe the basic element of the feature called deviating angle. The deviating angular feature, which is the sequence of deviating angles, will also be described.

3.1 The Description of Deviating Angle

The deviating angle is defined as the directional measurement of angular change between two vectors that concatenates at a single point on a plane in the rectangular coordinate system, and in which the magnitude of the deviating angle is less than or equal to π.

Suppose a, b, and c are points on a plane in rectangular coordinate system, and the deviating angle from vector $\vec{v}_{ab}$ to vector $\vec{v}_{bc}$ is the angle indicating how much the course of vector $\vec{v}_{bc}$ moves away from the course of vector $\vec{v}_{ab}$. The deviating angle

is positive if the angular excursion between two vectors swerves in counter clockwise, and it is negative if the excursion swerves in the opposite direction.

Definition 3.1 Let a, b and c be any points on a plane in rectangular coordinate system; $\vec{v}_{ab}$ is a vector from a to b, and $\vec{v}_{bc}$ is a vector from b to c. Let $\vec{k}$ be the unit vector perpendicular to the plane of $\vec{v}_{ab}$ and $\vec{v}_{bc}$ that points to the direction agreeable to a right-handed system of $\vec{v}_{ab}$, $\vec{v}_{bc}$, and $\vec{k}$. The deviating angle from $\vec{v}_{ab}$ to $\vec{v}_{bc}$ is defined to be

$$\mathcal{A} = \left\{ \frac{(\vec{v}_{ab} \times \vec{v}_{bc}) \bullet \vec{k}}{|\vec{v}_{ab} \times \vec{v}_{bc}|} \right\} \cos^{-1} \left\{ \frac{\vec{v}_{ab} \bullet \vec{v}_{bc}}{|\vec{v}_{ab}|\,|\vec{v}_{bc}|} \right\} \tag{3.1}$$

for which $-\pi < \mathcal{A} \leq +\pi$.

In figure 3.1, let a, b, and c be points on the a plane of rectangular coordinate system; $\vec{v}_{ab}$ is a vector from a to b and $\vec{v}_{bc}$ is a vector from b to c. Therefore $\mathcal{A}$ is the deviating angle from $\vec{v}_{ab}$ to $\vec{v}_{bc}$.

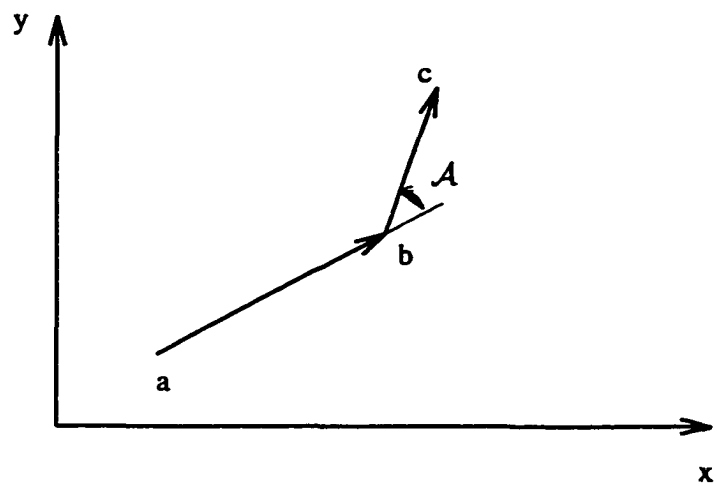


Figure 3.1: The deviating angles from $\vec{v}_{ab}$ to $\vec{v}_{bc}$.

Theorem 3.2 If a, b, and c are the points on a plan in rectangular coordinate system, then the deviating angle from $\vec{v}_{ab}$ to $\vec{v}_{bc}$ is translation invariance.

Proof: Let the coordinates of points a, b, and c on a plane in the rectangular coordinate system be (x_a, y_a), (x_b, y_b), and (x_c, y_c). And let the coordinates of the same a, b, and c on a plane in the new rectangular coordinate system which has its origin located at (x_o, y_o) relative to the old coordinate system be $(x_{a'}, y_{a'})$, $(x_{b'}, y_{b'})$, and $(x_{c'}, y_{c'})$(see figure 3.2). We are proving that the deviating angles from $\vec{v}_{ab}$ to $\vec{v}_{bc}$ are the same on both rectangular coordinate systems.

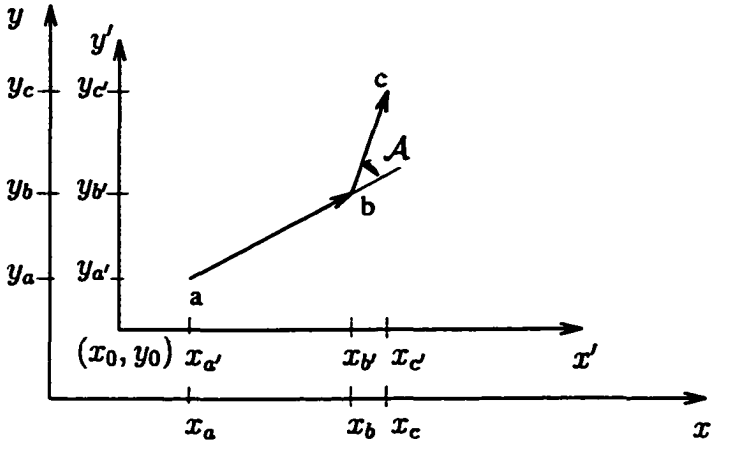


Figure 3.2: The coordinate translation of point a, b, and c.

Suppose $\vec{v}_{ab} = (x_b - x_a, y_b - y_a)$, $\vec{v}_{bc} = (x_c - x_b, y_c - y_b)$ and $\vec{v}_{a'b'} = (x_{b'} - x_{a'}, y_{b'} - y_{a'})$, $\vec{v}_{b'c'} = (x_{c'} - x_{b'}, y_{c'} - y_{b'})$.

From the transformation of coordinates involving pure translation[30],

$$x_{a'} = x_a - x_o, \quad y_{a'} = y_a - y_o \tag{3.2}$$

$$x_{b'} = x_b - x_o, \quad y_{b'} = y_b - y_o \tag{3.3}$$

$$x_{c'} = x_c - x_o, \quad y_{c'} = y_c - y_o \tag{3.4}$$

The equalities of vectors $\vec{v}_{a'b'}$ and $\vec{v}_{ab}$, $\vec{v}_{b'c'}$ and $\vec{v}_{bc}$ can be derived as the following:

$$\vec{v}_{a'b'} = ((x_b - x_o) - (x_a - x_o), (y_b - y_o) - (y_a - y_o))$$

$$= (x_b - x_a, y_b - y_a)$$

$$= \vec{v}_{ab} \tag{3.5}$$

$$\vec{v}_{b'c'} = ((x_c - x_o) - (x_b - x_o), (y_c - y_o) - (y_b - y_o))$$

$$= (x_c - x_b, y_c - y_b)$$

$$= \vec{v}_{bc} \tag{3.6}$$

Let $\mathcal{A}$ be the deviating angle from $\vec{v}_{ab}$ to $\vec{v}_{bc}$, and let $\mathcal{A}'$ be the deviating angle from $\vec{v}_{a'b'}$ to $\vec{v}_{b'c'}$. According to definition3.1, we have

$$\mathcal{A} = \left\{ \frac{(\vec{v}_{ab} \times \vec{v}_{bc}) \bullet \vec{k}}{|\vec{v}_{ab} \times \vec{v}_{bc}|} \right\} \cos^{-1} \left\{ \frac{\vec{v}_{ab} \bullet \vec{v}_{bc}}{|\vec{v}_{ab}|\,|\vec{v}_{bc}|} \right\}$$

$$\mathcal{A}' = \left\{ \frac{(\vec{v}_{a'b'} \times \vec{v}_{b'c'}) \bullet \vec{k}}{|\vec{v}_{a'b'} \times \vec{v}_{b'c'}|} \right\} \cos^{-1} \left\{ \frac{\vec{v}_{a'b'} \bullet \vec{v}_{b'c'}}{|\vec{v}_{a'b'}|\,|\vec{v}_{b'c'}|} \right\}$$

From equations 3.5 and 3.6, we prove that

$$\vec{v}_{ab} = \vec{v}_{a'b'}, \qquad \vec{v}_{bc} = \vec{v}_{b'c'}$$

Therefore we conclude that

$$\mathcal{A} = \left\{ \frac{(\vec{v}_{ab} \times \vec{v}_{bc}) \bullet \vec{k}}{|\vec{v}_{ab} \times \vec{v}_{bc}|} \right\} \cos^{-1} \left\{ \frac{\vec{v}_{ab} \bullet \vec{v}_{bc}}{|\vec{v}_{ab}|\,|\vec{v}_{bc}|} \right\}$$

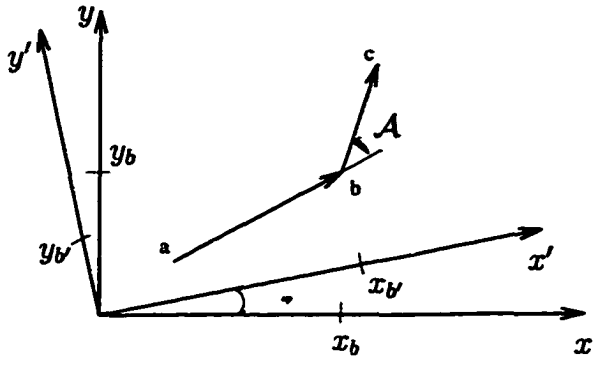
$$= \mathcal{A}'$$

and the theorem is proved.

Theorem 3.3 If a, b, and c are any points on a plane in rectangular coordinate system, the deviating angle from $\vec{v}_{ab}$ to $\vec{v}_{bc}$ is rotation invariance.

Proof: Let the coordinates of points a, b, and c on a plane in a rectangular coordinate system be (x_a, y_a), (x_b, y_b), and (x_c, y_c). And let the coordinates of the same a, b,

and c on a plane in a new rectangular coordinate system, which has its x-axis make an angle α with the positive x-axis of the old coordinate system while the origins of both rectangular system are the same, be $(x_{a'}, y_{a'})$, $(x_{b'}, y_{b'})$, and $(x_{c'}, y_{c'})$ (see figure 3.3). We are proving that the deviating angles from $\vec{v}_{ab}$ to $\vec{v}_{bc}$ are the same on both rectangular coordinate systems.



x

Figure 3.3: The coordinate rotation of point a, b, and c.

Suppose $\vec{v}_{ab} = (x_b - x_a, y_b - y_a)$, $\vec{v}_{bc} = (x_c - x_b, y_c - y_b)$ and $\vec{v}_{a'b'} = (x_{b'} - x_{a'}, y_{b'} - y_{a'})$, $\vec{v}_{b'c'} = (x_{c'} - x_{b'}, y_{c'} - y_{b'})$.

From the transformation of coordinates involving pure rotation[30],

$$x_{a'} = x_a \cos(\alpha) + y_a \sin(\alpha), \qquad y_{a'} = y_a \cos(\alpha) - x_a \sin(\alpha) \tag{3.7}$$

$$x_{b'} = x_b \cos(\alpha) + y_b \sin(\alpha), \qquad y_{b'} = y_b \cos(\alpha) - x_b \sin(\alpha) \tag{3.8}$$

$$x_{c'} = x_c \cos(\alpha) + y_c \sin(\alpha), \qquad y_{c'} = y_c \cos(\alpha) - x_c \sin(\alpha) \tag{3.9}$$

First we derive $\vec{v}_{a'b'}$ and $\vec{v}_{b'c'}$ in terms of x_a, x_b, y_a, and y_b only. From equations 3.7, 3.8, 3.9

$$\vec{v}_{a'b'} = \{(x_b \cos(\alpha) + y_b \sin(\alpha)) - (x_a \cos(\alpha) + y_a \sin(\alpha)),$$

$$(y_b\cos(\alpha) - x_b\sin(\alpha)) - (y_a\cos(\alpha) - x_a\sin(\alpha))\}$$

$$= \{(x_b - x_a)\cos(\alpha) + (y_b - y_a)\sin(\alpha),$$

$$(y_b - y_a)\cos(\alpha) - (x_b - x_a)\sin(\alpha)\} \tag{3.10}$$

$$\vec{v}_{b'c'} = \{(x_c\cos(\alpha) + y_c\sin(\alpha)) - (x_b\cos(\alpha) + y_b\sin(\alpha)),$$

$$(y_c\cos(\alpha) - x_c\sin(\alpha)) - (y_b\cos(\alpha) - x_b\sin(\alpha))\}$$

$$= \{(x_c - x_b)\cos(\alpha) + (y_c - y_b)\sin(\alpha),$$

$$(y_c - y_b)\cos(\alpha) - (x_c - x_b)\sin(\alpha)\} \tag{3.11}$$

We then prove that $\vec{v}_{ab} \times \vec{v}_{bc}$ is equal to $\vec{v}_{a'b'} \times \vec{v}_{b'c'}$.

$$\vec{v}_{ab} \times \vec{v}_{bc} = \{(x_b - x_a)(y_c - y_b) - (x_c - x_b)(y_b - y_a)\}\vec{k}$$

From equations 3.10 and 3.11, we have

$$\vec{v}_{a'b'} \times \vec{v}_{b'c'} = \{((x_b - x_a)\cos(\alpha) - (y_b - y_a)\sin(\alpha))((y_c - y_b)\cos(\alpha) -$$

$$(x_c - x_b)\sin(\alpha)) - ((y_b - y_a)\cos(\alpha) - (x_b - x_a)\sin(\alpha))$$

$$((x_c - x_b)\cos(\alpha) + (y_c - y_b)\sin(\alpha))\}\vec{k}$$

$$= \{(x_b - x_a)(y_c - y_b)\cos^2(\alpha) +$$

$$(y_b - y_a)(y_c - y_b)\sin(\alpha)\cos(\alpha) -$$

$$(x_b - x_a)(x_c - x_b)\sin(\alpha)\cos(\alpha) -$$

$$(y_b - y_a)(x_c - x_b)\sin^2(\alpha) -$$

$$(x_c - x_b)(y_b - y_a)\cos^2(\alpha) +$$

$$(x_b - x_a)(x_c - x_b)\sin(\alpha)\cos(\alpha) -$$

$$(y_b - y_a)(y_c - y_b)\sin(\alpha)\cos(\alpha) +$$

$$
\begin{aligned}
& (x_b - x_a)(y_c - y_b)\sin^2(\alpha)\vec{k} \\
= \; & \{((x_b - x_a)(y_c - y_b) - (x_c - x_b)(y_b - y_a))\cos^2(\alpha) + \\
& ((x_b - x_a)(y_c - y_b) - (x_c - x_b)(y_b - y_a))\sin^2(\alpha)\}\vec{k} \\
= \; & \{((x_b - x_a)(y_c - y_b) - (x_c - x_b)(y_b - y_a))(\sin^2(\alpha) + \\
& \cos^2(\alpha))\}\vec{k} \\
= \; & \{(x_b - x_a)(y_c - y_b) - (x_c - x_b)(y_b - y_a)\}\vec{k} \\
= \; & \vec{v}_{ab} \times \vec{v}_{bc} \qquad (3.12)
\end{aligned}
$$

We further derive the equality of $\vec{v}_{ab} \bullet \vec{v}_{bc}$ and $\vec{v}_{a'b'} \bullet \vec{v}_{b'c'}$.

$$
\vec{v}_{ab} \bullet \vec{v}_{bc} = (x_b - x_a)(x_c - x_b) + (y_b - y_a)(y_c - y_b)
$$

From equations 3.10 and 3.11, we have

$$
\begin{aligned}
\vec{v}_{a'b'} \bullet \vec{v}_{b'c'} = \; & ((x_b - x_a)\cos(\alpha) + (y_b - y_a)\sin(\alpha)) \\
& ((x_c - x_b)\cos(\alpha) + (y_c - y_b)\sin(\alpha)) + \\
& ((y_b - y_a)\cos(\alpha) - (x_b - x_a)\sin(\alpha)) + \\
& ((y_c - y_b)\cos(\alpha) - (x_c - x_b)\sin(\alpha)) \\
= \; & (x_b - x_a)(x_c - x_b)\cos^2(\alpha) + \\
& (x_c - x_b)(y_b - y_a)\sin(\alpha)\cos(\alpha) + \\
& (x_b - x_a)(y_c - y_b)\sin(\alpha)\cos(\alpha) + \\
& (y_b - y_a)(y_c - y_b)\sin^2(\alpha) + \\
& (y_b - y_a)(y_c - y_b)\cos^2(\alpha) - \\
& (x_b - x_a)(y_c - y_b)\sin(\alpha)\cos(\alpha) -
\end{aligned}
$$

$$
\begin{aligned}
& (x_c - x_b)(y_b - y_a)\sin(\alpha)\cos(\alpha) + \\
& (x_b - x_a)(x_c - x_b)\sin^2(\alpha) \\
= \; & (x_b - x_a)(x_c - x_b)(\sin^2(\alpha) + \cos^2(\alpha)) + \\
& (y_b - y_a)(y_c - y_b)(\sin^2(\alpha) + \cos^2(\alpha)) \\
= \; & (x_b - x_a)(x_c - x_b) + (y_b - y_a)(y_c - y_b) \\
= \; & \vec{v}_{ab} \bullet \vec{v}_{bc}
\end{aligned} \tag{3.13}
$$

The equality of $|\vec{v}_{ab}|$ and $|\vec{v}_{a'b'}|$ is being derived as the following.

$$
|\vec{v}_{ab}| = \sqrt{(x_b - x_a)^2 + (y_b - y_c)^2}
$$

From equation 3.10, we have

$$
\begin{aligned}
|\vec{v}_{a'b'}| = \; & \{((x_b - x_a)\cos(\alpha) + (y_b - y_a)\sin(\alpha))^2 + \\
& ((y_b - y_a)\sin(\alpha) - (x_b - x_a)\cos(\alpha))^2\}^{\frac{1}{2}} \\
= \; & \{(x_b - x_a)^2\cos^2(\alpha) + (y_b - y_a)^2\sin^2(\alpha) + \\
& (x_b - x_a)(y_b - y_a)\sin(\alpha)\cos(\alpha) + (y_b - y_a)^2\cos^2(\alpha) - \\
& (x_b - x_a)(y_b - y_a)\sin(\alpha)\cos(\alpha) + (x_b - x_a)\sin^2(\alpha)\}^{\frac{1}{2}} \\
= \; & \{(x_b - x_a)^2(\sin^2(\alpha) + \cos^2(\alpha)) + (y_b - y_a)^2(\sin^2(\alpha) + \\
& \cos^2(\alpha))\}^{\frac{1}{2}} \\
= \; & \sqrt{(x_b - x_a)^2 + (y_b - y_c)^2} \\
= \; & |\vec{v}_{ab}|
\end{aligned} \tag{3.14}
$$

Finally the equality of $|\vec{v}_{bc}|$ and $|\vec{v}_{b'c'}|$ is derived.

$$
|\vec{v}_{bc}| = \sqrt{(x_c - x_b)^2 + (y_c - y_b)^2}
$$

From equation 3.11, we have

$$\begin{aligned}
|\vec{v}_{b'c'}| &= \{((x_c - x_b)\cos(\alpha) + (y_c - y_b)\sin(\alpha))^2 + \\
&\quad ((y_c - y_b)\sin(\alpha) - (x_c - x_b)\cos(\alpha))^2\}^{\frac{1}{2}} \\
&= \{(x_c - x_b)^2\cos^2(\alpha) + (y_c - y_b)^2\sin^2(\alpha) + \\
&\quad (x_c - x_b)(y_c - y_b)\sin(\alpha)\cos(\alpha) + \\
&\quad (y_c - y_b)^2\cos^2(\alpha) - (x_c - x_b)(y_c - y_b)\sin(\alpha)\cos(\alpha) + \\
&\quad (x_c - x_b)\sin^2(\alpha)\}^{\frac{1}{2}} \\
&= \{(x_c - x_b)^2(\sin^2(\alpha) + \cos^2(\alpha)) + (y_c - y_b)^2(\sin^2(\alpha) + \\
&\quad \cos^2(\alpha))\}^{\frac{1}{2}} \\
&= \sqrt{(x_c - x_b)^2 + (y_c - y_b)^2} \\
&= |\vec{v}_{bc}| \qquad (3.15)
\end{aligned}$$

Let $\mathcal{A}$ be the deviating angle from $\vec{v}_{ab}$ to $\vec{v}_{bc}$, and let $\mathcal{A}'$ be the deviating angle from $\vec{v}_{a'b'}$ to $\vec{v}_{b'c'}$. According to definition3.1, we have

$$\begin{aligned}
\mathcal{A} &= \left\{\frac{(\vec{v}_{ab} \times \vec{v}_{bc}) \bullet \vec{k}}{|\vec{v}_{ab} \times \vec{v}_{bc}|}\right\} \cos^{-1}\left\{\frac{\vec{v}_{ab} \bullet \vec{v}_{bc}}{|\vec{v}_{ab}|\,|\vec{v}_{bc}|}\right\} \\
\mathcal{A}' &= \left\{\frac{(\vec{v}_{a'b'} \times \vec{v}_{b'c'}) \bullet \vec{k}}{|\vec{v}_{a'b'} \times \vec{v}_{b'c'}|}\right\} \cos^{-1}\left\{\frac{\vec{v}_{a'b'} \bullet \vec{v}_{b'c'}}{|\vec{v}_{a'b'}|\,|\vec{v}_{b'c'}|}\right\}
\end{aligned}$$

From equations 3.12, 3.13, 3.14, and 3.15, we prove that

$$\begin{aligned}
\vec{v}_{ab} \times \vec{v}_{bc} &= \vec{v}_{a'b'} \times \vec{v}_{b'c'} \\
\vec{v}_{ab} \bullet \vec{v}_{bc} &= \vec{v}_{a'b'} \bullet \vec{v}_{b'c'} \\
|\vec{v}_{ab}| &= |\vec{v}_{a'b'}| \\
|\vec{v}_{bc}| &= |\vec{v}_{b'c'}|
\end{aligned}$$

Therefore we conclude that

$$\begin{aligned}\mathcal{A} &= \left\{\frac{(\vec{v}_{ab}\times\vec{v}_{bc})\bullet\vec{k}}{|\vec{v}_{ab}\times\vec{v}_{bc}|}\right\}\cos^{-1}\left\{\frac{\vec{v}_{ab}\bullet\vec{v}_{bc}}{|\vec{v}_{ab}|\,|\vec{v}_{bc}|}\right\}\\ &= \left\{\frac{(\vec{v}_{a'b'}\times\vec{v}_{b'c'})\bullet\vec{k}}{|\vec{v}_{a'b'}\times\vec{v}_{b'c'}|}\right\}\cos^{-1}\left\{\frac{\vec{v}_{a'b'}\bullet\vec{v}_{b'c'}}{|\vec{v}_{a'b'}|\,|\vec{v}_{b'c'}|}\right\}\\ &= \mathcal{A}'\end{aligned}$$

and the theorem is proved.

Theorem 3.4 If a, b, and c are any points on a plane in rectangular coordinate system, then the deviating angle from $\vec{v}_{ab}$ to $\vec{v}_{bc}$ is scaling invariance.

Proof: Let the coordinates of points a, b, and c on a plane in rectangular coordinate system be (x_a, y_a), (x_b, y_b), and (x_c, y_c). And let the coordinates of points a', b', and c' in the same coordinate system be $(x_{a'}, y_{a'})$, $(x_{b'}, y_{b'})$, and $(x_{c'}, y_{c'})$ where $x_a = sx'_a$, $y_a = sy'_a$, $x_b = sx'_b$, $y_b = sy'_b$, $x_c = sx'_c$, $y_c = sy'_c$ and s is any positive real constant(see figure 3.4). We are proving that the deviating angles from $\vec{v}_{ab}$ to $\vec{v}_{bc}$ is equal to the deviating angle from $\vec{v}_{a'b'}$ to $\vec{v}_{b'c'}$.

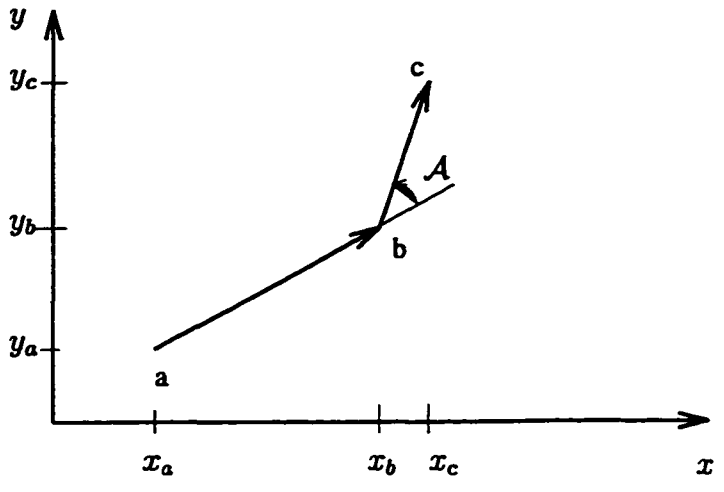
Suppose $\vec{v}_{ab} = (x_b - x_a, y_b - y_a)$, $\vec{v}_{bc} = (x_c - x_b, y_c - y_b)$ and $\vec{v}_{a'b'} = (x_{b'} - x_{a'}, y_{b'} - y_{a'})$, $\vec{v}_{b'c'} = (x_{c'} - x_{b'}, y_{c'} - y_{b'})$.

From the vector representation[30], we derive the equality of $\frac{\vec{v}_{ab}\times\vec{v}_{bc}}{|\vec{v}_{ab}\times\vec{v}_{bc}|}$ and $\frac{\vec{v}_{a'b'}\times\vec{v}_{b'c'}}{|\vec{v}_{a'b'}\times\vec{v}_{b'c'}|}$.

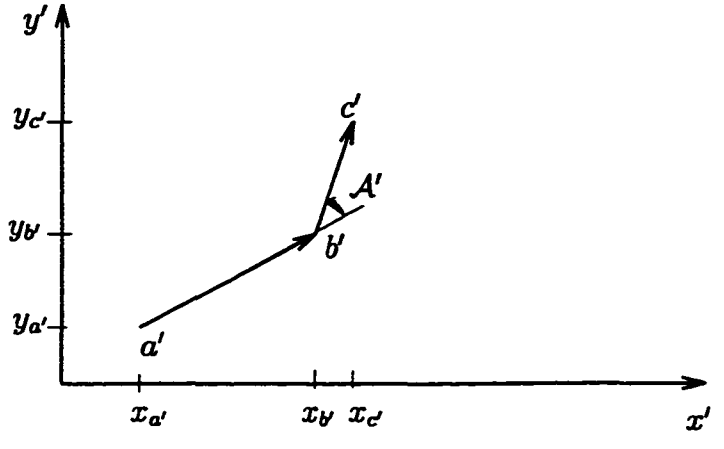
$$\begin{aligned}\vec{v}_{ab}\times\vec{v}_{bc} &= \{(x_b - x_a)(y_c - y_b) - (x_c - x_b)(y_b - y_a)\}\vec{k}\\ |\vec{v}_{ab}\times\vec{v}_{bc}| &= |(x_b - x_a)(y_c - y_b) - (x_c - x_b)(y_b - y_a)|\\ \frac{\vec{v}_{ab}\times\vec{v}_{bc}}{|\vec{v}_{ab}\times\vec{v}_{bc}|} &= \frac{\{(x_b - x_a)(y_c - y_b) - (x_c - x_b)(y_b - y_a)\}\vec{k}}{|(x_b - x_a)(y_c - y_b) - (x_c - x_b)(y_b - y_a)|}\end{aligned} \tag{3.16}$$

From the scaling transformation, we have

$$\vec{v}_{a'b'}\times\vec{v}_{b'c'} = \{(x_{b'} - x_{a'})(y_{c'} - y_{b'}) - (x_{c'} - x_{b'})(y_{b'} - y_{a'})\}\vec{k}$$



(a)



(b)

Figure 3.4: The coordinate scaling of point a, b, and c.

$$= \{(sx_b - sx_a)(sy_c - sy_b) - (sx_c - sx_b)(sy_b - sy_a)\}\vec{k}$$

$$= s^2\{(x_b - x_a)(y_c - y_b) - (x_c - x_b)(y_b - y_a)\}\vec{k} \tag{3.17}$$

$$|\vec{v}_{a'b'} \times \vec{v}_{b'c'}| = |(x_{b'} - x_{a'})(y_{c'} - y_{b'}) - (x_{c'} - x_{b'})(y_{b'} - y'_a)|$$

$$= |(sx_b - sx_a)(sy_c - sy_b) - (sx_c - sx_b)(sy_b - sy_a)|$$

$$= \left|s^2\{(x_b - x_a)(y_c - y_b) - (x_c - x_b)(y_b - y_a)\}\right| \tag{3.18}$$

From equations 3.16, 3.17, and 3.18, we prove the following equalities.

$$\frac{\vec{v}_{a'b'} \times \vec{v}_{b'c'}}{|\vec{v}_{a'b'} \times \vec{v}_{b'c'}|} = \frac{s^2\{(x_b - x_a)(y_c - y_b) - (x_c - x_b)(y_b - y_a)\}\vec{k}}{|s^2\{(x_b - x_a)(y_c - y_b) - (x_c - x_b)(y_b - y_a)\}|}$$

$$= \frac{\{(x_b - x_a)(y_c - y_b) - (x_c - x_b)(y_b - y_a)\}\vec{k}}{|(x_b - x_a)(y_c - y_b) - (x_c - x_b)(y_b - y_a)|}$$

$$= \frac{\vec{v}_{ab} \times \vec{v}_{bc}}{|\vec{v}_{ab} \times \vec{v}_{bc}|} \tag{3.19}$$

Here we derive $\vec{v}_{a'b'} \bullet \vec{v}_{b'c'}$ in term of $\vec{v}_{ab} \bullet \vec{v}_{bc}$.

$$\vec{v}_{ab} \bullet \vec{v}_{bc} = (x_b - x_a)(x_c - x_b) + (y_b - y_a)(y_c - y_b)$$

$$\vec{v}_{a'b'} \bullet \vec{v}_{b'c'} = (x_{b'} - x_{a'})(x_{c'} - x_{b'}) + (y_{b'} - y_{a'})(y_{c'} - y_{b'})$$

$$= (sx_b - sx_a)(sx_c - sx_b) + (sy_b - sy_a)(sy_c - sy_b)$$

$$= s^2\{(x_b - x_a)(x_c - x_b) + (y_b - y_a)(y_c - y_b)\}$$

$$= s^2(\vec{v}_{ab} \bullet \vec{v}_{bc}) \tag{3.20}$$

Finally we derive $|\vec{v}_{a'b'}|$ in term of $|\vec{v}_{ab}|$ and $|\vec{v}_{b'c'}|$ in term of $|\vec{v}_{bc}|$.

$$|\vec{v}_{ab}| = \sqrt{(x_b - x_a)^2 + (y_b - y_c)^2}$$

$$|\vec{v}_{a'b'}| = \sqrt{(x_{b'} - x_{a'})^2 + (y_{b'} - y_{c'})^2}$$

$$= \sqrt{(sx_b - sx_a)^2 + (sy_b - sy_c)^2}$$

$$
\begin{aligned}
&= s\sqrt{(x_b - x_a)^2 + (y_b - y_c)^2} \\
&= s\,|\vec{v}_{ab}| \qquad (3.21)
\end{aligned}
$$

$$
\begin{aligned}
|\vec{v}_{bc}| &= \sqrt{(x_c - x_b)^2 + (y_c - y_b)^2} \\
|\vec{v}_{b'c'}| &= \sqrt{(x_{c'} - x_{b'})^2 + (y_{c'} - y_{b'})^2} \\
&= \sqrt{(sx_c - sx_b)^2 + (sy_c - sy_b)^2} \\
&= s\sqrt{(x_c - x_b)^2 + (y_c - y_b)^2} \\
&= s\,|\vec{v}_{bc}| \qquad (3.22)
\end{aligned}
$$

From equations 3.20, 3.21, and 3.22, we derive the equality of $\frac{\vec{v}_{ab}\bullet\vec{v}_{bc}}{|\vec{v}_{ab}||\vec{v}_{bc}|}$ and $\frac{\vec{v}_{a'b'}\bullet\vec{v}_{b'c'}}{|\vec{v}_{a'b'}||\vec{v}_{b'c'}|}$.

$$
\begin{aligned}
\frac{\vec{v}_{ab}\bullet\vec{v}_{bc}}{|\vec{v}_{ab}|\,|\vec{v}_{bc}|} &= \frac{(x_b - x_a)(x_c - x_b) + (y_b - y_a)(y_c - y_b)}{\sqrt{(x_b - x_a)^2 + (y_b - y_c)^2}\sqrt{(x_c - x_b)^2 + (y_c - y_b)^2}} \\
\frac{\vec{v}_{a'b'}\bullet\vec{v}_{b'c'}}{|\vec{v}_{a'b'}|\,|\vec{v}_{b'c'}|} &= \frac{s^2\{(x_b - x_a)(x_c - x_b) + (y_b - y_a)(y_c - y_b)\}}{\left(s\sqrt{(x_b - x_a)^2 + (y_b - y_c)^2}\right)\left(s\sqrt{(x_c - x_b)^2 + (y_c - y_b)^2}\right)} \\
&= \frac{(x_b - x_a)(x_c - x_b) + (y_b - y_a)(y_c - y_b)}{\sqrt{(x_b - x_a)^2 + (y_b - y_c)^2}\sqrt{(x_c - x_b)^2 + (y_c - y_b)^2}} \\
&= \frac{\vec{v}_{ab}\bullet\vec{v}_{bc}}{|\vec{v}_{ab}|\,|\vec{v}_{bc}|} \qquad (3.23)
\end{aligned}
$$

From equations 3.19 and 3.23, we prove that

$$
\begin{aligned}
\frac{\vec{v}_{ab}\times\vec{v}_{bc}}{|\vec{v}_{ab}\times\vec{v}_{bc}|} &= \frac{\vec{v}_{a'b'}\times\vec{v}_{b'c'}}{|\vec{v}_{a'b'}\times\vec{v}_{b'c'}|} \\
\frac{\vec{v}_{ab}\bullet\vec{v}_{bc}}{|\vec{v}_{ab}|\,|\vec{v}_{bc}|} &= \frac{\vec{v}_{a'b'}\bullet\vec{v}_{b'c'}}{|\vec{v}_{a'b'}|\,|\vec{v}_{b'c'}|}
\end{aligned}
$$

Therefore we conclude that

$$
\begin{aligned}
\mathcal{A} &= \left\{\frac{(\vec{v}_{ab}\times\vec{v}_{bc})\bullet\vec{k}}{|\vec{v}_{ab}\times\vec{v}_{bc}|}\right\}\cos^{-1}\left\{\frac{\vec{v}_{ab}\bullet\vec{v}_{bc}}{|\vec{v}_{ab}|\,|\vec{v}_{bc}|}\right\} \\
&= \left\{\frac{(\vec{v}_{a'b'}\times\vec{v}_{b'c'})\bullet\vec{k}}{|\vec{v}_{a'b'}\times\vec{v}_{b'c'}|}\right\}\cos^{-1}\left\{\frac{\vec{v}_{a'b'}\bullet\vec{v}_{b'c'}}{|\vec{v}_{a'b'}|\,|\vec{v}_{b'c'}|}\right\} \\
&= \mathcal{A}'
\end{aligned}
$$

where $\mathcal{A}$ is the deviating angle from $\vec{v}_{ab}$ to $\vec{v}_{bc}$, $\mathcal{A}'$ is the deviating angle from $\vec{v}_{a'b'}$ to $\vec{v}_{b'c'}$, and the theorem is proved.

3.2 Description of Deviating Angular Feature

Previously the deviating angle from any three points on a plane has been defined, we will now describe the deviating angular feature as a finite sequence of the deviating angles calculated from a finite sequence of points on a plane in rectangular coordinate system. If the finite sequence of points is an ordered set of pixel points on an image curvature, the deviating angular feature can be the excellent description of changes in directions of image curvature, and it is completely invariant in translation, rotation, and scaling. In figure 3.5, suppose that point a, b, c, d, and e be a sequence of points on the curvature line. Consequently, we can generate a deviating angular feature consisting of three deviating angles $\mathcal{A}_1$, $\mathcal{A}_2$, and $\mathcal{A}_3$ from these five points.

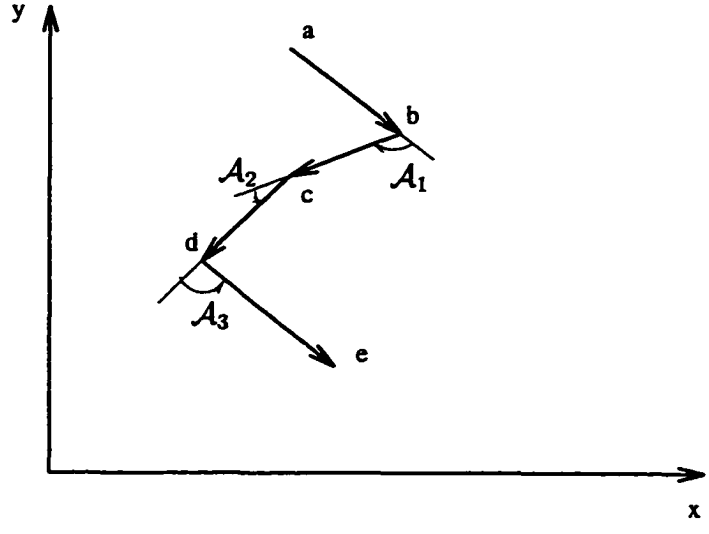


Figure 3.5: The deviating angular feature of points a,b, c, d, e.

Definition 3.5 Let $\{p_0, p_1, p_2, \ldots, p_n\}$ be a sequence of points on a plane in rectangular coordinate system. The deviating angular feature of these sequential points is

defined to be

$$\mathcal{DF} = \{\mathcal{A}_k\}$$

where $\mathcal{A}_k$ is the deviating angle from $\vec{v}_{p_{k-1}p_k}$ to $\vec{v}_{p_kp_{k+1}}$ for which $1 \leq k \leq n-1$.

Theorem 3.6 If $\{p_0, p_1, p_2, \ldots, p_n\}$ is a sequence of points on a plane in rectangular coordinate system, then $\mathcal{DF}$ is the deviating angular feature of these sequential points. The deviating angular feature, $\mathcal{DF}$, is translation invariance, rotation invariance and scaling invariance.

Proof: From definition 3.5, the deviating angular feature of a sequence of $n+1$ points on a plane in rectangular coordinate system is defined to be

$$\mathcal{DF} = \{\mathcal{A}_k\} \tag{3.24}$$

where

$$\mathcal{A}_k = \left\{ \frac{(\vec{v}_{p_{k-1}p_k} \times \vec{v}_{p_kp_{k+1}}) \bullet \vec{k}}{\left|\vec{v}_{p_{k-1}p_k} \times \vec{v}_{p_kp_{k+1}}\right|} \right\} \cos^{-1} \left\{ \frac{\vec{v}_{p_{k-1}p_k} \bullet \vec{v}_{p_kp_{k+1}}}{\left|\vec{v}_{p_{k-1}pk}\right| \left|\vec{v}_{pkpk+1}\right|} \right\} \tag{3.25}$$

for which $1 \leq k \leq n-1$ (see definition3.1).

From theorem 1, theorem 2 and theorem 3, we proved that $\mathcal{A}_k$ is translation invariance, rotation invariance, and scaling invariance. Since $\mathcal{A}_k$ represents all elements of $\mathcal{DF}$, we can now prove that the deviating angular feature, $\mathcal{DF}$, is also translation invariance, rotation invariance, and scaling invariance.

3.3 Summary

In this chapter we have established the formal definition of the deviating angular feature to be a finite sequence of the deviating angles derived from a fixed ordered set of points on a plane in rectangular coordinate system. Further we provide the

proved theories explaining that deviating angles and deviating angular feature are translation, scaling, and rotation invariance.

Chapter 4
Feature Extractions

Perhaps the most influential process of image recognition system is the feature extraction, because this is the process that translates images to the encoded information which is sent to the classification process where the recognition takes place. Obviously, the qualities of the output from feature extraction process have an immense contribution to the success of the system accuracy. Generally, the powerful feature extraction techniques for image recognition system must generate the features, which typically contain the common characteristics of images, while suppress any meaningless interferences. These techniques should also be reliable enough to withstand any nonuniform distortions and vagueness which images may have.

In the current chapter we will discuss two feature extraction processes which we have specially customized for a handwritten numeral recognition system and an automatic aircraft identification system. We will then elaborate on the uses of the deviating angular feature which is used in both customized processes.

4.1 The Feature Extraction for Handwritten Numerals

In order to achieve a highly valuable features from handwritten numeral images, we have to customize a few techniques for being used in specific purposes. As a results of our preliminary experiments, we found that a combination of deviating angular feature and Kirsch feature can improve the accuracy of handwritten numeral recognition system. Thus our feature extraction process is designed to generate two

types of features: the deviating angular feature of image profiles, and the Kirsch feature in horizontal and vertical directions.

4.1.1 Deviating Angular Feature of Handwritten Numerals

The definition of the deviating angular feature has been presented, and now the image profile extracting techniques for handwritten numeral will be described. If we intuit that the algorithm for extracting the profile of binary bitmap images is the process of identifying and collecting the sequence of image pixel points on the profiles, then we may sample the collected sequence of pixel points in order to regulate the size of the deviating angular feature.

Continuing in this direction, our feature extraction process for deviating angular feature of handwritten numeral images is divided into three operational steps. The first step was to develop algorithms to collect pixels representing the left profile and the right profile of images. Second we treated these collected pixels as points in a rectangular coordinate system; the result of conjoining these sequential pixel points with straight lines is a piecewise image curvature; we sampled the piecewise image curvature to get n marking points. Third we calculated the deviating angular feature of size $n - 2$ from the n marking points. The followings are the details of these operations and the analysis of their results.

4.1.1.1 Algorithms of image profile extraction

In this section we will describe two algorithms to extract the left profile and right profile of a handwritten numeral image through a high-level language named *Pidgin ALGOL*[1]. Here we will assume that **PIX**[N, N] is two dimensional array representing the image binary bitmap, while **CURV**[N] is an array of structure which consists

of two variables for containing row index and column index of pixel points on the image profile.

The first algorithm, **RIGHT_CURV**, basically searches for black pixels of a binary bitmap image from the top to bottom rows of the right profile and collects any first black pixels found on each rows in sequential order (see the algorithm in figure 4.1).

The second algorithm, **LEFT_CURV**, works very much the same way as **RIGHT_CURV**, except this algorithm collects the black pixel of a binary bitmap image from top to bottom row of the left profile. The complete details of the algorithm are also described in figure 4.2.

The outputs of these two algorithms can be illustrated in figure 4.3, which illustrates how image of a numeral(number four) is processed and the pixels of left and right profiles are extracted. After these pixel points are connected with straight lines, we get two piecewise profiles which will be used in the next operation.

4.1.1.2 Placing a number of Marking Points

Next we sample the image profile piecewise in such a way that the marking points are spread equally in vertical(see figure 4.4). In practice, the method of placing a number of marking points on the image piecewise profiles can be done by computing the coordinates of n marking points through interpolation. This occurs because we can get the y-coordinates of all marking by dividing the difference between the y-coordinates of the highest and lowest pixel points collected in the previous process into equally $n-1$ intervals(Δy). Thus, y-coordinate of the first sampling point(y_1) will be equal to the y-coordinate of the highest pixel point, and the y-coordinate of n^{th}

```
input
      ROW  : maximum numbers of rows in the bitmap
      COL  : maximum numbers of columns in the bitmap
      PIX  : two dimentional array(binary bitmap)
output
      CURV : two dimentional array collecting pixels
```

```
0.  procedure RIGHT_CURVE
1.  begin
2.        i ← ROW;
3.        j ← COL;
4.        k ← 0;
5.        while i ≥ 0 do
6.         begin
7.              while j ≥ 0 do
8.              begin
9.                    if(PIX[i, j] = BLACK) then
10.                   begin
11.                         CURV[k].Y ← i;
12.                         CURV[k].X ← j;
13.                         k ← k + 1;
14.                         break;
15.                   end
16.                   j ← j − 1;
17.             end
18.             i ← i − 1;
19.       end
20.  end
```

Figure 4.1: Algorithm for Right Curvature extraction

```
input
        ROW  : maximum numbers of rows in the bitmap
        COL  : maximum numbers of columns in the bitmap
        PIX  : two dimentional array(binary bitmap)
output
        CURV : two dimentional array collecting pixels
```

```
0.  procedure LEFT_CURV
1.  begin
2.          i ← ROW;
3.          j ← 0;
4.          k ← 0;
5.          while i ≥ 0 do
6.          begin
7.                  while j ≤ COL do
8.                  begin
9.                          if( PIX[i, j] = BLACK ) then
10.                         begin
11.                                 CURV[k].Y ← i;
12.                                 CURV[k].X ← j;
13.                                 k ← k + 1;
14.                                 break;
15.                         end
16.                         j ← j + 1;
17.                 end
18.                 i ← i + 1;
19.         end
20.  end
```

Figure 4.2: Algorithm for Left Curvature extraction

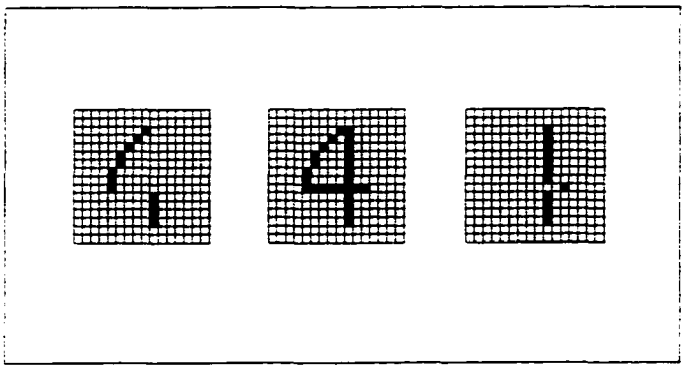


Figure 4.3: The left and right profiles of a numeral image

marking point will be $y_1 - (n-1)\Delta y$. From these y-coordinates of n marking points, we can get their x-coordinates from a linear interpolation. For example, suppose point a, z and b are sequential on a straight line in which the x-coordinates and y-coordinates of point a and b are known. The x-coordinate of point z can be calculated from the formula for linear interpolation:

$$x_z = \frac{y_z - y_a}{y_b - y_z}(x_b - x_z) + x_a \tag{4.1}$$

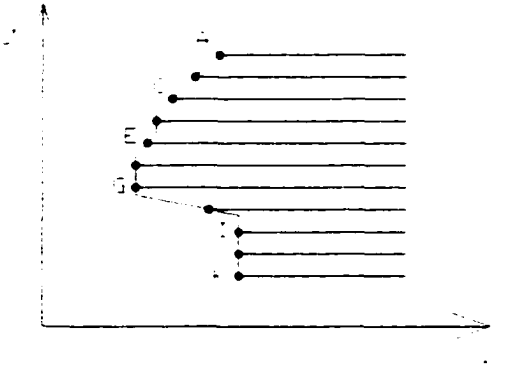


Figure 4.4: Placing the vertical marking points in the left profile

4.1.1.3 Calculating the Deviating Angular Feature

From the sequence of n marking points $\{p_0, p_1, p_2, \ldots, p_n\}$, we calculate a sequence of deviating angles from formula 3.25. The sequence of $n-2$ deviating angles becomes the deviating angular feature of the image curvature which is ready to be used in the classification process.

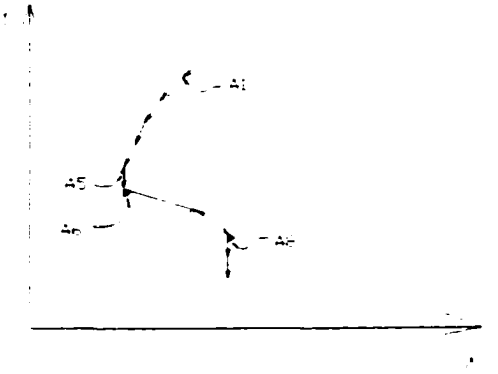


Figure 4.5: The deviating angular feature

From the figure 4.5, suppose $\{a, b, c, \ldots, j, k\}$ are eleven marking points of the left profile piecewise on the rectangular coordinate system and that we have the sequence of vectors $\{\vec{v}_{ab}, \vec{v}_{bc}, \vec{v}_{cd}, \ldots, \vec{v}_{ij}, \vec{v}_{jk}\}$ over the marking points. Obviously we get a deviating angular feature of nine deviating angles $\{\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \ldots, \mathcal{A}_9\}$.

Finally it is important to notice that the number of deviating angles of the feature is always two less than the number of the marking points.

4.1.1.4 The Analysis

The deviating angular features we get from the left profile and the right profile are theoretically translation, rotation, and scaling invariances. However, the approach utilizing image profiles has a limitation due to ambiguities from the information of

image outer profiles. This limitation is definitely a serious problem if images from different categories have no differences on their profiles.

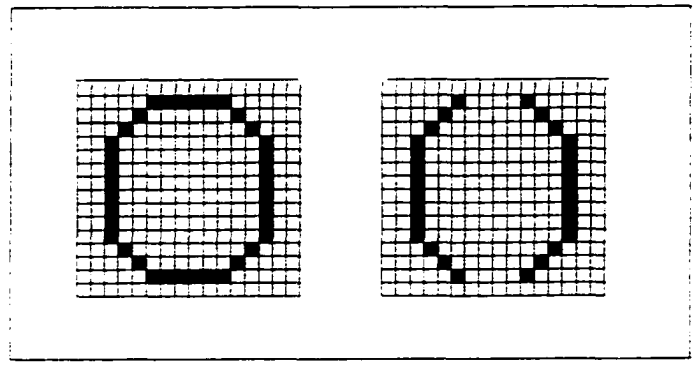


Figure 4.6: Left:The image of number 0, Right:The extracted left and right profiles.

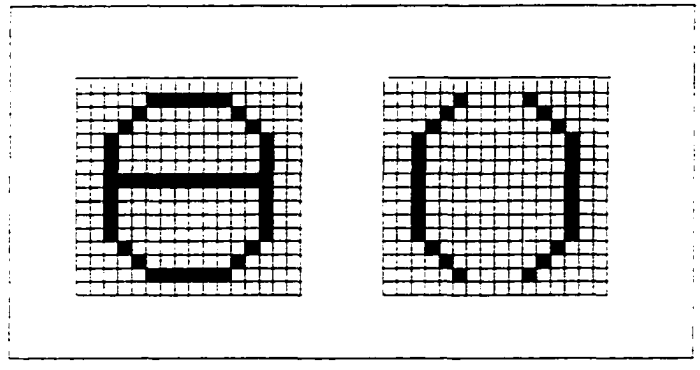


Figure 4.7: Left:The image of number 8, Right:The extracted left and right profiles.

To illustrate the problematic scenario, we suppose that $\mathcal{DF}_0$ and $\mathcal{DF}_8$ be the deviating angular features from the images of numeral 0 and 8 in figure 4.6(L) and 4.7(L).

$$\mathcal{DF}_0 = \{\mathcal{A}_1^0, \mathcal{A}_2^0, \ldots, \mathcal{A}_n^0\}$$

$$\mathcal{DF}_8 = \{\mathcal{A}_1^8, \mathcal{A}_2^8, \ldots, \mathcal{A}_n^8\}$$

From figures 4.6(R) and 4.7(R), we can see that the profiles of two numeral images extracted by the the left and right profile generating algorithms are exactly identical therefore, the deviating angular features of both images are also identical. In other

words

$$\mathcal{DF}_0 = \mathcal{DF}_8 \tag{4.2}$$

In order to get a correct classifications, classifiers(functions) must produce two different outputs (or function values) from the identical inputs to distinguish the image of number 0 from the image of number 8. By definition, a *function* is an association between each element in a set of independent variables(domain) to an *unique* element of a set of dependent variables(range) governed by an unambiguous rule. Obviously, to distinguish the previous two numeral images would require the one-to-many association(mapping), which is mathematically impossible(see figure 4.8). Fortunately the handwritten numeral images similar to this example are very rare, though they cannot be ignored.

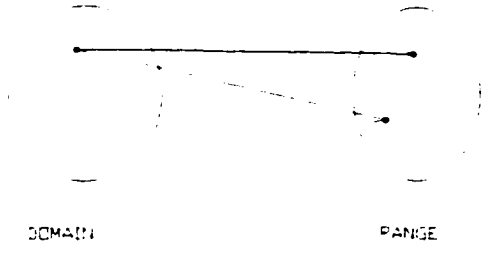


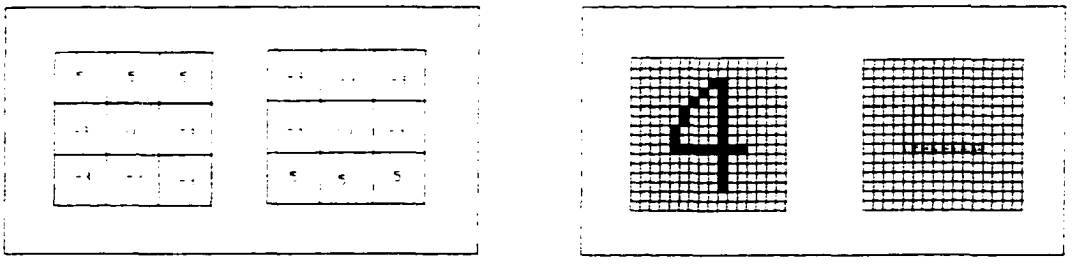
Figure 4.8: One-to-many association(mapping), $f : x \rightarrow y$.

To solve the problem, we have used more features to reveal the internal details of the numerals, which in turn give the combined distinguishable features for the images from different categories.

4.1.2 Kirsch Feature

The Kirsch feature is the feature of local line segment indication. The feature can be generated by applying the directional filter functions to each pixels of an binary bitmap image, and the function values of each pixels becomes a sequence of quantities representing the Kirsch feature in one direction. Following are two types of Kirsch functions used in our handwritten numeral recognition system: the horizontal Kirsch function, and the vertical Kirsch function. The horizontal Kirsch function is defined as

$$k_h(i,j) = \max\left[\left|5u_1(i,j) - 3d_1(i,j)\right|, \left|5u_2(i,j) - 3d_2(i,j)\right|\right]$$



(a) Horizontal Kirsch filter function (b) Horizontal Kirsch feature

Figure 4.9: Horizontal Kirsch feature and its filter function

where i and j are the indexes of the pixel to which the horizontal Kirsch function is applied and

$$u_1 = pix[i+1, j-1] + pix[i+1, j] + pix[i+1, j+1]$$

$$d_1 = pix[i-1, j-1] + pix[i-1, j] + pix[i-1, j+1] + pix[i, j-1] + pix[i, j+1]$$

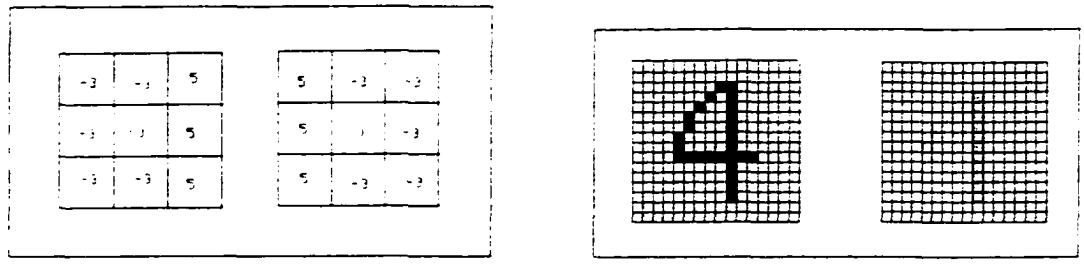
$$u_2 = pix[i+1, j-1] + pix[i+1, j] + pix[i+1, j+1] + pix[i, j-1] + pix[i, j+1]$$

$$d_2 = pix[i-1, j-1] + pix[i-1, j] + pix[i-1, j+1]$$

For a further illustration of Kirsch feature extraction, figure 4.9(a) describes the horizontal Kirsch filter as it is applied to the image pixel at row i, column j, and the horizontal Kirsch feature of a numeral is depicted in figure 4.9(b).

The vertical Kirsch function is similarly defined as

$$k_v(i,j) = \max\left[\left|5l_1(i,j) - 3r_1(i,j)\right|, \left|5l_2(i,j) - 3r_2(i,j)\right|\right]$$



(a) Vertical Kirsch filter function (b) Vertical Kirsch feature

Figure 4.10: Vertical Kirsch feature and its filter function

where i and j are the indexes of the pixel to which the vertical Kirsch function is applied, and

$$l_1 = pix[i+1, j-1] + pix[i, j-1] + pix[i-1, j-1] + pix[i+1, j] + pix[i-1, j]$$

$$r_1 = pix[i+1, j+1] + pix[i, j+1] + pix[i-1, j+1]$$

$$l_2 = pix[i+1, j-1] + pix[i, j-1] + pix[i-1, j-1]$$

$$r_2 = pix[i+1, j+1] + pix[i, j+1] + pix[i-1, j+1] + pix[i+1, j] + pix[i-1, j]$$

The vertical Kirsch function applied to image pixel at row i and column j is described in figure 4.10(a) and the vertical Kirsch feature from a numeral is also depicted in figure 4.10(b).

4.2 The Feature Extraction for Aircraft Silhouette

The feature extraction technique which we introduce specially for the automatic aircraft identification system is mostly suitable to extract the deviating angular feature from a silhouette, since it guarantees to produce a feature with pure qualities of translation, scaling, rotation invariance. Because each particular types of image usually required customizations, the specialized feature extraction technique introduced here can be divided into three unique operational steps. First, a highly efficient algorithm based on the approach of finite state machine was developed to generate a sequence of pixels on the aircraft silhouette curvature. Second, we calculated the deviating angular feature from the sequence of collected pixel points, using the formulas in equation 3.1 and 3.25 with a smoothing method to reduce rippling changes in the feature caused by low bitmap resolution. Third, we treated the smooth deviating angular feature as a periodic function which was then sampled for calculating the complex coefficient modulus in the Fourier series of the function.

4.2.1 Algorithm of Silhouette Curvature Extraction

In order to collect the pixels representing the image represented on the silhouette curvature, we invented an algorithm which begins its search from a single pixel on the silhouette curvature; the algorithm then traversed counterclockwise to the next pixel by inspecting the eight closet neighboring pixels. The algorithm ceases its search when the complete sequence of the pixels on the silhouette curvature has been found(see figure 4.11). This search algorithm can precisely be described in pseudo code in figure 4.12 and figure 4.13.

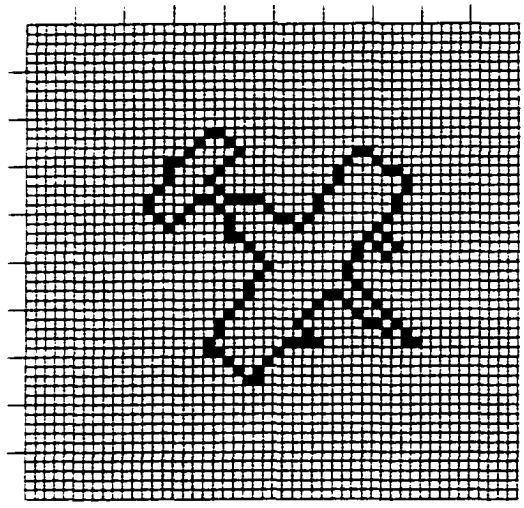


Figure 4.11: The silhouette curvature of an aircraft

input
SX : column index of the first pixel on curvature
SY : row index of the first pixel on curvature
output
CURV : array of structures for containing pixel point coordinates

```
0.  procedure SILHOUETTE_CURV
1.  begin
2.      j ← SX;
3.      i ← SY;
4.      k ← 0;
5.      while (j, i) ≠ (SX, SX) do
6.      begin
7.          CURV[k].X ← j;
8.          CURV[k].Y ← i;
9.          k ← k + 1;
10.         (j, i) ← NEXT_POINT( j, i );
11.     end
12. end
```

Figure 4.12: Algorithm for silhouette curvature extraction

input

X : column index of the pixel on curvature
Y : row index of the pixel on curvature
PIX : two dimentional array(binary bitmap)
N : two dimentional array indicating the offset coordinates of eight neighboring pixels

return

the new coordinate of the next pixel on the image curvature

```
0.  procedure NEXT_POINT
1.  begin
2.      i ← 0;
3.      while i ≤ 8 do
4.      begin
5.          if X+N[i].X or Y+N[i].Y out of bound then
6.              break;
7.          if PIX[X+N[i].X, Y+N[i].Y] ≠ BLACK then
8.              break;
9.          i ← i + 1;
10.     end;
11.     while i ≤ 8 do
12.     begin
13.         if X+N[i].X or Y+N[i].Y out of bound then
14.             continue;
15.         if PIX[X+N[i].X, Y+N[i].Y] ≠ WHITE then
16.             break;
17.         i ← i + 1;
18.     end;
19.     return ( X+N[i].X, Y+N[i].Y )
20. end
```

Figure 4.13: Algorithm for the next point finding

4.2.2 The Smooth Deviating Angular Feature

As previous research established, it is possible for a deviating angle to be computed from a set of three points on a plane and a sequence of deviating angles eventually became the deviating angular feature. From the result of silhouette curvature extracting algorithm, we have a circular sequence of n pixel points on the silhouette curvature. From this circular sequence, we can calculate a deviating angular feature consisting of n deviating angles. However the deviating angular feature of the silhouette curvature may exhibit meaningless rippling changes due to the low resolution of binary bitmap. To solve this problem, we have to calculate a deviating angle from a pixel point and two other points at the average locations of m pixel points(where m is a small integer) before and after the pixel point we select. The formulas for calculating the locations of these two average points at which the pixel point i are located are as follows:

$$\begin{aligned}(x_f, y_f) &= ((x_{i-1} + x_{i-2} + \ldots + x_{i-m})/m, (y_{i-1} + y_{i-2} + \ldots + y_{i-m})/m) \\ (x_l, y_l) &= ((x_{i+1} + x_{i+2} + \ldots + x_{i+m})/m, (y_{i+1} + y_{i+2} + \ldots + y_{i+m})/m)\end{aligned}$$

where (x_f, y_f) and (x_l, y_l) are the coordinates of the average points before and after the pixel i and its coordinate (x_i, y_i). Consequently, $\vec{v}_{fi}$ and $\vec{v}_{il}$ are defined to be

$$\begin{aligned}\vec{v}_{fi} &= (x_i - x_f, y_i - y_f) \\ \vec{v}_{il} &= (x_l - x_i, y_l - y_i)\end{aligned} \tag{4.3}$$

Finally we calculate the deviating angle($\mathcal{A}_i$) from

$$\mathcal{A}_i = \left\{ \frac{(\vec{v}_{fi} \times \vec{v}_{il}) \bullet \vec{k}}{|\vec{v}_{fi} \times \vec{v}_{il}|} \right\} \cos^{-1} \left\{ \frac{\vec{v}_{fi} \bullet \vec{v}_{il}}{|\vec{v}_{fi}|\,|\vec{v}_{il}|} \right\} \tag{4.4}$$

The deviating angular feature derived from this calculation is called the smooth deviating angular feature; it is the feature that better reflects the fidelity of the silhouette curvature for binary bitmap image.

4.2.3 The Modulus of Complex Coefficients in Fourier Series

As soon as we get the deviating angular feature, we get the image feature which is absolutely invariant to translation, scaling, and rotation of a image. However, if we interpret the deviating angular feature as a periodic function of a complete cycle over the distance of an silhouette curvature, we can see that the periodic function of the same image can differ when the feature extraction algorithm selects different points on the curvature silhouette for calculating the smooth deviating angular feature. In other words, we still have one more variance to be solved: the phase shifting variance of periodic function. We solve this problem by calculating the modulus of complex coefficients in the Fourier series expansion of the deviating angular feature. Then we use the finite sequence of the modulus of complex coefficients as the input to the classification process because the modulus of complex coefficients are absolutely invariant to the phase shifting in a periodic function. The justification for the claim is achieved through the following Fourier series analysis [11, 17, 35].

Suppose $f(x)$ is a periodic function,

$$f(x) = \frac{a_0}{2} + \sum_{j=1}^{\infty} [a_j \cos(jx) + b_j \sin(jx)] \tag{4.5}$$

where the period is 2π.

From Euler's identity

$$e^{ijx} = \cos(jx) + i \sin(jx) \tag{4.6}$$

where $i = \sqrt{-1}$, the Fourier series 4.5 can be written in another form,

$$f(x) = \sum_{j=0}^{\infty}(c_j e^{ijx} + c_{-j}e^{-ijx}) \tag{4.7}$$

where $c_j = \frac{a_j - ib_j}{2}$, $c_{-j} = \frac{a_j + ib_j}{2}$.

From equation 4.7, we rearrange $f(x)$ to be

$$\begin{aligned} f(x) &= 2c_0 + \sum_{j=1}^{\infty}[(c_j + c_{-j})\cos(jx) + i\,(c_j - c_{-j})\sin(jx)] \\ &= \sum_{j=-\infty}^{\infty} c_j e^{ijx} \end{aligned} \tag{4.8}$$

From the formula of complex coefficients in Fourier series[30], we have

$$c_j = \frac{1}{2\pi}\int_0^{2\pi} f(x)e^{-ijx}dx \tag{4.9}$$

Please note that if the $f'(x)$ is actually $f(x)$ shifted by τ, the Fourier series of this shifted function would be the following,

$$\begin{aligned} f'(x-\tau) &= f(x) = \sum_{j=-\infty}^{\infty} c'_j e^{ij(x-\tau)} \\ c'_j &= \frac{1}{2\pi}\int_0^{2\pi} f(x)e^{-ij(x-\tau)}dx \end{aligned} \tag{4.10}$$

If we focus on only the modulus of c_j and c'_j in equation 4.8 and equation 4.10, we will see that

$$\begin{aligned} |c_j| &= \left|\frac{1}{2\pi}\int_0^{2\pi} f(x)e^{-ijx}dx\right| \\ &= \left|\frac{1}{2\pi}\right|\left|\int_0^{2\pi} f(x)e^{-ijx}dx\right| \end{aligned} \tag{4.11}$$

and

$$|c'_j| = \left|\frac{1}{2\pi}\int_0^{2\pi} f(x)e^{-ij(x-\tau)}dx\right|$$

$$
\begin{aligned}
&= \left|\frac{1}{2\pi}\right| \left| \int_0^{2\pi} f(x) e^{-ijx} e^{ij\tau} dx \right| \\
&= \left|\frac{1}{2\pi}\right| \left| e^{ij\tau} \right| \left| \int_0^{2\pi} f(x) e^{-ijx} dx \right| \\
&= \left|\frac{1}{2\pi}\right| \left| \int_0^{2\pi} f(x) e^{-ijx} dx \right| \qquad (4.12)
\end{aligned}
$$

where $|e^{ij\tau}| = 1$.

Consequently we have $|c_j| = |c'_j|$ which means that the modulus of complex coefficients in Fourier series of two functions different only in phase shifting are equivalent. (see equation 4.11 and equation 4.12) and the modulus of the complex coefficient can be calculated from equation 4.13,

$$|c_j| = \frac{\sqrt{a_{|j|}^2 + b_{|j|}^2}}{2} \qquad (4.13)$$

4.3 Summary

In this chapter we introduce the customized feature extraction processes originally invented for handwritten numeral images and aircraft silhouettes. The feature extraction process for handwritten numeral images consists of four operational stages: 1) image profile extraction; 2) placing a finite marking points on the image profiles; 3) calculation of deviating angular features; and 4) calculation of the Kirsch features. This feature extraction process can provide the valuable features to the system classifier.

Another feature extraction process for aircraft silhouettes consists of three operational stages: silhouette curvature extraction; calculation of smooth deviating angular feature; and calculation of the modulus of complex coefficients in the Fourier series of the smooth deviating angular feature. This feature extraction process can provide

a valuable feature which is completely invariant in translation, scaling, and rotation to the system classifier.

Chapter 5
The Neural Network Classifiers

The last process of an image recognition system is Classification. This process is very crucial to the entire operations of the recognition system because it is the mechanism that operates on the abstract information of images and produces the outputs that distinguish images of one class from the others. In mathematical sense, this process is basically a particular case of function approximation which can be defined as the classification function $g : R^d \longrightarrow \{1, \ldots, c\}$, where $x \in R^d$, $g(x)$ is an integer output in $1, \ldots, c$ and c is the number of classes[4, 25].

To approximate the classification function, we have many techniques from statistical methods, neural networks, and expert systems that can usually be used as the classifiers of the image recognition systems. In spite of the availabilities of many alternative methods, we choose neural network approach for our system classifiers for many reasons. From our extensive literature researches, we discover that neural network is better than classical statistical regression methods because neural network does not require a specific form of a function to be fitted to the train data, neither it depends on linear superposition and orthogonal functions which linear statistical regression methods are required to use[14]. Further more there are experimental evidents which indicates that the performance of neural network is at least comparable to the best nonlinear regression method [37, 38] and it has significant advantages in solving many problems that have high dimensional input[7]. The advantages of neural networks over expert systems are also significant since neural networks require no

guidances from rules or human experts. Neural networks can easily be trained continuously from the updated train data while expert systems require rule reconstruction in the programs [27].

In this chapter we are describing an particular type of multilayered feedforward neural network called the committee of neural network classifier and the original training algorithm named *Epoch Gradual Increase in Accuracy.* Epoch Gradual Increase in Accuracy training algorithm is invented to improve the performance of Gradual Increase in Accuracy training algorithm for backpropagation by utilizing a set of train patterns that produce errors higher than the intermediate tolerance at every epochs. From this modification, Epoch Gradual Increase in Accuracy can obtain the better estimates of the error gradient at all time which in turn guide the weight updating of the neural network to reduce the overall error faster than Gradual Increase in Accuracy. In this chapter we are first reviewing the backpropagation training algorithm, its mathematical model and problems. Later we describe Gradual Increase in Accuracy training algorithm and its improvements to Backpropagation. Finally the algorithm of *Epoch Gradual Increase in Accuracy* is explained and the mathematical analysis of the improvement of Epoch Gradual in Accuracy training algorithm will be given.

5.1 The Backpropagation Training Algorithm

The basic idea of the backpropagation training algorithm is the search algorithm using an optimizing method named gradient descent as the guiding for minimizing a sum-of-square error function. Assuming $F(w) = F(w_1, w_2, .., w_q)$ be an differentiable error function of a system varies on a number of independent weight variables, w_n [14]. According to gradient descent method, the weight vector should be updated for

a small distance in weight space in the direction of $-\nabla_w F(w)$. Therefore we calculate the updating of w_{lij}(weight between node i and node j in layer l) for input pattern k^{th} by

$$w_{lij}^{new} = w_{lij}^{old} - \eta \frac{\partial F_k(w)}{\partial w_{lij}} \tag{5.1}$$

where η is the learning rate. To avoid the overshooting problem, η would be set to a small positive number.

By iterating the updating process for all q weights and all N train patterns, the updating of all w proceeding in this direction will decrease $F(w)$ at the fastest rate[4].

In order to apply equation 5.1 for training the multilayer feedforward neural network, $\frac{\partial F_k(w)}{\partial w_{lij}}$ for the output layer and hidden layers can be derived as the following.

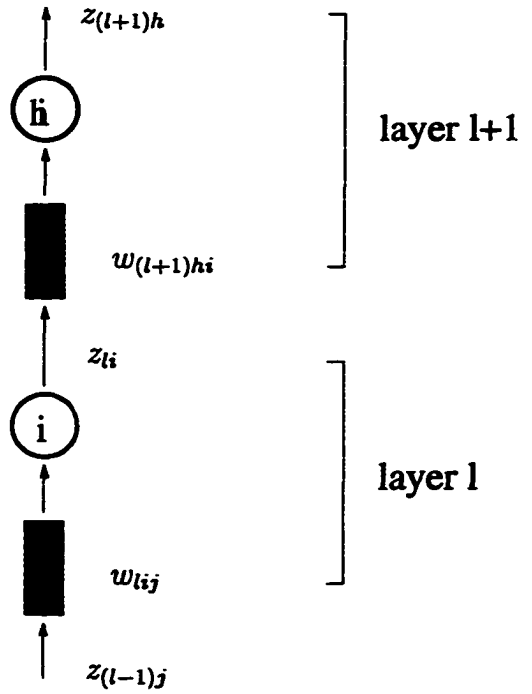


Figure 5.1: The feedforward neural networks

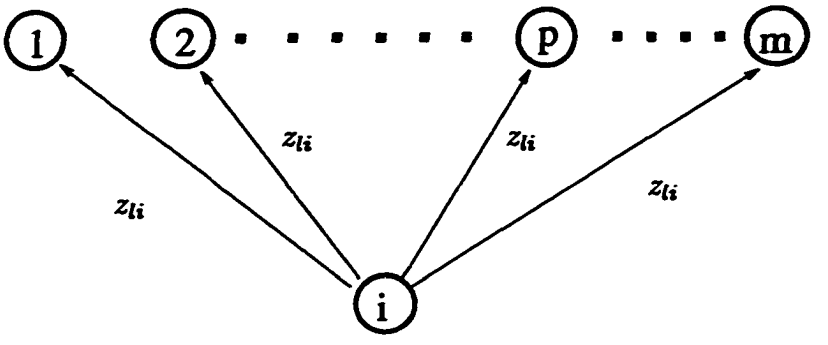


Figure 5.2: The output signal is distributed to every nodes in the next layer.

Using the chain rules(see also figure 5.1),

$$\begin{aligned}\frac{\partial F_k(w)}{\partial w_{lij}} &= \frac{\partial F_k}{\partial I_{li}}\frac{\partial I_{li}}{\partial w_{lij}} \\ &= \delta_{li}^k \frac{\partial(\sum_q w_{liq} Z_{(l-1)q}^k)}{\partial w_{lij}} \\ &= \delta_{li}^k Z_{(l-1)j}^k\end{aligned}$$

for the k^{th} train pattern.

If l^{th} row is **not** the output layer of the neural network, δ_{li}^k can be derived as the following.

$$\begin{aligned}\delta_{li}^k &= \frac{\partial F_k}{\partial I_{li}} \\ &= \frac{\partial F_k}{\partial Z_{li}}\frac{\partial Z_{li}}{\partial I_{li}} \\ &= \frac{\partial F_k}{\partial Z_{li}} s'(I_{li}) \qquad (5.2)\end{aligned}$$

Using the chain rules(see also figure 5.2)

$$\begin{aligned}\frac{\partial F_k}{\partial Z_{li}} &= \sum_{p=1}^{m_{(l+1)}} \left(\frac{\partial F_k}{\partial I_{(l+1)p}}\frac{\partial I_{(l+1)p}}{\partial Z_{li}}\right) \\ &= \sum_{p=1}^{m_{(l+1)}} (\delta_{(l+1)p}^k w_{(l+1)pi}) \qquad (5.3)\end{aligned}$$

From equation 5.2 and 5.3, for the hidden layer,

$$\delta_{li}^{k} = \left\{ \sum_{p=1}^{m_{(l+1)}} (\delta_{(l+1)p}^{k} w_{(l+1)pi}) \right\} s'(I_{li})$$

and the weight in hidden layer are updated by

$$w_{lij}^{new} = w_{lij}^{old} - \eta \left\{ \sum_{p=1}^{m_{(l+1)}} (\delta_{(l+1)p}^{k} w_{(l+1)pi}) \right\} s'(I_{li}) z_{(l-1)j}^{k}$$

If l^{th} is the output layer, then

$$\begin{aligned} \frac{\partial F_k}{\partial Z_{li}} &= \frac{\partial \sum_{p=1}^{m} (y_p^k - z_{lp}^k)^2}{\partial Z_{li}} \\ &= -2(y_i^k - z_{li}^k) \end{aligned}$$

where y_p^k is the p^{th} element of y_k(the k^{th} target).

Thus, for the output layer,

$$\delta_{li}^{k} = -2(y_i^k - z_{li}^k)$$

Therefore $F(w)$ can be reduced by moving w_{lij} in $\frac{\partial F(w)}{\partial w_{lij}}$, from equation 5.1 the weight in output layer are updated by

$$w_{lij}^{new} = w_{lij}^{old} + 2\eta(y_i^k - z_{li}^k) z_{(l-1)j}^{k}$$

With a small positive η, the summation weight updating in $\frac{\partial F(w)}{\partial w_{lij}}$ will be the continuous changes of weight vector that reduces the error function($F(w)$) in the fashion of sliding downhill on the error function surface.

5.1.1 The Problems of the Backpropagation

Since the back propagation training algorithm was introduced in 1985[26], several improvement techniques to the backpropagation have been invented to accelerate the training processes. Among these techniques, momentum weight change, the sign change in individual steps, selective weight initialization, weight decay and so forth are very well-known. Most of the above improvement approaches are mainly derived from the ideas of correcting the problems of steepest descent and ravines of the error functions[32] and they indeed succeeded in accelerating the backpropagation training algorithm at certain degrees. However there are less satisfactions from the techniques mentioned above when we want the improved backpropagation training algorithm to work not only at reducing the output error faster but also searching the weight space exhaustively for all training data patterns to be trained(in other words the training algorithm will end only when all outputs of the train patterns are in the small range of error tolerance). Recently there has been an improvement technique for backpropagation training algorithm from another school of thought, the approach comes from the idea of utilizing only the selective train patterns to facilitate the speedup and promote the complete learning. *Gradual Increase in Accuracy* is the training algorithm that follows this approach and it is the forerunner of Epoch Gradual Increase in Accuracy training algorithm.

5.2 Gradual Increase in Accuracy

This improvement technique for backpropagation training algorithm was firstly introduced by Ravi Kothari, Powsiri Klinkhachorn and Roy S. Nutter in 1991[20]. The approach is that we firstly force the training algorithm to train the neural network

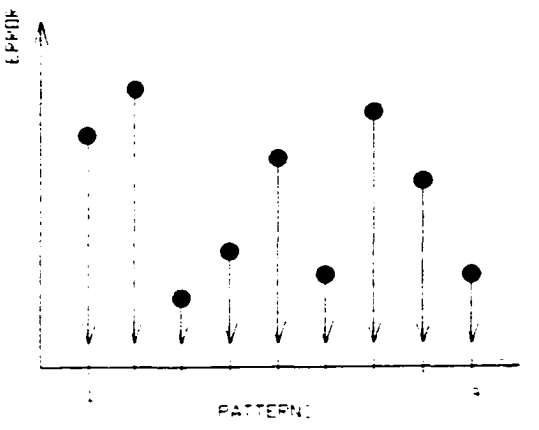


Figure 5.3: All training patterns are used for training with single accuracy target.

at low level of accuracy(large intermediate tolerance) using only one train pattern at every epochs. After the neural network learned all train patterns at the intermediate tolerance, the training algorithm increases the accuracy level (reduce the intermediate tolerance) and restarts the training process again(see figure 5.4). The experimental results using Gradual Increase in Accuracy training algorithm (using the same weight updating rules and momentum weight change as the one used in backpropagation training algorithm) done by Kothari, Klinkhachorn and Nutter showed that the improvement technique could result in complete learning while the algorithm accelerates the training process by 2~5 times comparing with the backpropagation alone.

5.2.1 The Algorithm

For illustration, we explain Gradual Increase in Accuracy train algorithm (GIA) with pseudo code in figure 5.5. This algorithm takes **PAT**[N] as its inputs and the output of the algorithm is the final updated weight set of neural network.

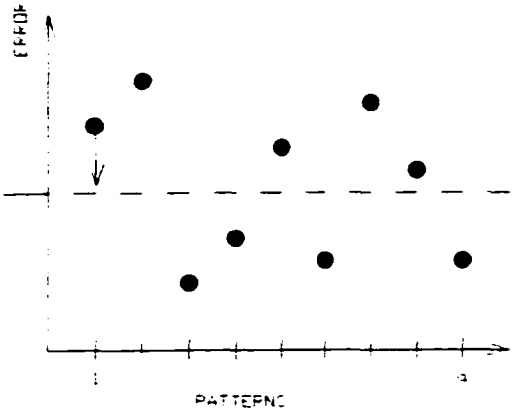


Figure 5.4: Gradual Increase in Accuracy Technique.

5.3 Epoch Gradual Increase in Accuracy

As we mentioned previously that Gradual Increase in Accuracy training algorithm(GIA) uses only one train pattern at each epoches for an intermediate tolerance and once the train pattern was learned, the training algorithm obtains another train pattern for the training process. Consequently some learned train patterns(the learned patterns are the train patterns that produce error less than an intermediate tolerance) can be unlearned by the latter training and the training algorithm must iterate the training process until all train patterns are learned.

Using different policy for selecting train patterns, Epoch Gradual Increase in Accuracy training algorithm is the improved version of GIA, it starts the training process with all train patterns and withdraws any train patterns from the process if the train patterns are learned up to an intermediate tolerance at each epochs. Eventually the training process at the specific intermediate tolerance ends when there is no train patterns left to be trained. And if the intermediate tolerance is not less than or equal to

```
input
        ITOLER  : intermediate tolerance
        DECRT   : decrement of intermediate tolerance
        FTOLER  : final tolerance
        PAT     : array of the train patterns
output
        the trained neural network
```

```
0.  procedure GIA
1.  begin
2.          while ITOLER ≤ FTOLER do
3.          begin
4.                  repeat
5.                  begin
6.                          k ← 0;
7.                          i ← 0;
8.                          while i < N do
9.                          begin
10.                                 bf TRAIN_SINGLE_PAT(PAT[i], ITOLER);
11.                                 i ← i + 1;
12.                         end
13.                 end while k > 0 do
14.                 ITOLER ← ITOLER−DECRT;
15.         end
16.  end
```

Figure 5.5: The algorithm of Gradual Increase in Accuracy

```
input
      ITOLER  : intermediate tolerance
      PAT[i]  : array of the train patterns
output
      the neural network weight updating
```

```
0.  procedure TRAIN_SINGLE_PAT
1.  begin
2.        flag ← 0;
3.        while flag = 0 do
4.        begin
5.              FORWARD_PASS( PAT[i] );
6.              ERROR ← ABS( NN_OUT−PAT[i].OUT );
7.              if( ERROR < ITOLER ) then
8.              begin
9.                    k ← k + 1;
10.                   BACKWARD_PASS( PAT[i] );
11.                   continue;
12.             end
13.             flag ← 1;
14.       end
15.  end
16.  end
```

Figure 5.6: The algorithm of training a single pattern

the final tolerance, the intermediate tolerance will be reduced gradually and the new cycle of training process will be started over until the final tolerance is reached(see figure 5.7).

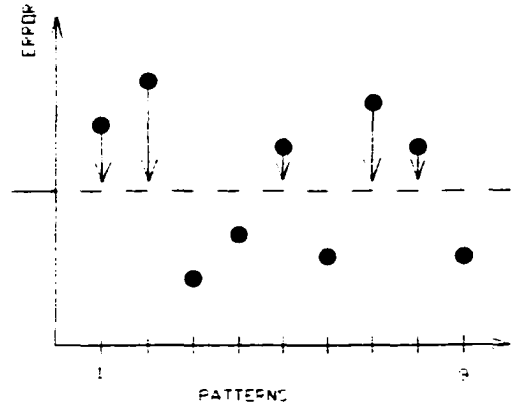


Figure 5.7: Epoch Gradual Increase in Accuracy Technique

5.3.1 The Algorithm

For better illustration, Epoch Gradual Increase in Accuracy training algorithm(EGIA) is explained with pseudo code in figure 5.8. This algorithm takes **PAT**[N] as its input and the output of the algorithm is the final updated weight set of neural network that produces overall error lower than the requirement.

In figure 5.9, procedure **SELECT_TRAIN_PAT** is called by procedure EGIA to calculate the errors of all train patterns in **PAT** and returns **TPAT** and n. **TPAT** is an array of train patterns that have error greater than the intermediate tolerance, and n is the number of train patterns in **TPAT**.

```
input
        ITOLER  : intermediate tolerance
        DECRT   : increment of intermediate tolerance
        FTOLER  : final tolerance
        PAT     : array of all train patterns
        TPAT    : array of the train patterns being used
output
        the trained neural network
```

```
0.  procedure EGIA
1.  begin
2.      while ITOLER ≤ FTOLER do
3.      begin
4.          k ← SELECT_TRAIN_PAT( PAT, N, TPAT );
5.          while k > 0 do
6.          begin
7.              repeat
8.              begin
9.                  i ← 0;
10.                 while i < k do
11.                 begin
12.                     FORWARD_PASS( TPAT[i] );
13.                     BACKWARD_PASS( TPAT[i] );
14.                 i ← i + 1;
15.                 end
16.                 k ← SELECT_TRAIN_PAT( TPAT, k, TPAT );
17.             end while k > 0 do
18.             k ← SELECT_TRAIN_PAT( PAT, N, TPAT );
19.         end
20.         ITOLER ← ITOLER - DECRT;
21.     end
22. end
```

Figure 5.8: algorithm of Epoch Gradual Increase in Accuracy

```
input
      PAT   : array of the train patterns being exploited
      MSIZE : the number of all patterns in PAT
output
      TPAT  : array of the train patterns having ERROR
              more than the intermediate tolerance
      n     : maximum number of patterns in TPAT
```

```
0.  procedure SELECT_TRAIN_PAT
1.  begin
2.        i ← 0;
3.        n ← 0;
4.        while i ≤ MSIZE do
5.        begin
6.              FORWARD_PASS( PAT[i] );
7.              ERROR ← ABS( NN_OUT − PAT[i].OUT );
8.              if( ERROR > ITOLER ) then
9.              begin
10.                   TPAT[n] ← PAT[i];
11.                   n ← n + 1;
12.             end
13.             i ← i + 1;
14.       end
15.       return n;
16. end
```

Figure 5.9: algorithm of selecting the train patterns

5.3.2 The Analysis

Because Epoch Gradual Increase in Accuracy training algorithm(EGIA) uses a group of train patterns(usually more than one) that the neural network does not yet learn at intermediate tolerance in every epochs, the training algorithm can achieve the neural network weight change in the direction which maximizes the reduction rate of the error function from all train patterns exploited at particular intermediate tolerance. This claims is supported by the interpretation of the gradient described in [14].

$$\nabla_w F(w) = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \nabla_w F_k(w)$$

In ordinary backpropagation training algorithm, N is the total number of trian patterns available(N is less than infinity) therefore the approximation of error gradient for backpropagation algorithm is

$$\nabla_w F(w) = \frac{1}{N} \sum_{k=1}^{N} \nabla_w F_k(w)$$

Under guiding from this approximation of the gradient, the weight updating is always moved in the direction that maximizes the reduction of the system error function according to the availability of the train patterns. In other words the backpropagation training algorithm searches the neural network weight space intelligently through the guide of gradient of the error function. However the quality of moving the weight space closely to the gradient is *not* preserved in Gradual Increase in Accuracy training algorithm because the algorithm is forced to train the network from a single train pattern until all train patterns generate errors less than the intermediate tolerance. Consequently the approximation of the gradient descent for the Gradual Increase in

Accuracy becomes

$$\nabla_w F(w) = \nabla_w F_k(w) \tag{5.4}$$

where k is the train pattern index.

Probably $\nabla_w F(w)$ in refeq:gia is the worst estimation of the error function gradient(if there are more than one training patterns available in each epoch). In order to combine the qualities of the maximum reduction of the system error function from the ordinary back propagation training algorithm and the complete learning from the Gradual Increase in Accuracy, Epoch Gradual Increase in Accuracy training algorithm is invented, its improvements come from exploiting all train patterns when the training process starts however the algorithm withdraws any train patterns that the network already learned up to the intermediate tolerance from the training process at every epochs. With this modification, the Epoch Gradual Increase in Accuracy algorithm can update the weight space of neural network at maximum rate(according to the availability of train patterns) while it is still promoting the search in neural network weight space for the complete learning. In this case, the gradient descent of the Epoch Gradual Increase in Accuracy is

$$\nabla_w F(w) = \frac{1}{M} \sum_{k=1}^{M} \nabla_w F_k(w)$$

where M is the number of training pattern used for training at intermediate tolerance for which $1 \leq M \leq N$.

5.4 The Performance Tests

In order to measure and compare the performance and specific advantages of Epoch Gradual Increase in Accuracy training algorithm(EGIA) with Backpropagation with

momentum weight change training algorithm(BPM) and Gradual Increase in Accuracy(GIA) training algorithm, we test all three algorithms on two problems. The first test is to compare speed and generalization of all three algorithms for the function approximation problem. The function used in this test is Mackey-Glass time series which is widely used as a benchmark for testing the neural network training algorithm. The second test is to compare speed and generalization of all three algorithms for classification problem that is classifying a single numeral from a pool of handwritten numeral images.

It is very important for every scientific tests that all principle parameters and environment of the tests must be defined explicitly. The followings are the test parameter, computer hardware, software we used in both performance tests.

Hardware

CPU: AMD-K6 166 Mhz

Memory: 32 Mbytes

Software

Operating System: Linux Redhat 5.1

Compiler: g++(GNU project C++ Compiler)

5.4.1 Mackey-Glass Time Series

The Mackey-Glass Time Series investigated in this test is generated from the iterative equation shown below[33].

$$x(k+1) - x(k) = \frac{\alpha x(k-\tau)}{1+x^{\gamma}(k-\tau)} - \beta x(k) \tag{5.5}$$

where the parameters α, β, γ are set as follows:

$$\alpha = 3, \quad \beta = 1.0005, \qquad \gamma = 6, \quad \tau = 3$$

In this test we set the first four initial steps to 1, 0.5, -0.5 and -0.8 then we reiterate the calculation for 254 more steps from equation 5.5. The train and test patterns are generated through the sliding window of size 4 on the series which means that we use a sequence of five contiguous number from the Mackey-Glass time series where the first four numbers are the inputs and the fifth number is the target output. The target output is normalized between 0.05 and 0.95. We allocate 150 patterns from the first 154 numbers of the series for training. Similarly we allocate 100 patterns from the next 104 numbers from the series for testing.

The structure of the neural network is the single hidden layer feedforward network of 4 input nodes, 17 hidden nodes, and a single output nodes. Every nodes in the networks has Sigmoid activation function and all links between them are initialized with random numbers range between -0.5 to +0.5.

The results of the test are the averages number of passes(forward and backward) which each training algorithms takes to bring the root mean square error(rmse) down to 0.05. The reason for using the number passes instead of the number of epoches is that GIA and EGIA training algorithms do not usually have the number of forward passes equal to the number of backward passes. We run every tests 5 times to get the average passes for each tests. Before being tested, the neural network is initialized with different set of random numbers on all of its links. The performance tests are repeated on six different learning rates 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9 and the

momentum weight change is set to 3 times the learning rate. The decrement of intermediate tolerance was fixed to 0.2 for every tests we run.

5.4.1.1 Analysis of Test Results

From the test results in the table(see figure 5.10), we have strong evidences indicating that EGIA is superior than GIA and BPM in speed, reliability, and accuracy in generalization over a wide range of learning rates

First let's look at the training speed of all three training algorithms, we can obviously see that the speed up ratio between BPM and EGIA is from 1.25 to 3.81 meanwhile the speed up ratio between GIA and EGIA is from 1.96 to 4.32. According to the test results, we are confirmed that EGIA is thoroughly faster than GIA and BPM in all learning rates used in the experiment.

The reliability of EGIA is also another quality which we should look at carefully. As it appears in the test results, BPM has only four successes in finishing the training process to bring the rmse down to 0.05 in 500,000 passes at learning rate = 0.8 and it has only one success in doing so at learning rate = 0.9. A similar trend also appears for GIA reliability, it has only one success in training at learning rate = 0.9. Contrary EGIA has all five successes in every learning rates. According to this information, we are confirmed that EGIA reliability to train the neural network for high accuracy output is higher than the reliabilities we got from BPM and GIA.

In addition the test results also show that EGIA has the highest accuracy in generalization in almost all learning rate. This quality of EGIA is also very important because what we also expect to have from the training algorithms is generalization, the neural network ability to response correctly to the unseen data.

	LR	0.4	0.5	0.6	0.7	0.8	0.9
BPM	*FW*	10504	8868	9080	8895	26210	38469
	BW	10504	8868	9080	8895	26210	38469
	TOTAL	21008	17736	18160	17790	52420(4)	76938(1)
		1.08	1.04	1.25	1.46	3.81	3.31
	TRAIN(rmse)	0.050	0.050	0.050	0.050	0.050	0.050
	TEST(rmse)	0.050	0.050	0.051	0.050	0.050	0.053
GIA	*FW*	30077	21699	22841	26181	31861	76765
	BW	13815	5688	5713	6510	8039	23788
	TOTAL	43892	27387	28554	32691	39900	100553(1)
		2.26	1.60	1.96	2.68	2.90	4.32
	TRAIN(rmse)	0.050	0.050	0.050	0.050	0.050	0.050
	TEST(rmse)	0.050	0.050	0.050	0.050	0.051	0.048
EGIA	*FW*	12286	12419	11138	9633	10927	18317
	BW	7125	4659	3429	2556	2819	4942
	TOTAL	19411	17078	14566	12189	13746	23259
		1.00	1.00	1.00	1.00	1.00	1.00
	TRAIN(rmse)	0.050	0.050	0.050	0.050	0.050	0.050
	TEST(rmse)	0.049	0.050	0.050	0.049	0.048	0.049

Figure 5.10: The comparison test results of Mackey-Glass problem

The advantages of EGIA can be understood by analyzing the frequency distribution of train pattern errors of all three training algorithms. From histogram in figure 5.14 to figure 5.17, we can see that BPM tries to reduce the overall error by exploiting all train patterns, however there are times in training process when errors from few train patterns cannot be reduced. To further reduce the overall error, BPM must search in the new direction in weight space in order to find a new location where the error surface of the system leads to further error reduction. During the searching for a new location in the weight space, many train patterns that are learned by the neural network may become more unlearned, which means that the train patterns produce more error than they used to. At this moment in the training process, the overall error will rise and fall further more when the errors from more patterns are reduced, in general this rising and falling of overall error is called local minimum of overall error. The local minimum problem which normally happens in BPM training algorithm is the phenomena that delays the training process. This phenomena can be viewed from various behaviors of neural network depending on the analytical perspective. From our perspective, the local minimum problem occurs when majority of train patterns push the weight updating too much by their errors and leave few train patterns unlearned. To solve this problem, we emphasize on giving more training to the train patterns that have bigger errors. From figure 5.18 to figure 5.23, we can see that GIA and EGIA training algorithms adopt the approaches of selective training and works more on the train patterns producing larger error and the results are that both training algorithms can avoid the serious problem of local minimum. Further more we see that GIA training algorithm is slower than BPM for this type

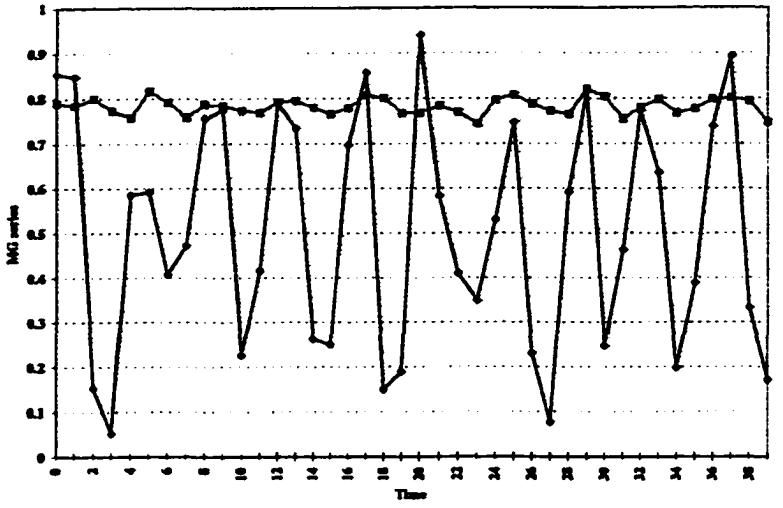
of test which is the function approximation problem. The explanation for the slow training of GIA is that it uses only one train pattern for training over and over in every epoches and that causes many train patterns to be more unlearned and GIA has to spend more time on training those train patterns.

5.4.2 Classification of Handwritten Numerals

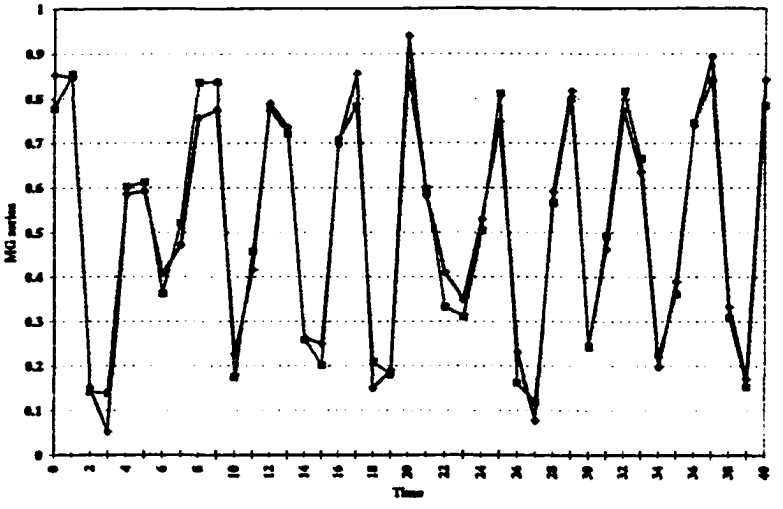
To demonstrate the abilities of EGIA and GIA in solving the incomplete learning, all three learning algorithms are used to train a neural network to classify the single numeral from our database of handwritten numeral images(the details of generating this database will be explained in chapter 6). From the database, we allocate 350 patterns for training and 150 patterns for testing. Each patterns consists of sixty-four inputs and one single output. The sixty-four inputs come from two image features, one of the features is angular feature and the other is Kirsch feature. The target output is binary number(0 or 1 used for classification).

The structure of the neural network is the single hidden layer feedforward network of 64 input nodes, 128 hidden nodes, and a single output node. Every nodes in the network has the same Sigmoid activation function and all links between them are initialized with random numbers range between -0.5 to +0.5.

Similar to the previous test, the results of this test are the average number of passes(forward and backward) which the backpropagation with momentum weight change training algorithm(BPM), Gradual Increase in Accuracy training algorithm(GIA), and Epoch Gradual Increase in Accuracy training algorithm(EGIA) spent to bring the root mean square error(RMSE) down to 0.05. Again we run every tests 5 times to get the average passes. Before being tested, the

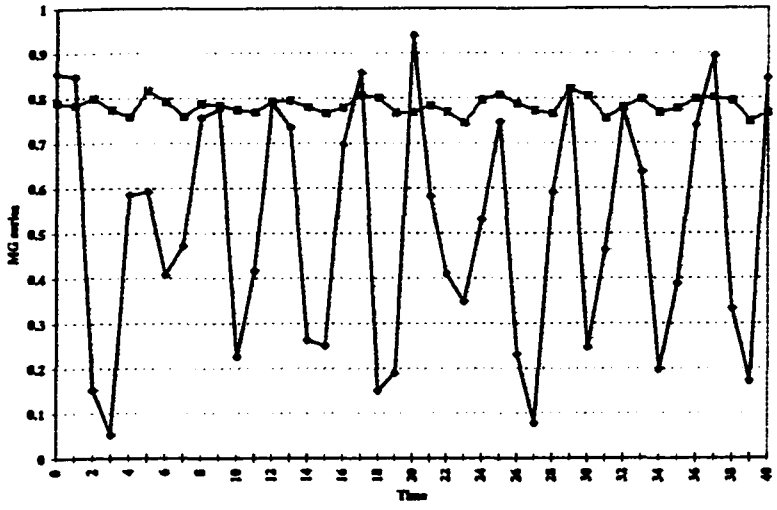


(a) Before the training, rmse = 0.374

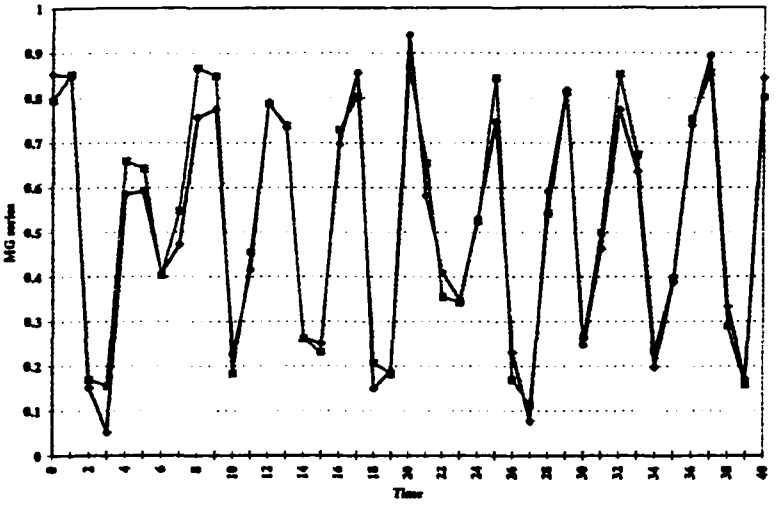


(b) After the training at 22894 passes, rmse = 0.050

Figure 5.11: The generalization of neural network trained by BPM

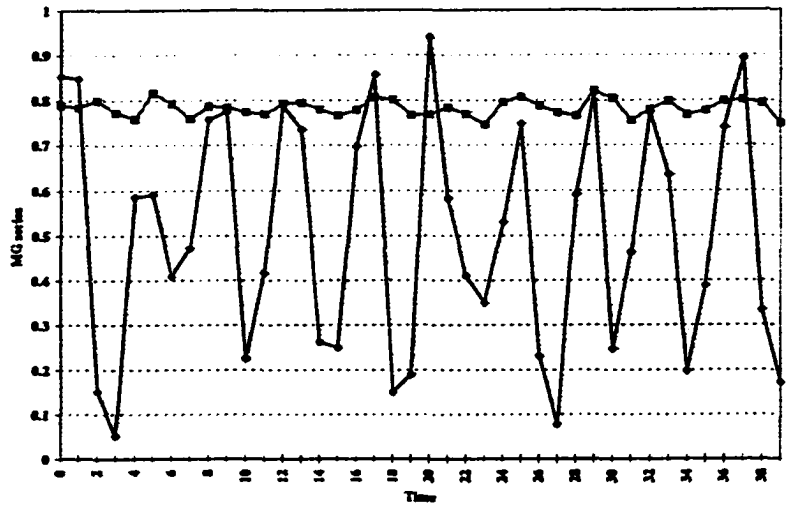


(a) Before the training, rmse = 0.374

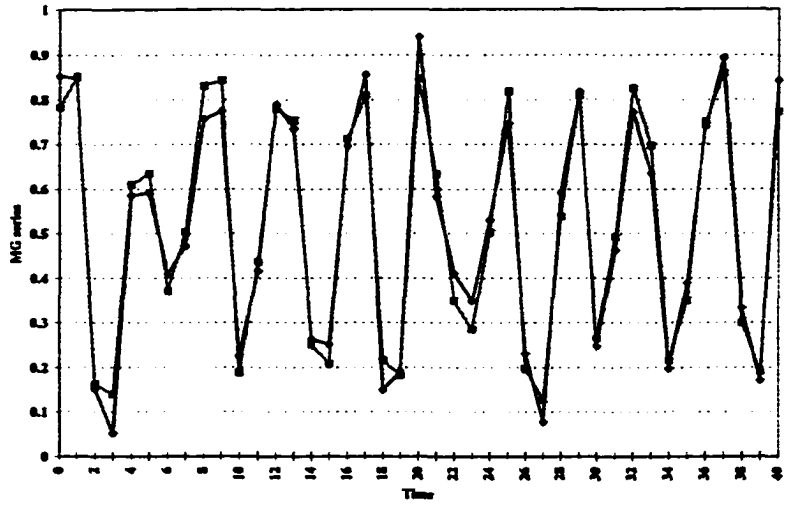


(b) After the training at 53026 passes, rmse = 0.049

Figure 5.12: The generalization of neural network trained by GIA

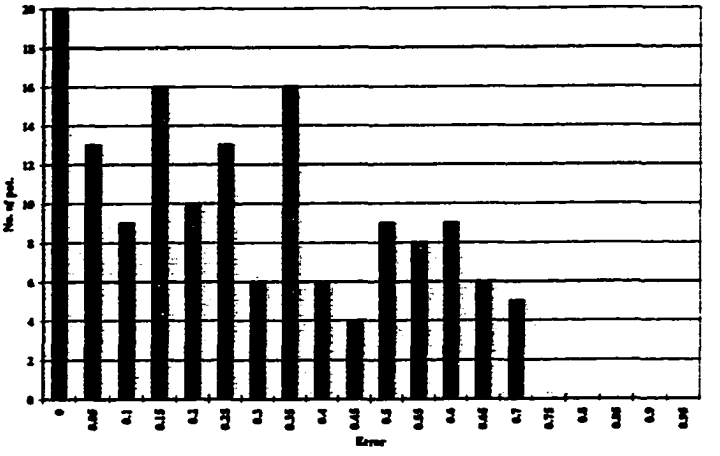


(a) Before the training, rmse = 0.374

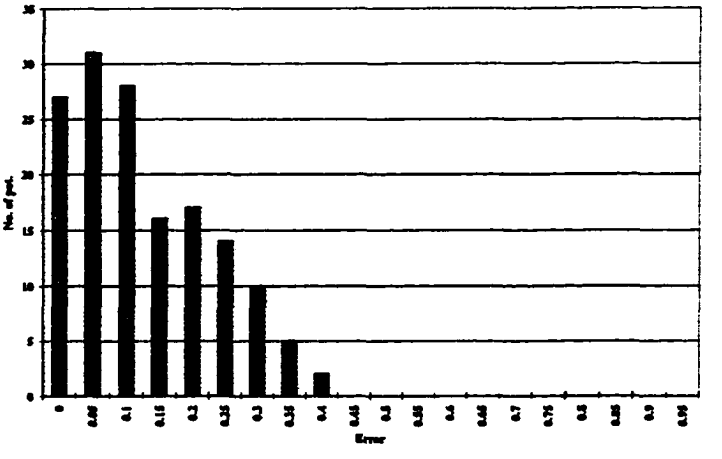


(b) After the training at 16289 passes, rmse = 0.049

Figure 5.13: The generalization of neural network trained by EGIA

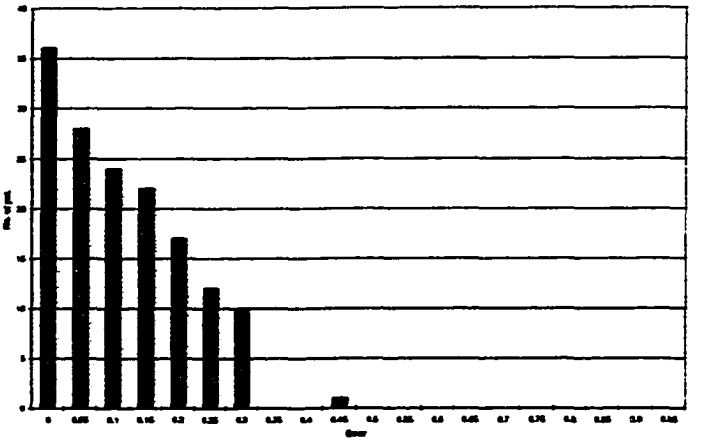


(a) At 0 passes, rmse = 0.374

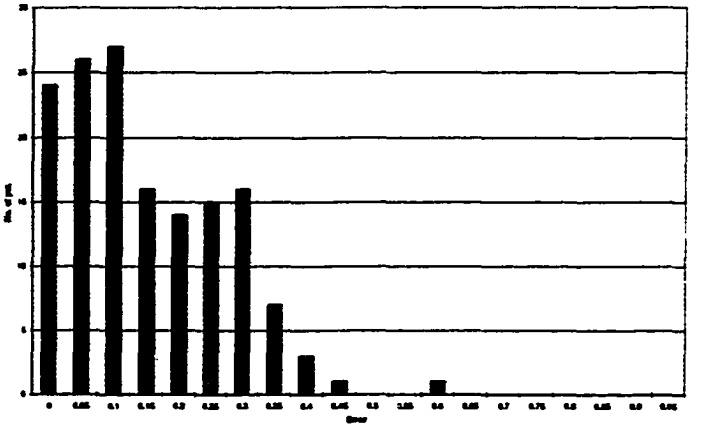


(b) At 6 passes, rmse = 0.250

Figure 5.14: The error distribution of neural network trained by BPM

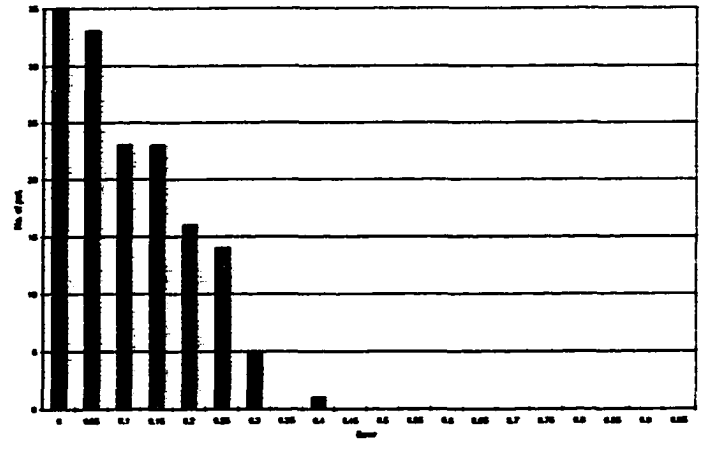


(a) At 300 passes, rmse = 0.170

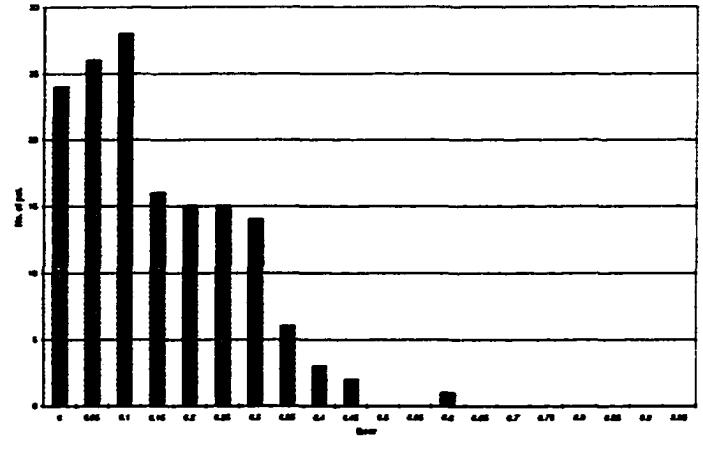


(b) At 400 passes, rmse = 0.211

Figure 5.15: The error distribution of neural network trained by BPM

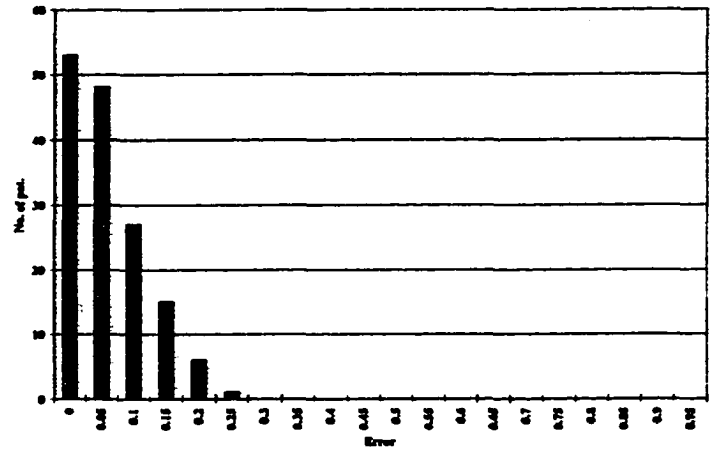


(a) At 500 passes, rmse = 0.162

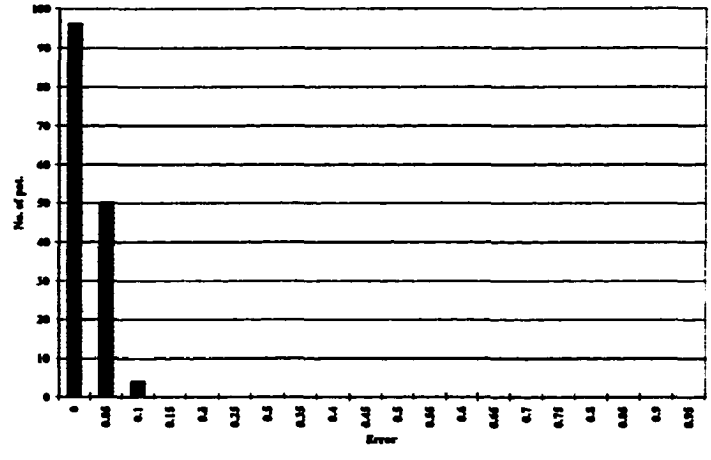


(b) At 700 passes, rmse = 0.211

Figure 5.16: The error distribution of neural network trained by BPM

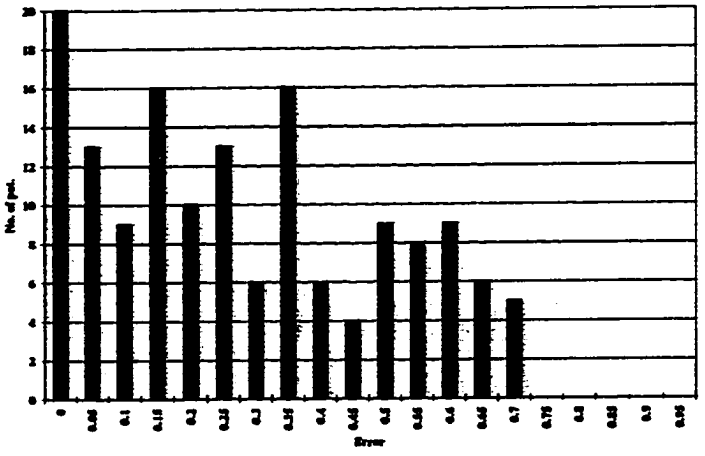


(a) At 3520 passes, rmse = 0.100

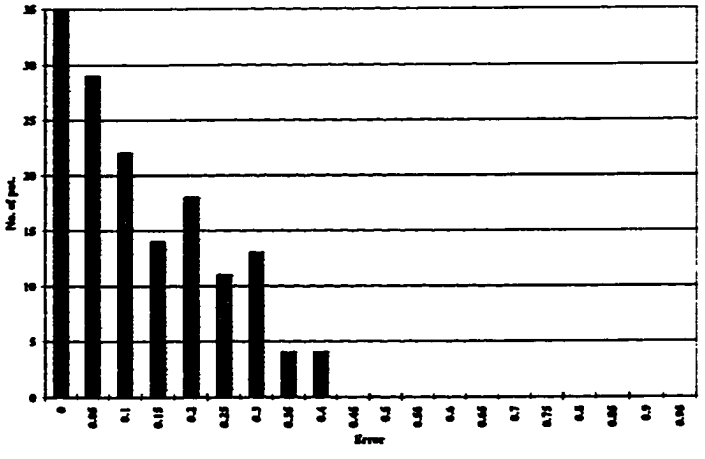


(b) At 22894 passes, rmse = 0.050

Figure 5.17: The error distribution of neural network trained by BPM

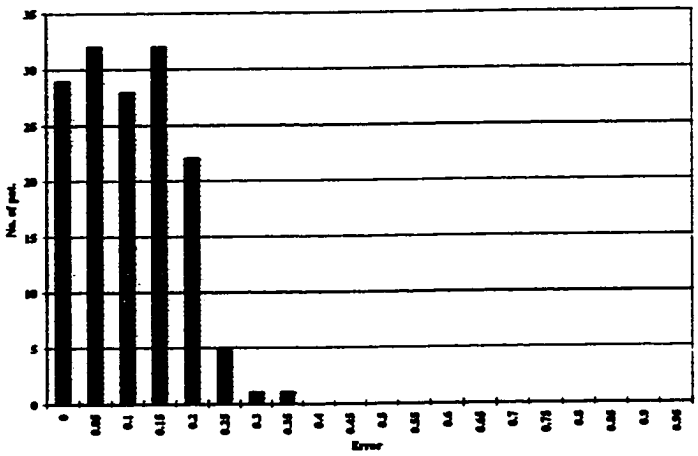


(a) At 0 passes, rmse = 0.374

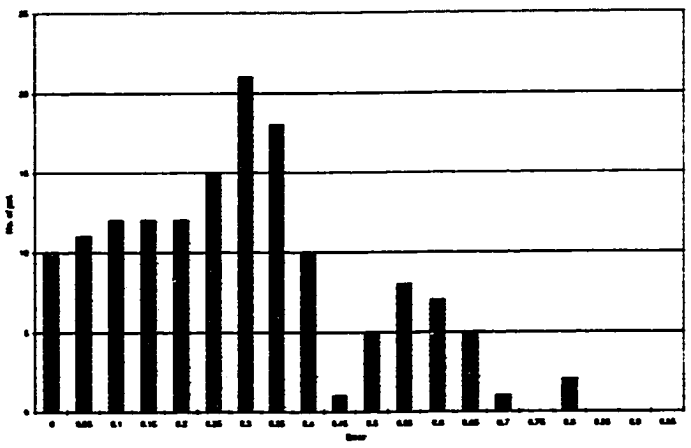


(b) At 7 passes, rmse = 0.250

Figure 5.18: The error distribution of neural network trained by GIA

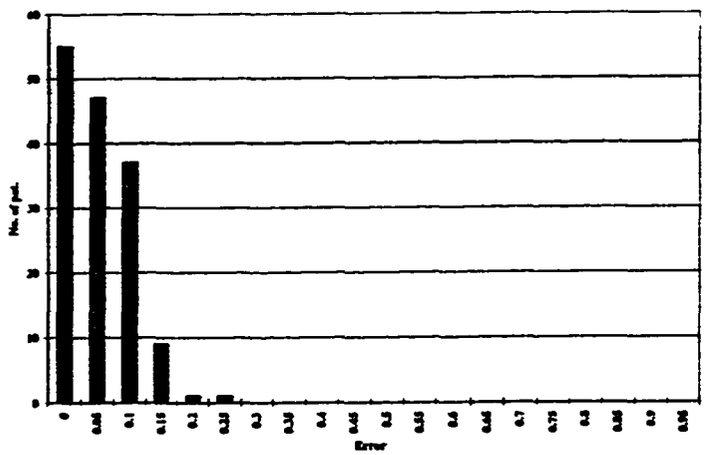


(a) At 546 passes, rmse = 0.150

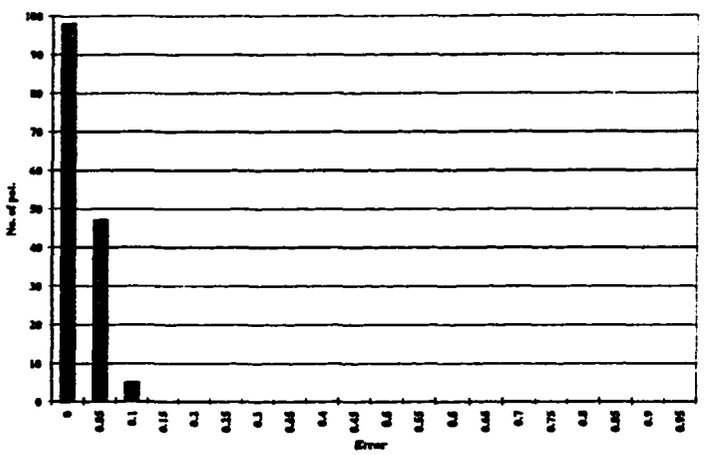


(b) At 1200 passes, rmse = 0.365

Figure 5.19: The error distribution of neural network trained by GIA

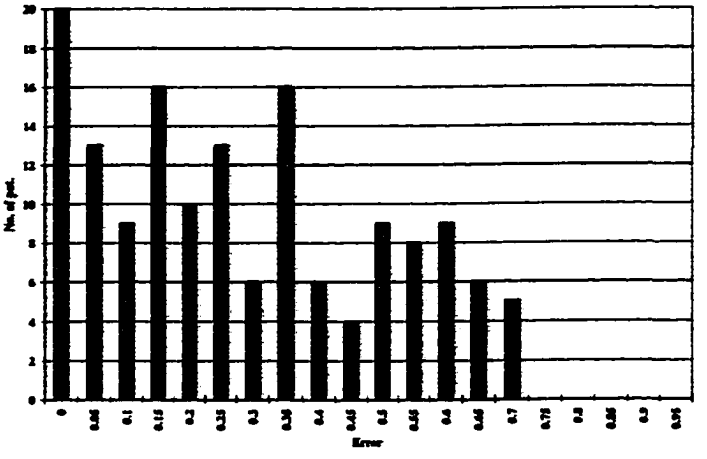


(a) At 5415 passes, rmse = 0.100

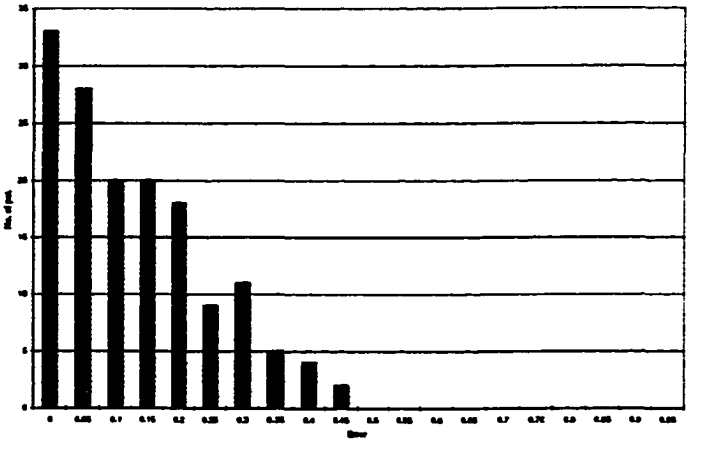


(b) At 53026 passes, rmse = 0.049

Figure 5.20: The error distribution of neural network trained by GIA

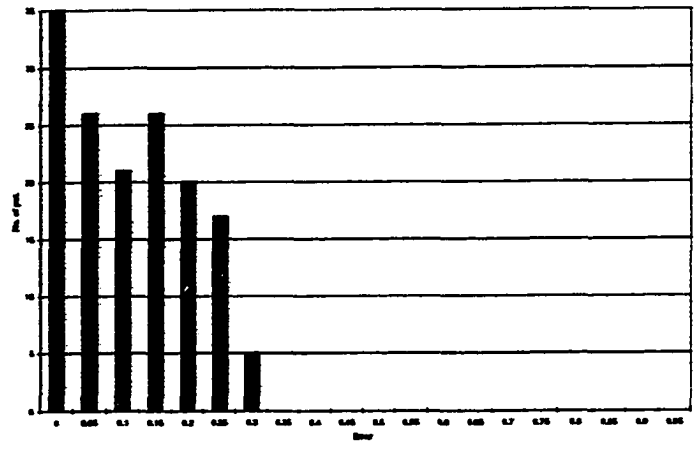


(a) At 0 passes, rmse = 0.374

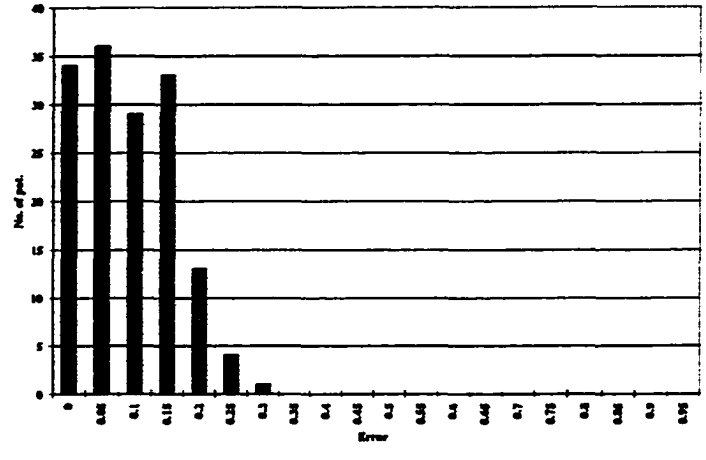


(b) At 100 passes, rmse = 0.196

Figure 5.21: The error distribution of neural network trained by EGIA

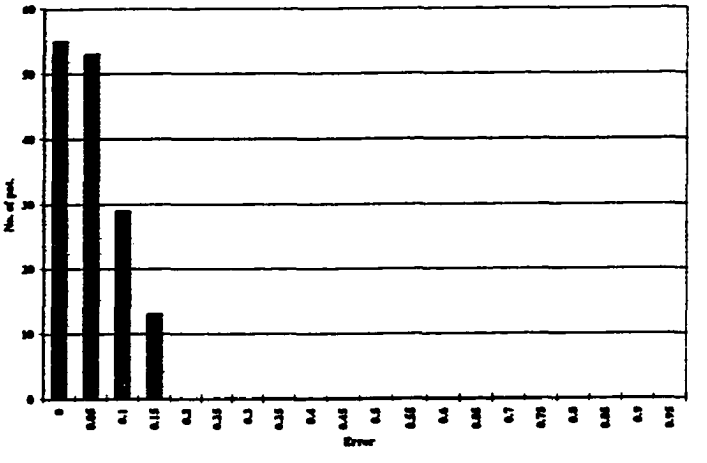


(a) At 400 passes, rmse = 0.168

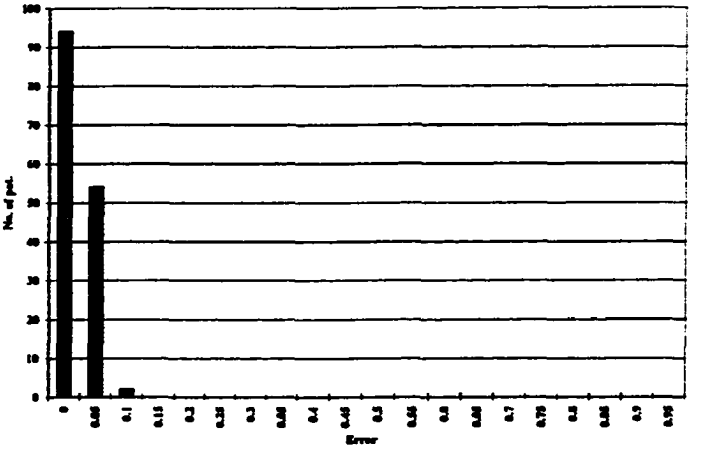


(b) At 1437 passes, rmse = 0.150

Figure 5.22: The error distribution of neural network trained by EGIA



(a) At 3672 passes, rmse = 0.100



(b) At 16289 passes, rmse = 0.049

Figure 5.23: The error distribution of neural network trained by EGIA

neural network is initialized with different set of random numbers on all of its links. The performance tests are repeated on four different learning rate 0.1, 0.2, 0.3, and 0.4 and the momentum weight change is always set to 3 times the learning rate. The decrement of intermediate tolerance is fixed to 0.2 for all tests.

5.4.3 Analysis of Test Results

From the test results on the table in figure 5.24, we can see that EGIA is the fastest meanwhile BPM is the slowest. The test results also confirm our speculations that EGIA and GIA will perform better than BPM on classification problems. In this test we can also see that the speedup of EGIA for this problem is much higher than its speedup in Magkey-Glass problem and the explanation to this phenomena is that EGIA and GIA do not have a severe problem of incomplete learning which BPM terribly suffers from.

This incomplete learning problem usually happens when the backpropagation training algorithm has to process a large number of train patterns of which the input dimension is high. By examine the error distribution graphs in figure 5.25 to figure 5.33 carefully we see that EGIA and GIA firstly spend more time learning the train patterns that have bigger error with respect to the intermediate tolerance then the intermediate tolerance is gradually reduced until the neural network reaches the final accuracy of the train process. Unlike the other two training algorithm, the back-propagation training algorithm has no policy to firstly correct the train patterns with bigger errors. This sometimes results in training the neural network to learn only the majority of train patterns and leave very few patterns unlearned. Clearly the

	LR	0.1	0.2	0.3	0.4
BPM	*FW*	24080	18620	73570	147700
	BW	24080	18620	73570	147700
	TOTAL	48160	37240	147140	295400(3)
		5.89	6.61	37.27	15.69
	TRAIN(350)	349	349	349	349
	TEST(150)	149	149	149	149
GIA	*FW*	11305	5508	4317	32401
	BW	929	263	830	29807
	TOTAL	12234	5771	5147	62208
		1.50	1.02	1.30	3.30
	TRAIN(350)	350	350	350	350
	TEST(150)	148	148	149	149
EGIA	*FW*	7160	5397	3685	11773
	BW	1017	234	263	7051
	TOTAL	8177	5631	3948	18824
		1.00	1.00	1.00	1.00
	TRAIN(350)	350	350	350	350
	TEST(150)	149	149	149	149

Figure 5.24: The comparison test result of handwritten numeral problem

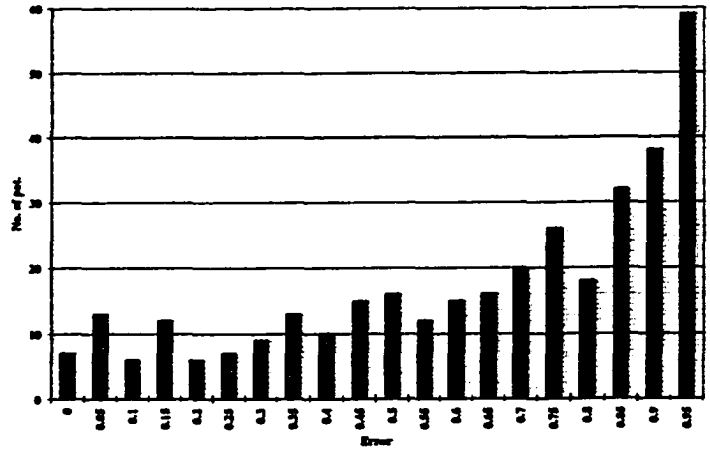
unlearned patterns cause a seriously slow training and undermine the generalization accuracy.

5.5 Summary

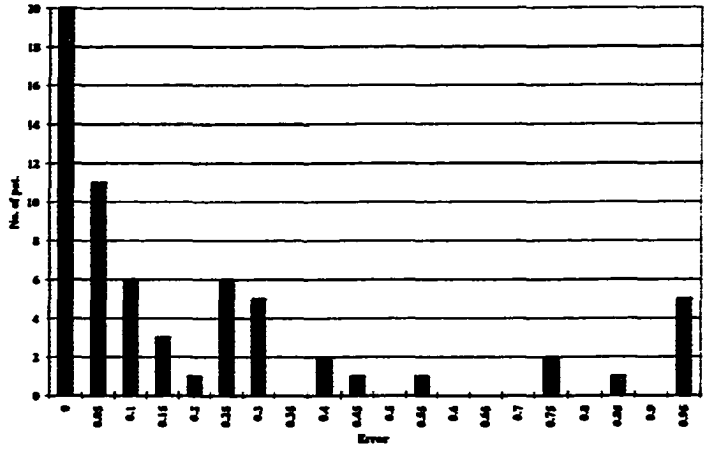
In this chapter we present an original training algorithm called Epoch Gradual Increase in Accuracy(EGIA). EGIA is the evolution of Gradual Increase in Accuracy training algorithm(GIA). The improvement of EGIA comes from the different policy in the selection of train patterns. Unlike GIA which uses only one train pattern at every epochs, EGIA uses a group of train patterns for training at particular intermediate tolerance. As a result of this different policy, EGIA can train the neural network faster and more reliable than GIA and Backpropagation with momentum weight change training algorithm(BPM).

According to the experimental results, we discover that EGIA can averagely finish the training 1.25 - 3.81 times faster than BPM and 1.92 - 4.32 times faster than GIA for the function approximation problem of Mackey-Glass series. And as far as the reliability concerns, EGIA finishes all five tests within 500,000 passes but BPM and GIA can finish only one test at learning rate = 0.9.

Continuing in the same direction, EGIA can averagely finish the training 5.89 - 37.27 times faster than BPM and 1.02 - 3.30 times faster than GIA for the classification problem of handwritten numeral images. Again EGIA finishes all five tests within 500,000 passes however BPM can finish only three tests at learning rate = 0.4.

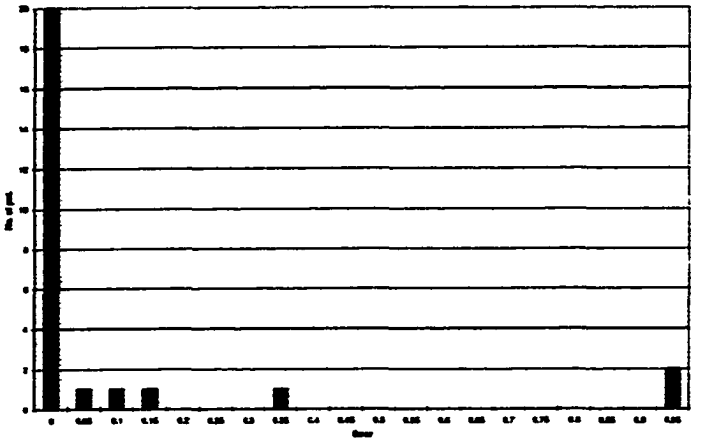


(a) At 0 passes, rmse = 0.722

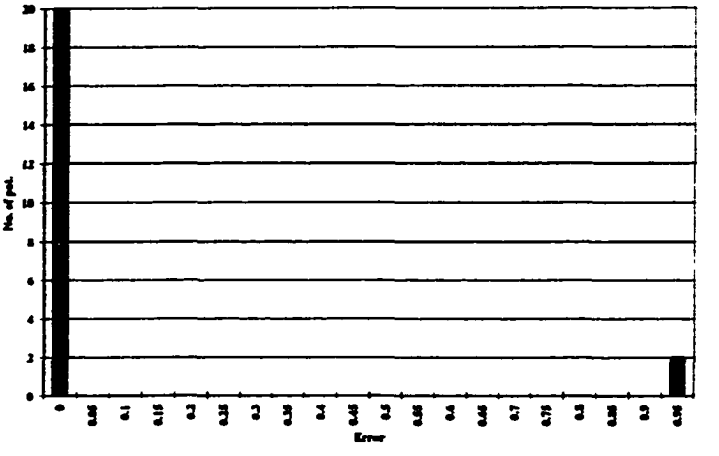


(b) At 1400 passes, rmse = 0.250

Figure 5.25: The error distribution of neural network trained by BPM

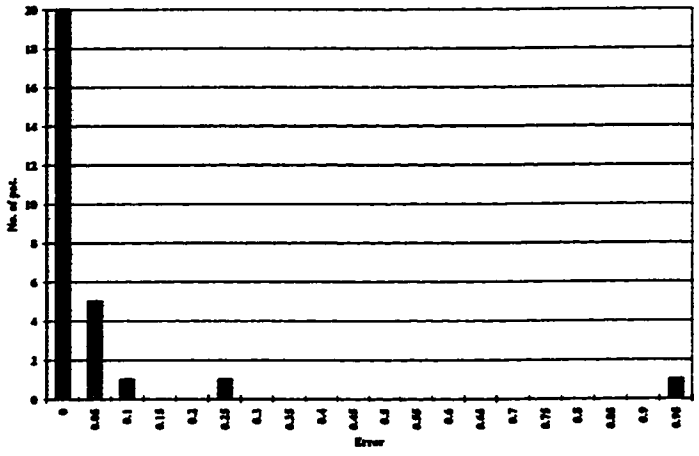


(a) At 5000 passes, rmse = 0.079

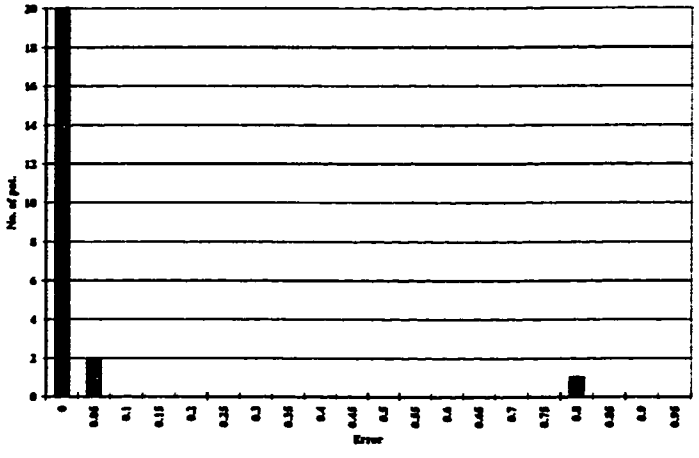


(b) At 8400 passes, rmse = 0.150

Figure 5.26: The error distribution of neural network trained by BPM

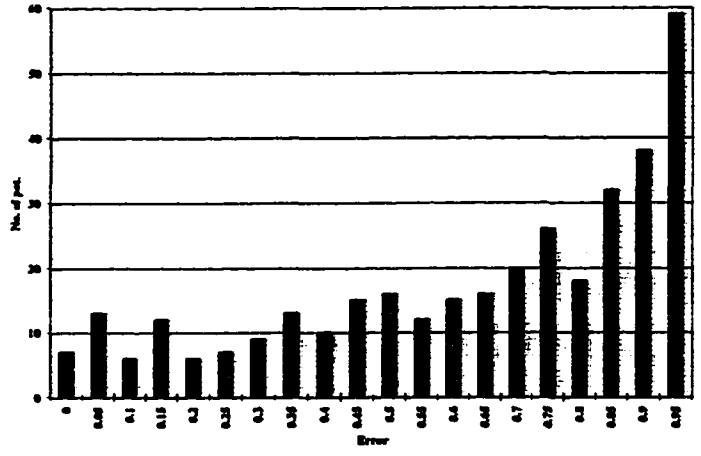


(a) At 11900 passes, rmse = 0.100

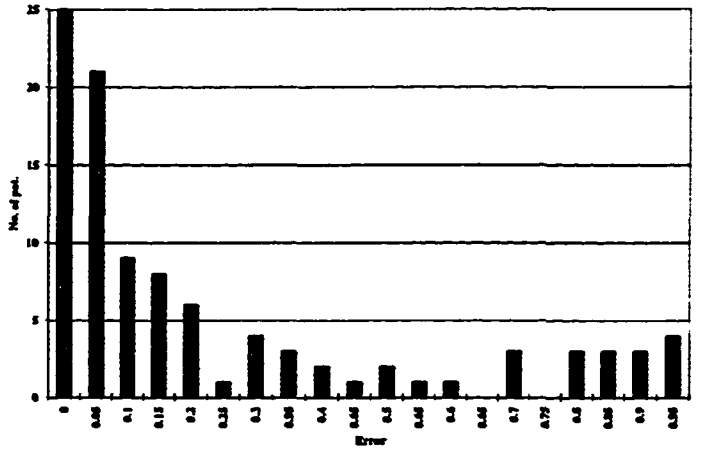


(b) At 312900 passes, rmse = 0.044

Figure 5.27: The error distribution of neural network trained by BPM

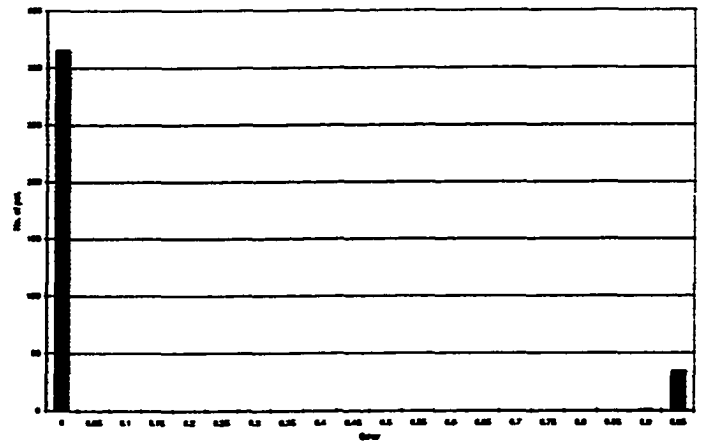


(a) At 0 passes, rmse = 0.722

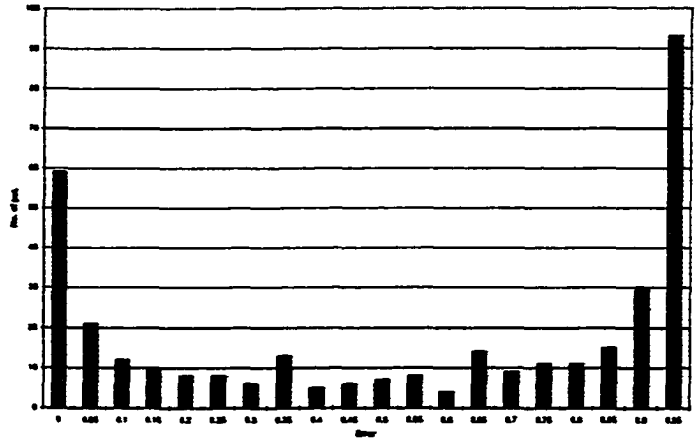


(b) At 4211 passes, rmse = 0.250

Figure 5.28: The error distribution of neural network trained by GIA

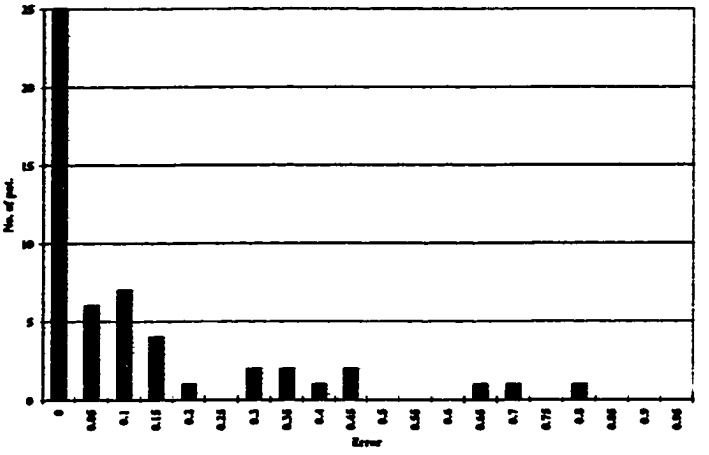


(a) At 56000 passes, rmse = 0.300

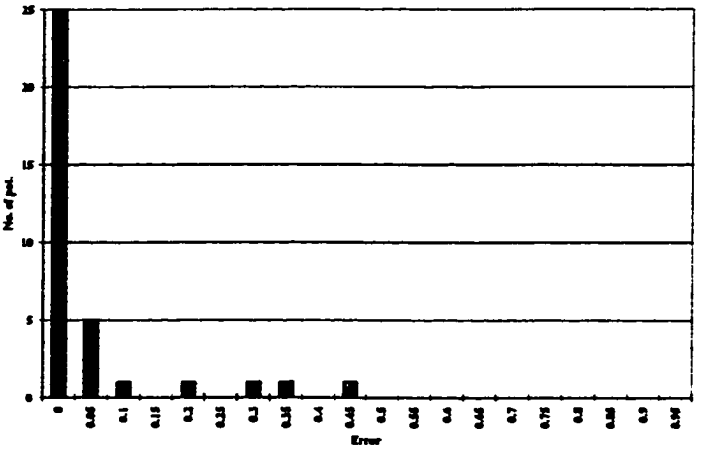


(b) At 57000 passes, rmse = 0.811

Figure 5.29: The error distribution of neural network trained by GIA

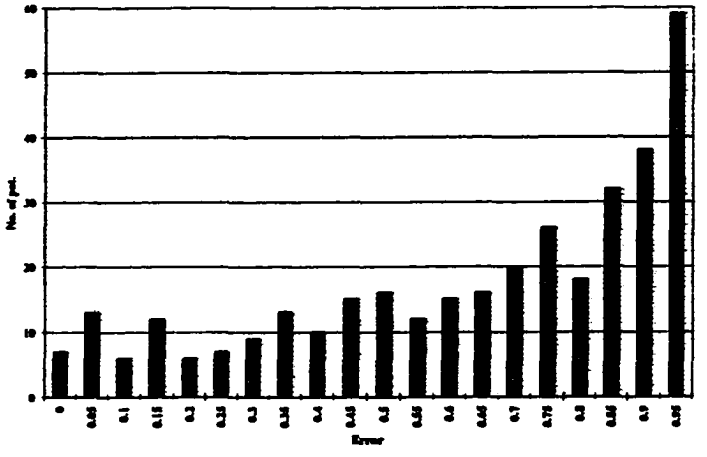


(a) At 59938 passes, rmse = 0.100

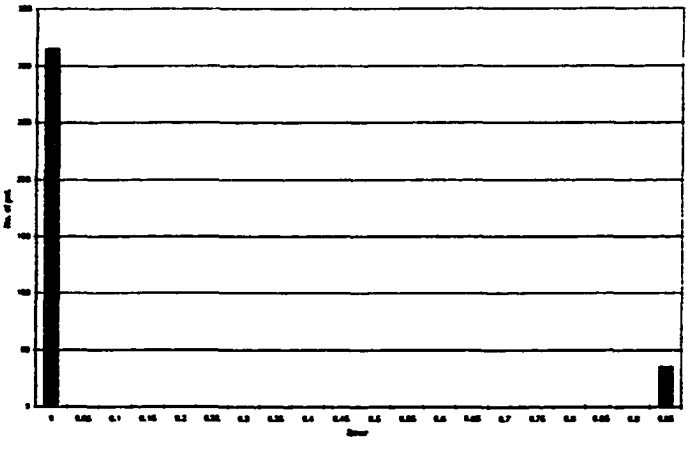


(b) At 62208 passes, rmse = 0.042

Figure 5.30: The error distribution of neural network trained by GIA

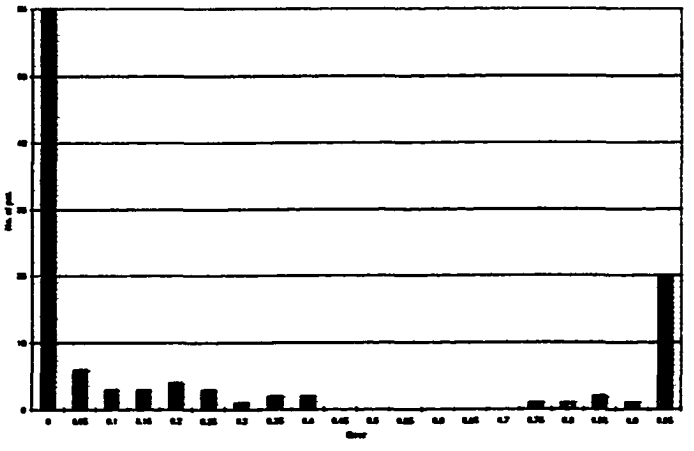


(a) At 0 passes, rmse = 0.722

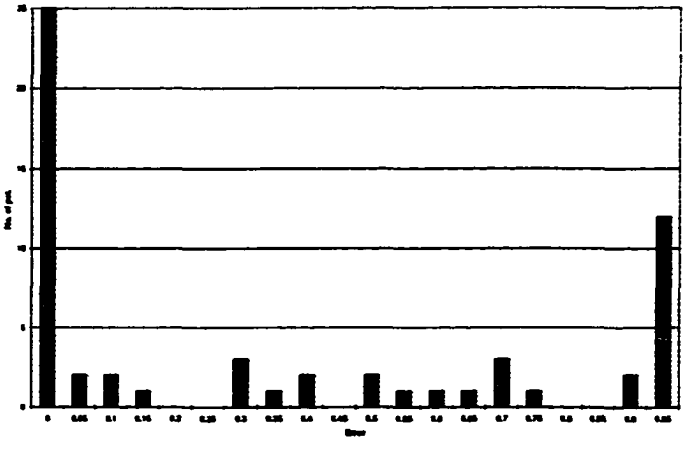


(b) At 800 passes, rmse = 0.316

Figure 5.31: The error distribution of neural network trained by EGIA

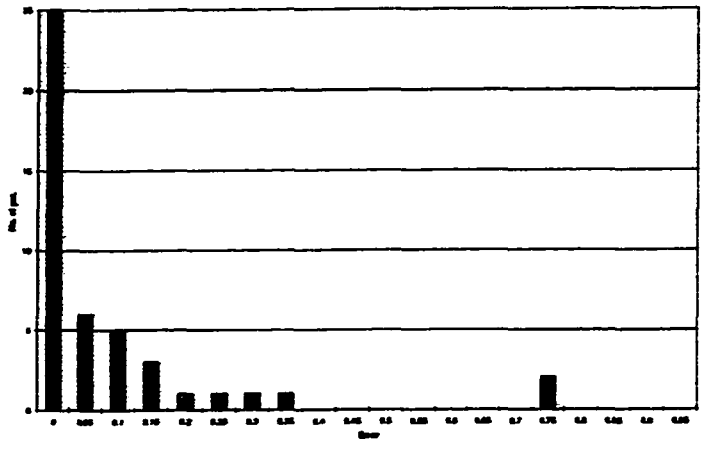


(a) At 1200 passes, rmse = 0.919

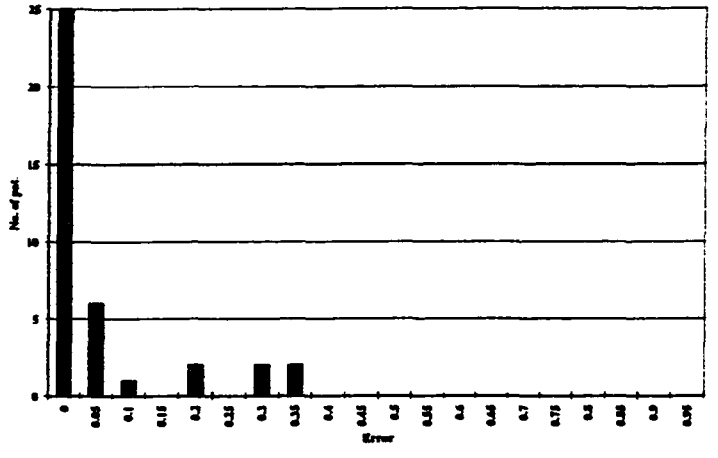


(b) At 1600 passes, rmse = 0.214

Figure 5.32: The error distribution of neural network trained by EGIA



(a) At 3800 passes, rmse = 0.094



(b) At 4300 passes, rmse = 0.044

Figure 5.33: The error distribution of neural network trained by EGIA

Chapter 6
Handwritten Numeral Recognition System

In this chapter we are ready to discuss the design and implementation of our first experimental prototype, the handwritten numeral recognition system. Base on the general design layout, our recognition system basically consists of three processes; preprocessing, feature extraction, and classification. In order to improve the system performance, we used the feature extraction process that generates deviating angular feature and Kirsch feature from numeral images(see section 4.1). The system classification process is a committee of multilayer feedforward neural network trained by Epoch Gradual Increase in Accuracy training Algorithm which we described in section 5.3. With these innovative designs, we test our system for performance measurements, later the test results are analyzed and compared with many successful handwritten numeral recognition systems invented previously by other researchers[6].

6.1 Preprocessing

In this experiment, we preprocess the 16 x 16 binary bitmap image so that the image object is centralized and normalized in height. Please note that these preprocessings are not actually required by the deviating angular feature however they are necessary for Kirsch feature since Kirsch feature is sensitive to translation variance and scaling variance in the image data (we discussed Kirsch feature in section 4.1.2).

6.2 Feature Extraction

In this process we generate deviating angular feature and Kirsch feature from the preprocessed image. The output of the process will later be used as the combined features sent to the classification process where half of the features is from deviating angular feature and the other half is from Kirsch feature.

The advantage of combining these two features to represent the numeral images is that they can mutually compensate and enhance the value of image signatures. Despite many advantages of the deviating angular feature, the feature extraction techniques generating the image profiles have their own limitation because the algorithms extracting image profiles cannot reach the internal details of images therefore the deviating angular features we get from the feature extraction process is only the features of image external profiles. Contrary to the deviating angular feature, Kirsch feature which is much more sensitive to translation and scaling variations of image data can capture the internal curvatures of images even the curvatures are in the closed image profile. In addition the combination of two features give the information about images from different aspects to the neural network classifier because deviating angular feature is the feature indicating sequentially relative angular changes of direction in image profiles but Kirsch feature is the feature for measuring the degree of slopes at a pixel point in image curvatures.

The combination of the two features bases on the above reasons is expected deliver the high valuable information to the classifier that performs recognition which is considered as the most important quality of the handwritten numeral recognition system.

6.2.1 Deviating Angular Feature

The deviating angular features used in this implementation are generated from two profiles of binary bitmap images(see sections 4.1). Next we place eighteen marking points on each image profiles. From these eighteen marking points we calculate the deviating angular feature of sixteen deviating angles. Therefore the deviating angular features from two image profiles provide the first 32 valuable inputs to the neural network classifier.

6.2.2 Kirsch Feature

As we mentioned earlier, horizontal and vertical Kirsch features extracted from handwritten numeral images are also used in our experimental system. The result of either horizontal or vertical Kirsch feature extraction is a set of 16x16 numbers in two-dimensional array(from a 16x16 binary bitmap image). We then coarsen these 16x16 number array down to 4x4 numbers in two-dimensional array which finally gives us a sequence of sixteen numbers representing Kirsch feature in on one direction. Since the value of Kirsch feature is in the range of 0 to 15 and its highest value is about four times the magnitude of deviating angles which is between $-\pi$ to $+\pi$, we normalize the value of Kirsch features to be in the range of 0 to 7.5. Eventually horizontal and vertical Kirsch features provide another 32 valuable inputs to the neural network classifier.

6.3 Classification

In order to classify the numeral features effectively, we use a committee of multilayer feedforward neural networks for the final operational stage of the system. The classifier consists of 10 multilayer neural networks, each of these neural networks is

responsible for classifying a single numeral(see figure 6.1 and figure 6.2). The training algorithm for every neural networks is Epoch Gradual Increase in Accuracy(see section 5.3). The winner-take-all decision policy is used to decide which handwritten numeral the system is processing.

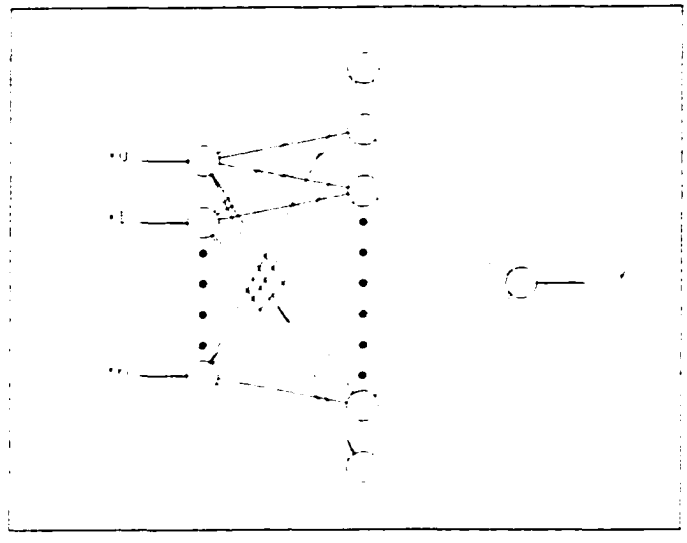


Figure 6.1: The multilayer neural network

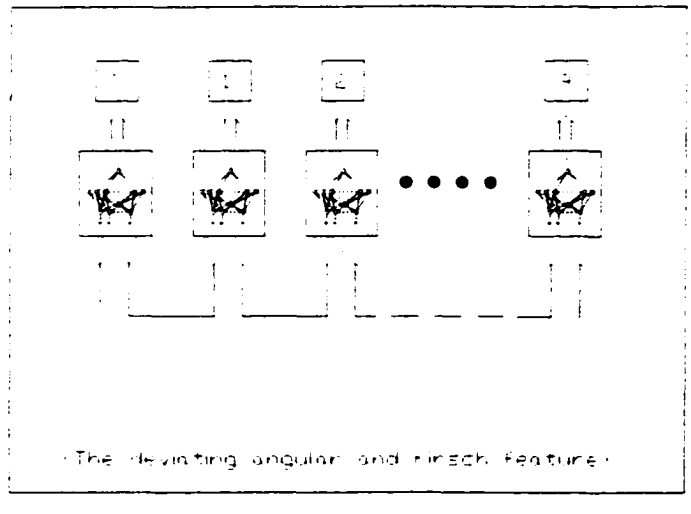


Figure 6.2: The committee of 10 multilayer neural networks

6.4 Database

We collected 500 handwritten numerals from the LSU student volunteers on the 16x16 grids. Then we manually translated these handwritten images into the binary bitmap images. The examples of forty handwritten numerals are shown in figure 6.3.

6.5 Test Parameters

The followings are the test parameters and the configuration of neural network classifier used in this experiment.

Epoch Gradual Increase in Accuracy(EGIA)

Learning Rate = 0.3

Momentum Rate = 0.9

Range of initial weights = (-0.5,+0.5)

Initial Intermediate Tolerance = 0.5

Decrement of Intermediate Tolerance = 0.02

The Structure of Each Subneural Network

Input Layer = 64 nodes

Hidden Layer = 128 nodes

Output Layer = 1 nodes

6.6 Test Results

From 500 handwritten numeral images, we randomly allocate 150 images for testing and 350 images for training. After running the tests on five different test files, the test of generalization shows that the system can achieve 98.7% of maximum recognition rate(the system classified correctly 148 images out of 150 test images) and 98.1% of average recognition rate.

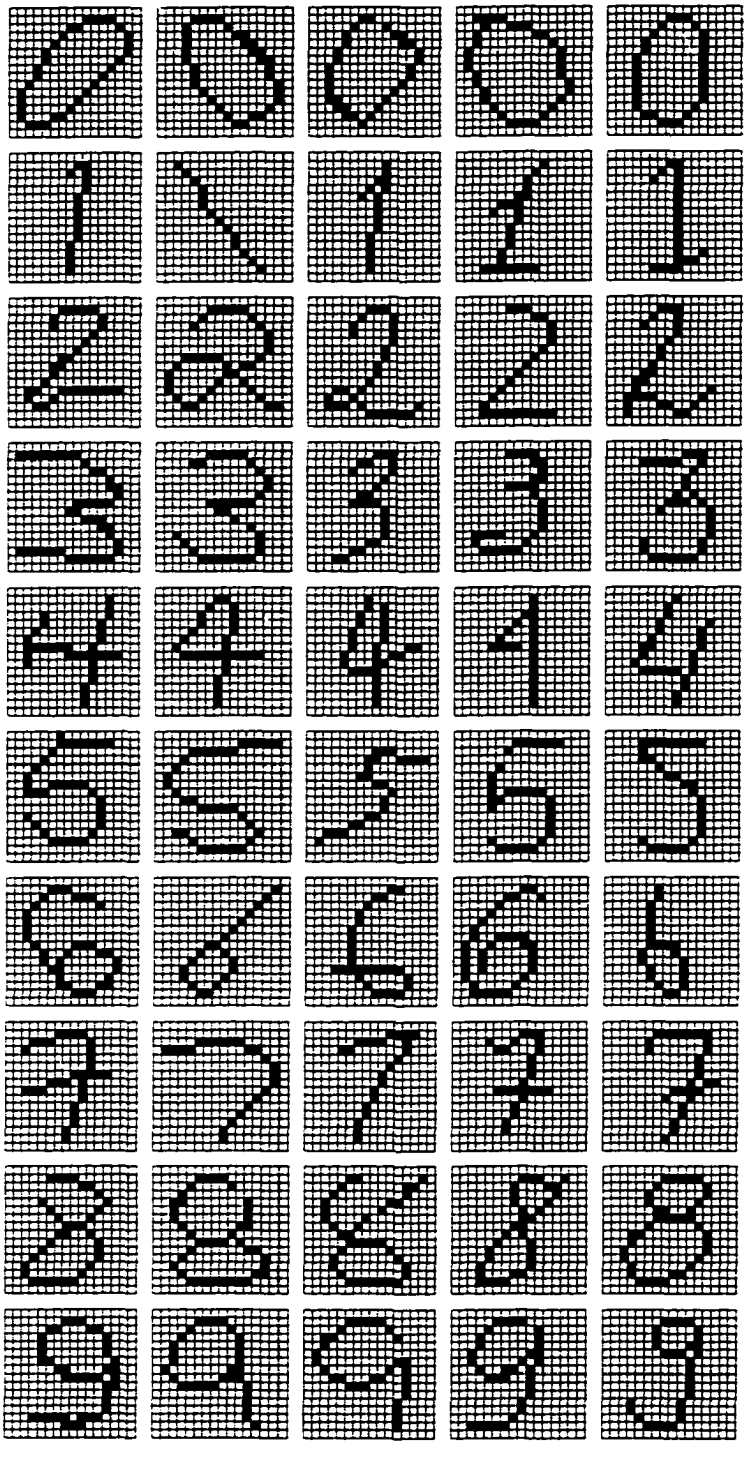


Figure 6.3: The 50 examples of handwritten numerals

METHODS	*FEATURE*	*CLASSIFICATION*	*MAX. RECOGNITION RATE*
Cohen	Twenty six features by mix approaches	Rule-based classifier and statistical approach	95.5%
Kageyu	Complex-log mapping	multilayer neural network trained by backpropagation	95.2%
Lee	Tracing sequence generating by a set of heuristic rules	Self-organizing neural network	97.0%
Cho	Kirsch feature, Complex Fourier , coefficients of outer contour and simple topological feature of inner contour	multiple multilayer perceptron and fuzzy consensus	97.4%

Figure 6.4: The experimental results of the other systems

6.7 The Analysis

After reviewing the experimental results and comparing them with the other important experiments done previously, we can point out many improvements of our handwritten numeral recognition system as the followings.

First the maximum recognition rate and the average recognition rate of our system are highest. Second our system uses less features for representing the abstracts of images without compromising the recognition ability. Practically using less feature means more simplicity for implementation and less computing power to operate since every images must go through the feature extraction process. Third our committee of multilayer feedforward neural networks provides the powerful classification machine. With winner-take-all consensus for final decision policy, the structure of our neural network classifier can be completely distributed in any standard computer networking. The distributed neural network classifier can be trained many times faster than the classifier operating in a single computer, and the complete distributed committee of neural network classifier can be ten times faster the one in single computer in operation. Unlike our committee of neural network classifier, the models of classifiers of other recognition systems are more complex and inflexible. As the result of that those classifiers of the other systems cannot be easily distributed in standard computer networking at reasonable costs.

It is clear that all improvements of our handwritten numeral recognition system provide the industry a choice of the highest performance recognition system which can easily be maintained and modified, while the simplicities of the system model

allow our recognition system to be utilized in the standard networking computer at low cost.

6.8 Summary

In this chapter we present the handwritten numeral recognition system that performs the unprecedented recognition rate from our realistic image database. The recognition rates which the system achieved are 98.7% of maximum recognition and 98.1% of average recognition.

Further we also emphasize that our handwritten numeral recognition system has more simplicities and flexibilities which allow the system to be maintained, modified, and distributed on the standard networking computers for the faster processing.

Chapter 7
Automatic Aircraft Identification System

An automatic aircraft identification system is another image recognition application which has been very interesting to many researchers in both industry and military[3, 8, 18]. In general, the application systems preprocess and extract features from the binary bitmap silhouettes of aircrafts, then the systems use the features for identifying what type of aircraft the image belongs to. The challenges of this applications come from the extreme degree of rotation variance, scaling variance, and distortion caused by the digitizing image data. Despite the above difficulties, we can now demonstrate that only deviating angular feature, the feature extraction technique invented particularly for this application and a committee of multilayer neural network classifier trained by Epoch Gradual Increase in Accuracy training algorithm can be utilized so well that our experimental system achieves a perfect recognition rate. In the followings we are presenting the details of the experimental configurations for all three operational stages of this image recognition system, test results, and the test analysis.

7.1 Preprocessing

The information of aircrafts used for training and testing is basically the aircraft silhouettes. Before the feature extraction process, each aircraft silhouettes must first be digitized into a 50x50 binary bitmap image(see figure 7.1). This preprocessing not only provides a suitable form of data to the feature extraction process but also compresses the raw image data which in turn can accelerate the next processing

operation. It should be noted that image centering and image normalizing are not the requirements for this application thanks to the unique qualities of the deviating angular feature.

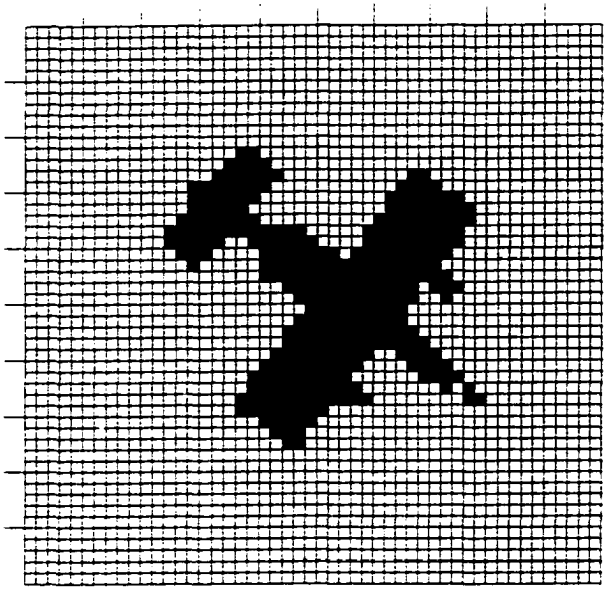


Figure 7.1: The 50x50 binary bitmap image of an aircraft

7.2 Feature Extraction

In this process only deviating angular feature from the silhouette curvature of a a 50x50 binary bitmap image will be generated. To solve the problem of variance in phase shifting, we compute the modulus of complex coefficients of Fourier series from the smooth deviating angular feature and we use the first 64 modulus of the complex coefficients in the classification process. As a result of these innovative approaches, we now have the image feature that is not only the compressed information of aircraft silhouette but also the feature that is absolutely translation, rotation, and scaling invariance.

7.3 Classification

Again we use a committee of multilayer feedforward neural networks for the final operational stage of the system. This classifier consists of 5 multilayer feedforward subneural networks(the number of neural neural networks must be equal to the number of aircraft types being classified), each of these neural networks is responsible for identifying a single type of aircrafts. The training algorithm for every neural networks is also Epoch Gradual Increase in Accuracy training algorithm we described in section 5.3. Similarly to the first experiment the winner-take-all decision policy is also used to get a final indication of what types of aircrafts the images represent(see figure 7.2).

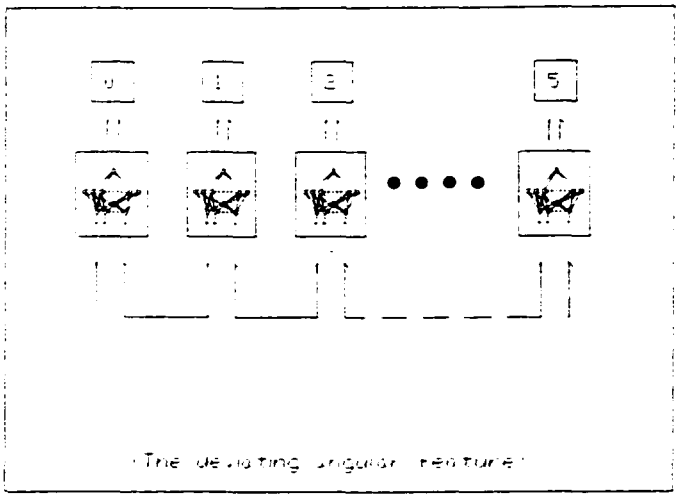


Figure 7.2: The committee of 5 multilayer neural networks

7.4 Database

We copied silhouettes of 5 different types of aircrafts from the aircraft models used in the experiment done by Dobnikar, Likar, Jurcic-Zlobec and Podbregar[8]. Then we manually digitized the aircraft models into 50x50 binary bitmap images, the total

number of images is 20(4 images in different rotation between 0 to $\pi/2$ for each aircraft models). The examples of five aircraft images are shown in figure 7.3.

7.5 Test Parameters

The followings are the test parameters and the configuration of neural network classifier used in this experiment.

Epoch Gradual Increase in Accuracy(EGIA)

Learning Rate = 0.7

Momentum Rate = 0.9

Range of Initial Weights = (-0.5, +0.5)

Initial Intermediate Tolerance = 0.5

decrement of Intermediate Tolerance = 0.03

The Structure of a Subneural Network

Input Layer = 64 nodes

Hidden Layer = 128 nodes

Output Layer = 1 nodes

7.6 Test Results

From the aircraft image database, we randomly allocate 10 images for training and the other 10 images for testing. After we run the experiments on five different test files, the experimental results show that the system can achieve 100% of recognition rate on every tests.

7.7 The Analysis

Obviously from the experimental results, we have perfected the automatic aircraft identification system. In comparison with the previous experimental systems similar

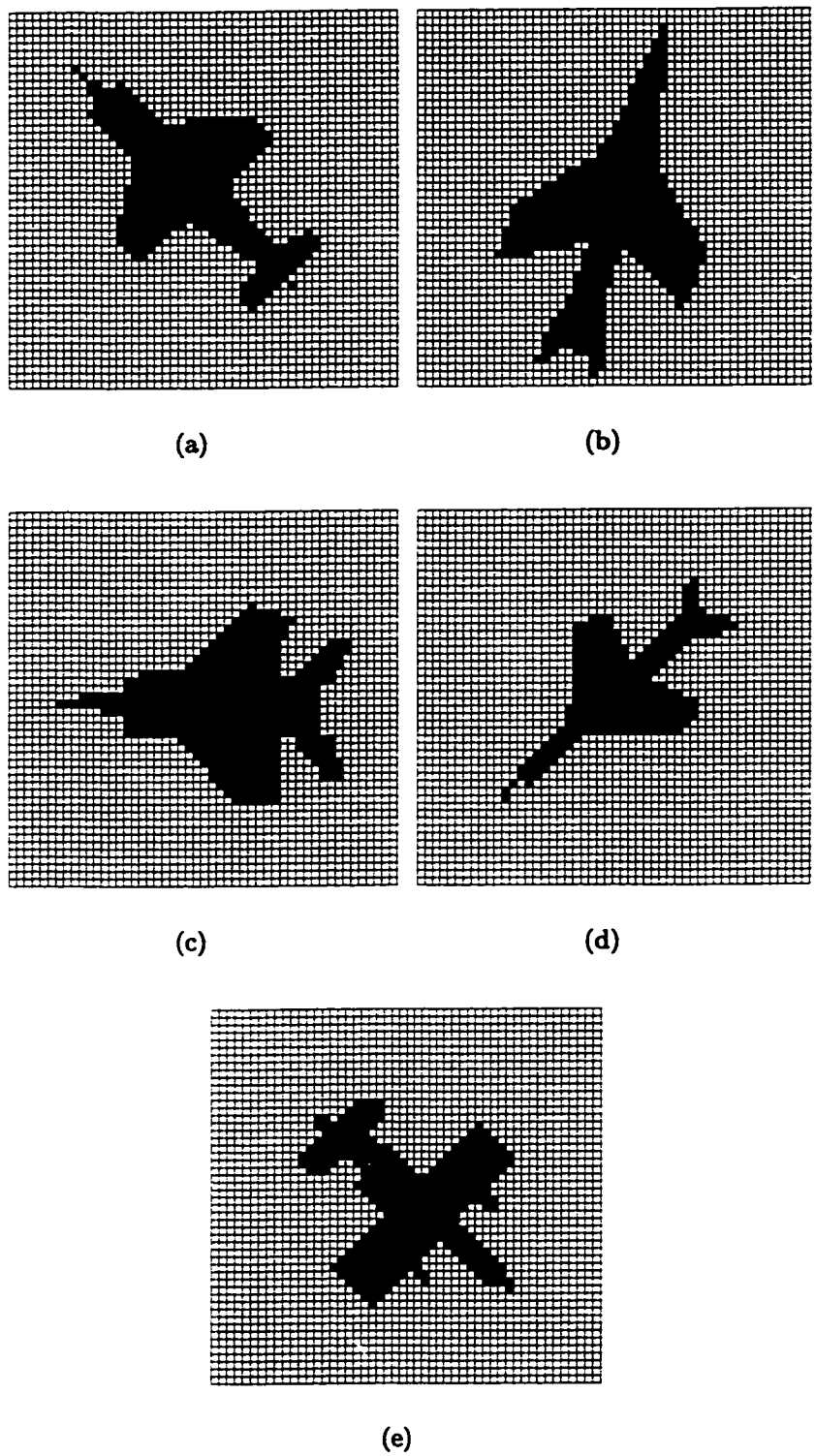


Figure 7.3: The 5 aircraft models used in our experiment

to ours, the performance of our recognition system has surpassed the classification neural network(the modified associative neural network) invented by Dobnikar, Likar, Jurcic-Zlobec, and Podbregar which achieved 99% of the highest recognition rate. It is important to be noted that the excellent achievement of our automatic aircraft identification system is the direct result of a perfect match between the image characteristics and the feature representation. With the powerful neural network classifier, the system prototype performs perfectly as we expected.

7.8 Summary

In this chapter we introduce the automatic aircraft identification system which utilizes only deviating angular feature and a committee of multilayer feedforward neural networks. The experimental results show that our prototype system performs perfectly in all tests of recognitions.

Chapter 8
Conclusion

The ability to recognize images makes it possible to abstractly conceptualize the world. This dissertation has presented a method of modifications to existent image recognition systems, which greatly improved the efficiency of data imaging methods. This recognition system has obvious applications in the field of handwriting recognition, as well as in the automatic identification of aircraft. The research has focused on the extraction of features and the training algorithm for neural network classifiers.

Our new approach for feature extraction process of handwritten numeral recognition system is to use the combination of the Kirsch feature and deviating angular feature to represent the numeral images. The design of our feature extraction process consists of four operational stages: 1) image profile extraction; 2) placing a finite marking points on the image profiles; 3) calculation of deviating angular features; and 4) calculation of Kirsch features. The deviating angular feature is a new feature representation which provides information about the sequence of angular changes on the image curvatures. We also prove that deviating angular feature is a translation, scaling, and rotation invariance. With the algorithms for image curvature extraction originally introduced in this research, Kirsch feature and deviating angular feature deliver the highly valuable information about the images to the next process. At the last operational stage in which the numeral image recognition takes place, our innovative approach is to use the parallel structure of neural network classifiers called a committee of multilayer feedforward neural networks for classification. In addition to

the new structural design, we invented the training algorithm named Epoch Gradual Increase in Accuracy(EGIA). EGIA is the evolution of the Gradual Increase in Accuracy training algorithm(GIA). The improvement of EGIA comes from the different policy in selection of train patterns. Unlike GIA which uses only one train pattern at every epochs, the EGIA uses a group of train patterns for training at particular intermediate tolerance. As a result of this different policy, the EGIA can train the neural network faster and more reliably than the GIA and the Backpropagation with momentum weight change training algorithm(BPM). According to the experimental results the EGIA can complete the training up to 37.27 times faster than the BPM and 3.30 times faster than the GIA for the classification problem of handwritten numeral images. The EGIA can also solve the incomplete learning problem which sometimes occurs in the neural networks trained by the BPM.

Utilizing new techniques on the two crucial operational stages, our handwritten numeral recognition system achieved the unprecedented rate of 98.7% maximum recognition and 98.1% average recognition.

Similar improvements also occur in the second application, the automatic aircraft identification system. First, we invented the customized feature extraction process for aircraft silhouettes which consists of three operational stages: 1) the silhouette curvature extraction; 2) calculation of smooth deviating angular feature; and 3) the calculation of the modulus of complex coefficients in Fourier series for the smooth deviating angular feature. This feature extraction process can provide the valuable feature which is completely invariant in translation, scaling, and rotation to the system classifier. Again we use a committee of multilayer feedforward neural networks

for the powerful classification, and the experimental results show that our prototype system performs perfectly in all tests of recognitions.

Finally we have come to the conclusion that our researches is successful in reaching the goal of improvements for the handwritten numeral recognition system and the automatic aircraft identification system. The prototype system delivers the highest recognition rates; meanwhile the designs of this system make it very easy for maintain, modify, and distribute on standard networking computers for fast processing.

Bibliography

[1] Alfred V. Aho, John E. Hopcroft, and *The Design and Analysis of Computer Algorithms.* Addison-Wesley Publishing Company, Reading, Massachusettes, 1974.

[2] Sven Behnke, Marcus Pfister, and Raul Rojas. Recognition of handwritten digits using structural information. In *International Conference on Neural Networks 3*, June 1997.

[3] Anoop K. Bhattacharjya and Badrinath Roysam. Joint Solution of Low, Intermediate, and High-Level Vision Tasks by Evolutionary Optimization:Application to Computer Vision at Low SNR. In *IEEE Transactions on Neural Networks 5, no.1*, January 1994.

[4] Christopher. M. bishop. *Neural Networks for Pattern Recognition.* Clarendon Press, Oxford, 1995.

[5] J. G. Carbonell. *Machine Learning Paradigms and Methods.* The MIT Press, Cambridge, 1990.

[6] Sung-Bae Cho. Neural-network classifiers for recognizing totally unconstrainted handwritten numerals. In *IEEE Transactions on Neural Networks 8, no.1*, January 1997.

[7] Cihan H. Dagli. *Artificial Neural Networks for Intelligent Manufacturing.* Chapman & Hall, London, 1994.

[8] A. Dobnikar, A. Likar, B. Jurcic-Zlobec, and D. Podbregar. Tracking and classifying 3d objects from tv pictures-a neural network approach. In *IEEE International Joint Conference on Neural Networks 3*, 1991.

[9] Kunikiko Fukushima and Nobuaki Wake. Handwritten alphanumeric character recognition by the neocognitron. In *IEEE Transactions on Neural Networks 2, no.3*, May 1991.

[10] Paul D. Gader, James M. Keller, Raghu Krishnapuram, Jung-Hsien Chiang, and Magdi Mohamed. Neural and fuzzy methods in handwritting recognition. In *Computer*, February 1997.

[11] Curtis F. Gerald and Patrick O. Wheatley. *Numerical Analysis.* Applied Addison-Wesley Publishing Company, Reading, fifth edition, 1994.

[12] Kazutoshi Gouhara, Tatsumi Watanabe, and Yoshiki Uchikawa. Learning process of recurrent neural networks. In *IEEE International Joint Conference on Neural Networks*, 1991.

[13] Mohamad H. Hassoun. *Fundamental of Artificial Neural Networks.* The MIT Press, Cambridge, 1995.

[14] Robert Hecht-Nielsen. *Neurocomputing.* Addison-Wesley Publishing Company, Reading, 1991.

[15] Yizhak Idan and Raymond C. Chevallier. Handwritten digits recognition by a supervised kohonen-like learning algorithm. In *IEEE International Joint Conference on Neural Networks 3*, 1991.

[16] Robert J. Jannarone. *Concurrent Learning and Information Processing, A Neuro-Computing System that Learns During Monitoring, Forecasting, and Control.* Chapman & Hall, New York, 1997.

[17] Satoshi. Kageyu, Noburu Ohnishi, and Noboru Sugie. Augmented multi-layer perceptron for rotation-and-scale invarient hand-written numerall recognition. In *IEEE International Joint Conference on Neural Networks 1*, 1991.

[18] Dae-Young Kim, Sung-II Chien, and Hyun Son. Multiclass 3-d aircraft identification and orientation estimation using multilayer feedforward neural network. In *IEEE International Joint Conference on Neural Networks 1*, 1991.

[19] Stefan Knerr, Leon Personnaz, and Gerard Dreyfus. Handwritten digit recognition by neural networks with single-layer training. In *IEEE Transactions on Neural Networks 3, No.6*, November 1992.

[20] Ravi Kothari, Powsiri Klinkhachorn, and Roy S. Nutter. An accelerated back propagation training algorithm. In *IEEE International Joint Conference on Neural Networks 1*, 1991.

[21] Raghu Krishnapuram and Ling-Fan Chen. Implementation of parallel thining algorithms using recurrent neural networks. In *IEEE Transactions on Neural Networks 4, No.1*, January 1993.

[22] S. Y. Kung. *Digital Neural Networks.* PTR Prentice Hall, Englewood Cliff, 1993.

[23] Sukhan Lee and Jack C. Pan. Handwritten numeral recognition based on hierarchically self-organizing learning networks. In *IEEE International Joint Conference on Neural Networks 2*, 1991.

[24] Sukhan Lee and Jack Chien-Jan Pan. Unconstrainted handwritten numeral recognition based on radial bases competitive and cooperative networks with spatio-temporal feature representation. In *IEEE Transaction on Neural Networks 7, no.2*, March 1996.

[25] Cornelius. T. Leondes. *Image Processing and Pattern Recognition.* Academic Press, San Diego, 1998.

[26] James L. McClelland, David E. Rumelhart, and the PDP Research Group. *Parallel Distributed Processing, Explorations in the Microstructure of Cognition*, volume second. The MIT Press, 1986.

[27] Larry R. Medsker. *Hybrid Intelligent Systems.* Kluwer Academic Publishers, Boston, 1995.

[28] Kishan Mehrotra, Chilukuri K. Mohan, and Sanjay Ranka. *Elements of Artificial Neural Networks.* The MIT Press, Cambridge, 1997.

[29] Stavros J. Perantonis and Paulo J. G. Lisboa. Translation, rotation, and scale invariant pattern recognition by high-order neural networks nd moment classifiers. In *IEEE Transactions on Neural Networks 3, no.2*, January 1992.

[30] Murray R. Spiegel. *Mathematical Handbook of Formulas and Tables.* McGraw-Hill Book Company, New York, 1968.

[31] Sargur N. Srihari. *Computer Text Recognition and Error Correction.* IEEE Computer Society Press, Silver Spring, 1985.

[32] Richard S. Sutton. Two problems with backpropagation and other steepest-descent learning procedures for networks. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, 1989.

[33] Kun Won Tang. Cc4: A new corner classification approach to neural network training. Master's thesis, Louisiana State University, 1997. Thesis(M.S. in E.E.)–Louisiana State University, Baton Rouge, 1997.

[34] Charles C. Tappert. Cursive script recognition by elastic matching. In *IBM Journal of Research and Development*, volume 26, November 1982.

[35] I. M. Vinogradov. *Encyclopaedia of Mathematics*, volume 26. Kluwer Academic Publishers, Boston, 1988.

[36] Li-Xin Wang. *A Course in Fuzzy Systems and Control.* Prentice Hall PTR, Upper Saddle River, 1997.

[37] Paul John Werbos. *Beyond Regression: New Tools for Prediction and Analysis the Behavioral Sciences.* PhD thesis, Harvard University, Cambridge, Massachusetts, August 1974.

[38] Paul John Werbos. *The Roots of Backpropagation, From Ordered Derivatives to Neural Networks and Political Forecasting.* John Wiley & Sons, Inc., New York, 1994.

[39] Qiangfu Zhao and Tatsuo Higuchi. Evolutionary learning of nearest-neighbor mlp. In *IEEE Transactions on Neural Networks 7, no. 3*, May 1996.

Vita

Adisak Srinakarin received his bachelor of engineering degree from Chulalongkorn University, Bangkok, Thailand in March 1983. After graduating, he worked as a field engineer at a telephone company in Thailand.

In the summer of 1986, he enrolled in the department of Computer Science at Southern University, Baton Rouge. He obtained his master of science degree in 1988. He worked as a programmer analyst at a company in Dallas, Texas for almost four years.

He joined the Computer Science department at Louisiana State University for a doctoral degree in January 1994.

His research interests include pattern recognition, training algorithm for neural networks, computer networks, and real-time computer programming.

DOCTORAL EXAMINATION AND DISSERTATION REPORT

Candidate: Adisak Srinakarin

Major Field: Computer Science

Title of Dissertation: Deviating Angular Feature for Image Recognition Using the Improved Neural Network Classifier

Approved:

Major Professor and Chairman

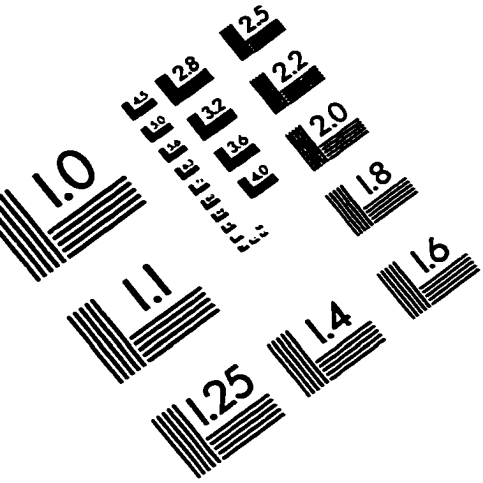
Dean of the Graduate School

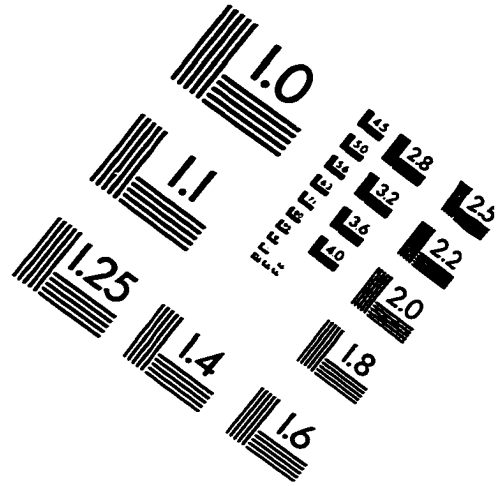
EXAMINING COMMITTEE:

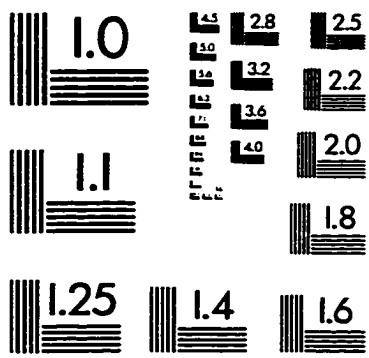
Date of Examination:

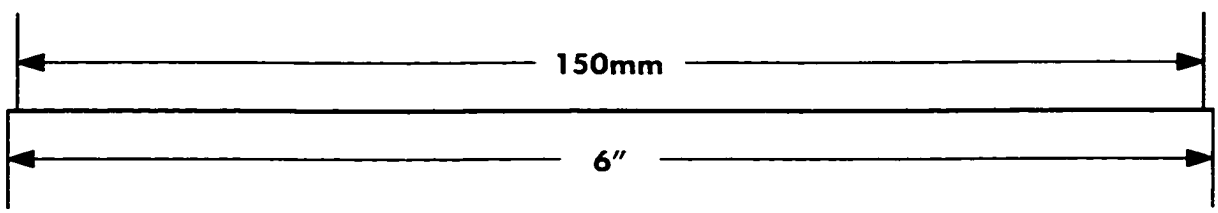
October 13, 1998

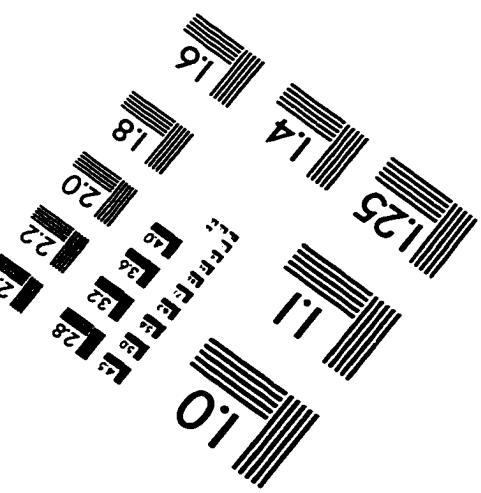
IMAGE EVALUATION
TEST TARGET (QA-3)

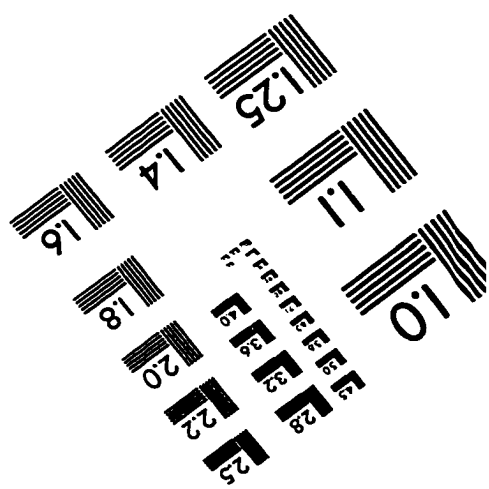












APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989