

AD-A256 468



Feature Extraction and Classification of FLIR Imagery
Using Relative Locations of Non-Homogeneous Regions
with Feedforward Neural Networks

DISSERTATION

Kevin L. Priddy
Captain, USAF

AFIT/DS/ENG/92-01

DTIC
ELECTE
OCT 27 1992
S E D

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

DISTRIBUTION STATEMENT A
Approved for public release
Distribution Unlimited

①

AFIT/DS/ENG/92-01

**Feature Extraction and Classification of FLIR Imagery
Using Relative Locations of Non-Homogeneous Regions
with Feedforward Neural Networks**

DISSERTATION

**Kevin L. Priddy
Captain, USAF**

AFIT/DS/ENG/92-01

Approved for public release; distribution unlimited

92-28251



**Feature Extraction and Classification of FLIR Imagery
Using Relative Locations of Non-Homogeneous Regions
with Feedforward Neural Networks**

DISSERTATION

**Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy**

**Kevin L. Priddy, B.S.E.E., M.S.E.E.
Captain, USAF**

7 April 1992

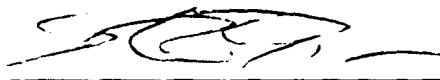
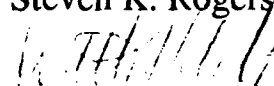
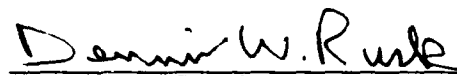

Accession For	
NTIS - CRA&I	<input checked="" type="checkbox"/>
ERIC - EBS	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Special Agent	
By _____	
Date _____	
7 April 1992	
Date	Approved
A-1	

AFIT/DS/ENG/92-01

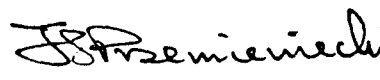
Feature Extraction and Classification of FI,IR Imagery
Using Relative Locations of Non-Homogeneous Regions
with Feedforward Neural Networks

Kevin L. Priddy, B.S.E.E., M.S.E.E.
Captain, USAF

Approved:

 _____ Steven K. Rogers, Chairman	<u>7 Apr 92</u>
 _____ Matthew Kabrisky	<u>7 Apr 92</u>
 _____ Dennis W. Ruck	<u>7 Apr 92</u>
 _____ John Jones Jr.	<u>7 April 92</u>

Accepted:

 10 April 1992

J. S. Przemieniecki
Senior Dean

Table of Contents

	Page
Table of Contents	iii
List of Figures	vi
List of Tables	ix
Abstract	x
Preface	xi
I. Introduction	1
1.1 Problem Statement	1
1.2 Motivation	2
1.3 Target Identification	5
1.3.1 Segmentation Using Gabor Functions	7
1.3.2 Feature Extraction	10
1.3.3 Classification Using Neural Networks	11
1.4 Significant Results	14
1.5 Dissertation Overview	16
II. Using Gabor Filter Methods to Learn The Gestalt	17
2.1 Background	17
2.2 The Lazofson Pattern Recognition Architecture	20
2.3 The Gestalt Network System Architecture	24
2.3.1 Gabor Processing	26
2.3.2 The Neural Network Processor	30

	Page
2.3.3 The Mean Squared Error Classifier	30
2.4 Understanding the Gestalt Representation	32
2.5 Testing the Notion of Learning the Gestalt	37
III. Alternative Classification Schemes Using Distance Metrics	41
3.1 Choosing the top N Gabor Coefficients and Their Locations	41
3.2 Delta Method	44
3.3 Euclidean Distance Metric	46
3.4 Centroid Metrics	47
3.4.1 Centroid Euclidean	48
3.4.2 Centroid Taxi Distance	49
3.4.3 Centroid Delta	50
3.5 Relative Locations of Spatial Textures	51
IV. Searching the Input Space for Optimal Features	57
4.1 The Curse of Dimensionality	57
4.2 Saliency of Features	57
4.3 Comparison With A Previous Saliency Metric	64
4.4 Application of the Saliency Metric For Feature Selection	65
V. Relative Locations and Their Importance in Object Recognition	72
5.1 The Concept of Relative Locations	73
5.2 The Relative Location System Architecture	74
5.3 Template Creation	76
5.4 Template Registration	79
5.5 Feature Vector Creation	80
5.6 Neural Network Training	84
5.7 Input Image Classification	87
5.8 System Performance	89

	Page
5.9 Discussion of Results	89
5.10 Why Relative Locations Are Better Than Matched Filter Techniques	93
5.11 Limitations of the Relative Location Concept	94
5.12 Conclusions	96
VI. Conclusions and Recommendations	98
6.1 Conclusions	98
6.2 Contributions	99
6.3 Recommendations	100
Bibliography	102
Vita	106

List of Figures

Figure		Page
1.	Block Diagram of General Architecture Used for Target Classification . .	3
2.	Neural Network Based Automatic Target Recognizer Using Roving Window	5
3.	Neural Network Based Automatic Target Recognizer Using Relative Locations.	6
4.	The Pattern Recognition Process	7
5.	Two-Dimensional Cosine Gabor Function	8
6.	Transformation of Cosine Gabor Function.	9
7.	Diagram of Perceptron Node with A Sigmoid Activation Function	13
8.	Diagram of Typical Backpropagation Net With A Single Hidden Layer .	14
9.	Architecture for the Solution of Handwritten Characters Developed by Le Cun	19
10.	Neocognitron Architecture for the Solution of Handwritten Characters Developed by Fukushima	20
11.	Biologically Inspired Pattern Recognition Architecture Used by Lazofson	21
12.	Image Classification Results For Lazofson Architecture	23
13.	Neural Network Based Automatic Target Recognizer	24
14.	Two-Dimensional Cosine Gabor Function for Various Rotation Angles and Spatial Frequencies	25
15.	Simplified Diagram of Gabor Processor with Two-Dimensional Arrays . .	27
16.	Diagram of Perceptron Node	29
17.	Diagram of Single Neural Network Processor	31
18.	Filters With Fixed (ρ) Variable (ϕ) in (f_x, f_y) Coordinate System	33
19.	Filters With Variable (ρ, ϕ) in (f_x, f_y) Coordinate System	34
20.	Simplified Diagram of Gabor Processor with Linear Arrays	35
21.	Three Dimensional View of Correlation Space	36

Figure	Page
22. Class Problem for Network Training	38
23. Mean Squared Error as a Function of Training Iteration For Each of The Classes in The Four Class Problem.	38
24. FLIR Image of Tank and Truck in Clutter	40
25. Truck With Associated Gabor Filter Images (16x16) Window.	43
26. Three Class Problem Overlaid With Top Ten Gabor Locations.	44
27. Comparison of Training Accuracy and Test Set Accuracy For Delta Rule .	45
28. Comparison of Training Accuracy and Test Set Accuracy For Euclidean Metric	46
29. Shift of Gabor Maxima For Two Similar Images	47
30. Comparison of Training Accuracy and Test Set Accuracy For Centroid Euclidean Metric	49
31. Comparison of Training Accuracy and Test Set Accuracy For Centroid Taxi Metric	50
32. Extraction of Centroid Delta Position Data	51
33. Comparison of Training Accuracy and Test Set Accuracy For Centroid Delta Metric	52
34. Result of Correlating a Truck With a Tire	53
35. Result of Correlating a Tank With a Cog	53
36. Comparison of Training Accuracy and Test Set Accuracy For Relative Lo- cation Data Set	54
37. Output Images obtained from Correlating Gabor Templates with Original Image.	56
38. Three-Layer Feedforward Network	60
39. Three Class Problem For Classification	66
40. Three Class Problem Overlaid With Top Ten Gabor Locations	66
41. Accuracy Using All Nine Features from Centroid Euclidean Distance Metric	68
42. Accuracy Using Top Three Features from Centroid Euclidean Distance Metric	71

Figure	Page
43. Eye Scanning Pattern For Girl Used by Luria (30:135)	73
44. Neural Network Based Automatic Target Recognizer Using Relative Locations	75
45. The Relative Location Screen Running Within NeuralGraphics	77
46. Creation of a Template	78
47. Using the Correlation of the Template and the Input Image for Registration	80
48. Two Separate Representations of The Same Target With Diurnal Differences	81
49. Typical Output From a Training Run for The Neural Network	86
50. Block Diagram of Neural Network Training	87
51. Block Diagram of Classifier	88
52. Comparison of Two Separate Templates And Their Separate Neural Network Outputs For The Same Input Image	95

List of Tables

Table		Page
1.	Figures of Merit for First Four Class Problem	39
2.	Figures of Merit for Second Four Class Problem	39
3.	Results For Tactical Target Problem	39
4.	Importance of Each Centroid Euclidean Feature Over 100 Runs	70
5.	Saliency of Each Centroid Euclidean Feature over 100 runs	70
6.	Classification Results Using Full Templates As Matched Filters	90
7.	Classification Results Using Cropped Templates As Matched Filters	91
8.	Classification Results Using Relative Location Templates As Matched Filters	91
9.	Classification Results Using Relative Locations	92
10.	Classification Results Using Relative Locations For Both Reverse and Normal Input Images	92

Abstract

The classification of forward looking infrared (FLIR) imagery is explored using Gabor transform decomposition and relative locations of non-homogeneous regions combined with feedforward neural networks. A feature saliency metric is developed from a Bayesian sensitivity analysis of feedforward neural networks. This metric is then used to reduce the dimensionality of the feature vectors used to identify FLIR imagery without any degradation of classification accuracy. Several system architectures are developed using a roving window combined with a series of Gabor filters to produce feature vectors for presentation to a neural network classifier. One architecture uses the Gabor filter coefficients to learn the gestalt of known images. Several of the gestalt networks are then combined to determine the class of an unknown image. Another architecture uses centroid metrics for the cluster of Gabor resonances to feed a backpropagation network acting as a traditional Bayesian classifier. A system architecture is developed which uses the relative locations of texture regions within a user defined template to classify imagery. The relative location architecture is shown to outperform traditional matched filter classifiers. The relative location architecture is shown to be robust in solving the problems presented by crepuscular lighting conditions.

Preface

This dissertation research is based upon the premise: while neural networks may have no relationship at all to how the brain really works, they are useful in solving many problems. Someday, in the future, someone will have an explanation for the bigger picture. Indeed, the Wright brothers showed that feathers had very little to do with flying, but they are still very useful for stuffing my pillow to give my head a soft place to lay down at night. Neural networks may not be a very good model of how the brain works, but they make us feel as though we have control over some of the mysteries of life. They are useful tools which, when applied properly, will aid in many applications.

I have many thoughts as I conclude this research and have finished writing the dissertation. I often think of my father who has been dead now for almost twenty years and how we would be able to discuss the latest research in the field of electrical engineering. He was the light that I have always looked to in the times which have been difficult or frustrating. I owe all that I am and all that I have learned to my father and my mother who instilled in me a zest for life and taught me to work hard at anything I attempted. I am forever grateful for my father's interest in helping me to understand electronics. My wife is certainly my chief supporter and has been a true companion to me as I have struggled to gain an education. My children have been pretty neglected the last few years and I look forward to spending more time with them in the future. To Wendy and the children I give my greatest thanks. You are what life is all about.

I also wish to thank my good friends Dr Greg Tarr, Dr Dennis Ruck, and Capt Tom Burns who have read my manuscripts and offered good advice and encouragement. In addition, I wish to thank my advisor Dr Steve Rogers (aka Captain Amerika) who has been patient when I was frustrated and when I was overly optimistic. He has always been there to ask the right question or help when discussing the latest architecture to be implemented. Dr Matt Kabisky has always been a big help especially with the "retrospectroscope". To

all of you I can only say thanks in three languages so here goes: "thanks", "velen danke", and "merci beaucoup".

Kevin L. Priddy

Feature Extraction and Classification of FLIR Imagery
Using Relative Locations of Non-Homogeneous Regions
with Feedforward Neural Networks

I. Introduction

1.1 Problem Statement

The problem addressed in this dissertation is that of recognizing objects in forward looking infrared (FLIR) imagery. Several approaches to solving this problem are reported along with their success in classifying objects. In addition, a Bayesian saliency measure (36) is introduced to aid in determining which features introduced to a feedforward, backpropagation trained, neural network (29, 46, 58) are useful in classifying objects. A database of FLIR imagery was used to develop and test the techniques described in this dissertation.

The primary goal of this research was to develop and test a recognition system based upon a combination of preprocessing techniques and neural network architectures which allow the recognition system to perform feature extraction and classification in a single step. An additional goal was to test the approach against matched filter approaches used in target classification today.

Figure 1 is a block diagram of the general system architecture used in the dissertation. The object to be classified is presented to the system. Several features (measurements) are extracted which describe the various spatial properties of the object. The features are processed with the results stored in a feature vector. The feature vector is then presented to several class specific neural network classifiers. Each neural network makes an independent assessment of the class of the target and the results are then sent to a

comparator for analysis. In the comparator the outputs of each of the neural networks are compared and the class of the network with the best match via error measurement or the largest in-class output is chosen as the class of the object. During the course of this dissertation several different preprocessing techniques will be discussed as well as two separate neural network classification algorithms. The application of this general architecture resulted in several contributions to the body of scientific knowledge in the neural network and pattern recognition fields.

The contributions of this research are: 1) the development of a processing architecture which is capable of determining the gestalt of a target from the spatial information available from Gabor filter techniques, 2) a saliency method for determining the optimum features for any feedforward neural networks based upon a Bayesian sensitivity analysis, 3) a neural network architecture which allows for the addition of new class types without retraining, and 4) the use of relative locations of spatial features in object recognition. The philosophical, theoretical and practical aspects of these contributions are discussed in the following chapters.

The rest of this chapter is divided into four sections. Motivating factors for conducting the research are discussed in the next section. This is followed by background material which is pertinent for understanding pattern recognition, Gabor transforms and backpropagation neural networks. Next, the significant results are summarized. The chapter concludes with an overview of the organization of the dissertation.

1.2 Motivation

The problem of target identification from FLIR, ladar or radar information has been of keen interest to the Air Force for several decades. The typical tracker used in our missiles today is nothing more than a hot spot tracker. Even with human intervention, the missile is dependent upon a laser spot to home into the target. The ideal situation for the combat operator would be to either have the capability to program the missile beforehand or the capability to highlight a certain type of target and then have the missile search until it

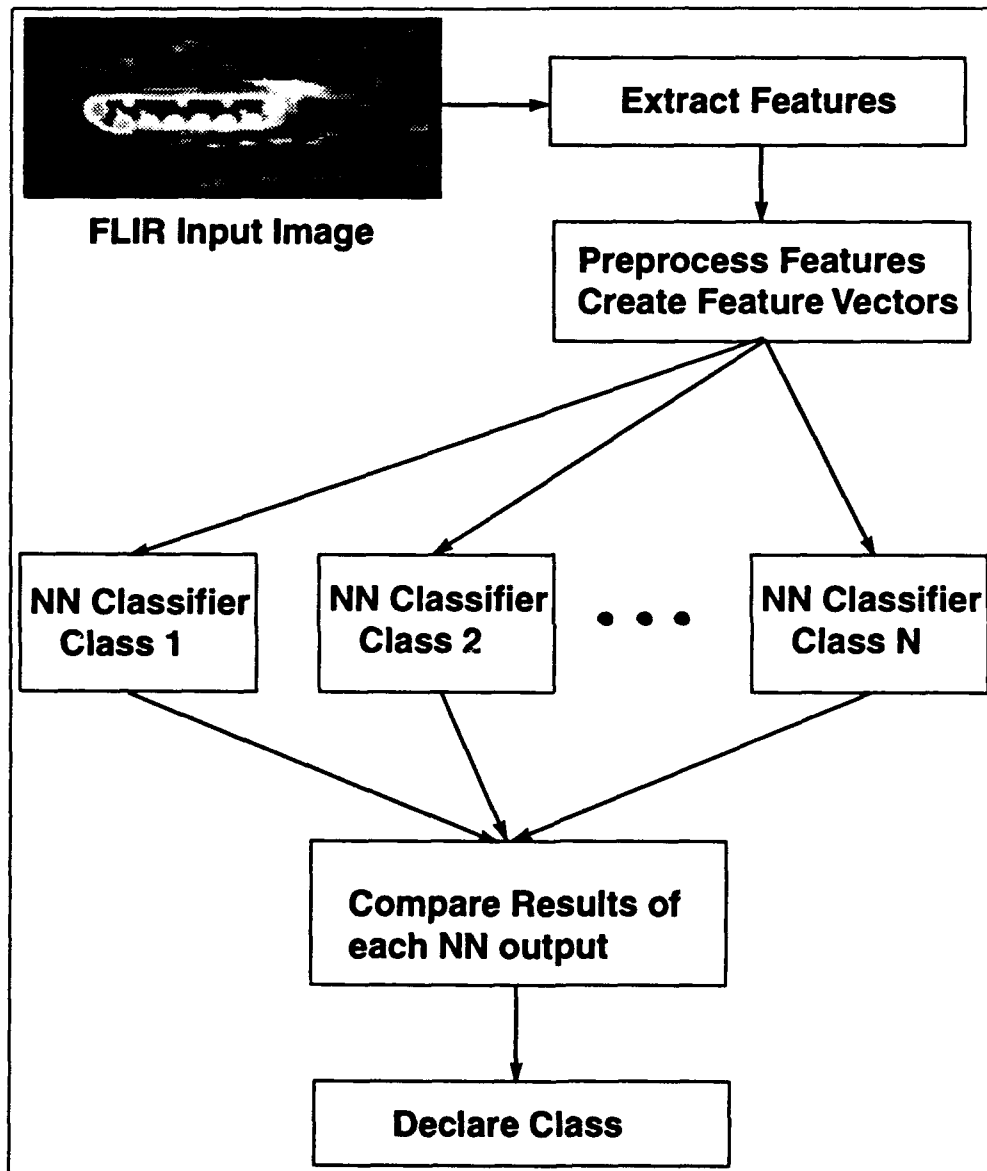


Figure 1. Block Diagram of General Architecture Used for Target Classification

finds the desired target. Currently this type of target matching is performed in laboratories using various correlation techniques (10, 17, 19, 20, 59). The correlator is one of the best target identification techniques because it precisely matches the target. But therein lies the rub: the correlator must have many templates to match all of the aspect angles, scale changes and rotations of objects. The goal of correlation based research is then to find an optimal match between accuracy/flexibility and storage. An alternative approach has been to use invariant features such as Zernike moments (41, 42) as inputs to neural networks or Bayesian (37, 38) statistical approaches to distinguish between targets and non-targets. While many of these approaches have been successful they all suffer from an inordinate amount of storage for matched filtering or preprocessing in order to create the invariant features.

Because of the processing bottleneck and storage limitation problem, the initial phase of the research was directed at determining a reduced representation for each class of target in the classification task. The research effort explored the use of Gabor filtering (2, 7, 13) as a means to reduce the required representation of each target. Initially an architecture was developed which used the inner product between scene data contained in a roving window and a bank of Gabor filters (fixed weights) to train a separate neural network to learn the gestalt representation of each target as depicted in Figure 2. Hence, instead of storing many matched filters, the weights of the trained neural network store the gestalt or reduced representation of the desired target. More traditional classification approaches were used with various distance metrics between selected Gabor spatial textures as features. In addition, the concept of using the relative locations of specific spatial textures was exploited for feature extraction and classification. The relative location system developed in this dissertation has the same architectural form (see Figure 3) as the Gabor approach but is significantly better at classifying objects with greatly reduced training times and far fewer features. Finally, because of the extensive use of artificial neural networks in this research, a method was developed to determine which features presented to a network were of importance in determining the proper class.

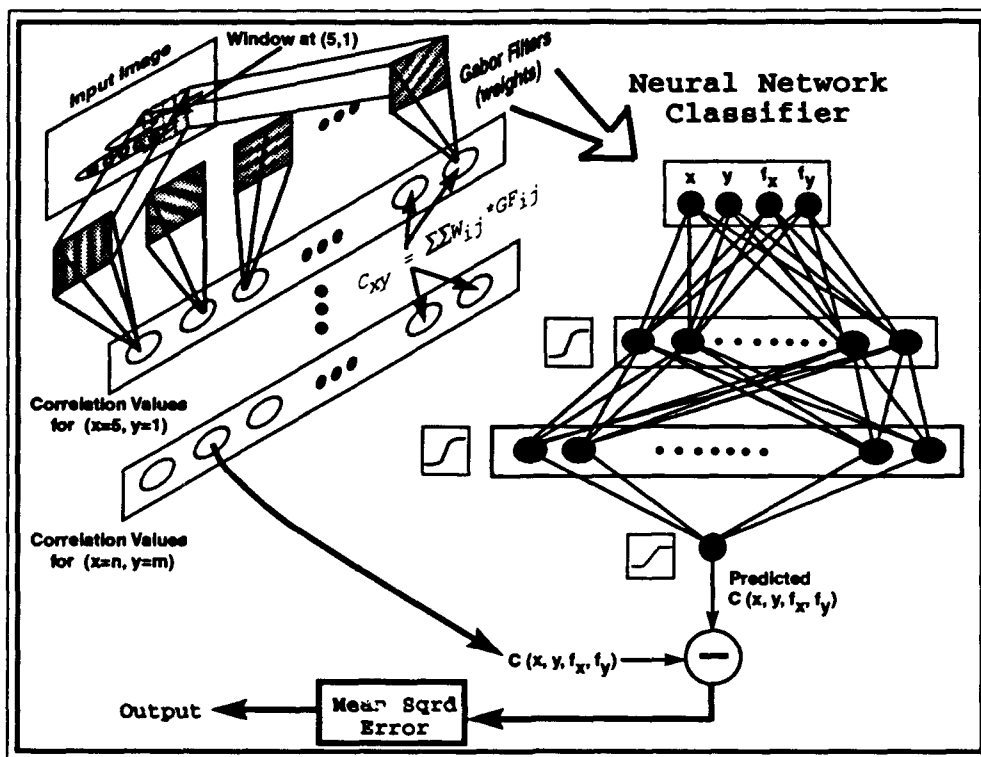


Figure 2. Neural Network Based Automatic Target Recognizer Using Roving Window

1.3 Target Identification

The entire process of picking out objects of interest from a scene and then classifying them by their features is complex. One of the big issues today is the inability of machines to segment and extract features from a scene of interest. The literature is replete with different techniques currently being tried by researchers (8, 40, 53) to solve the target identification problem.

Target Identification consists of three basic steps: 1) find areas of interest in the scene (segmentation), 2) make measurements on the areas of interest (feature extraction), and 3) based on the feature set, determine the appropriate classes for the areas of interest (classification). Figure 4 outlines the basic steps used in pattern recognition. Current techniques today might use edges, blobs, or spatial frequency for segmentation. These techniques use large amounts of computer processing before areas of interest are located in the input scene. The feature extraction process often involves calculation of moments,

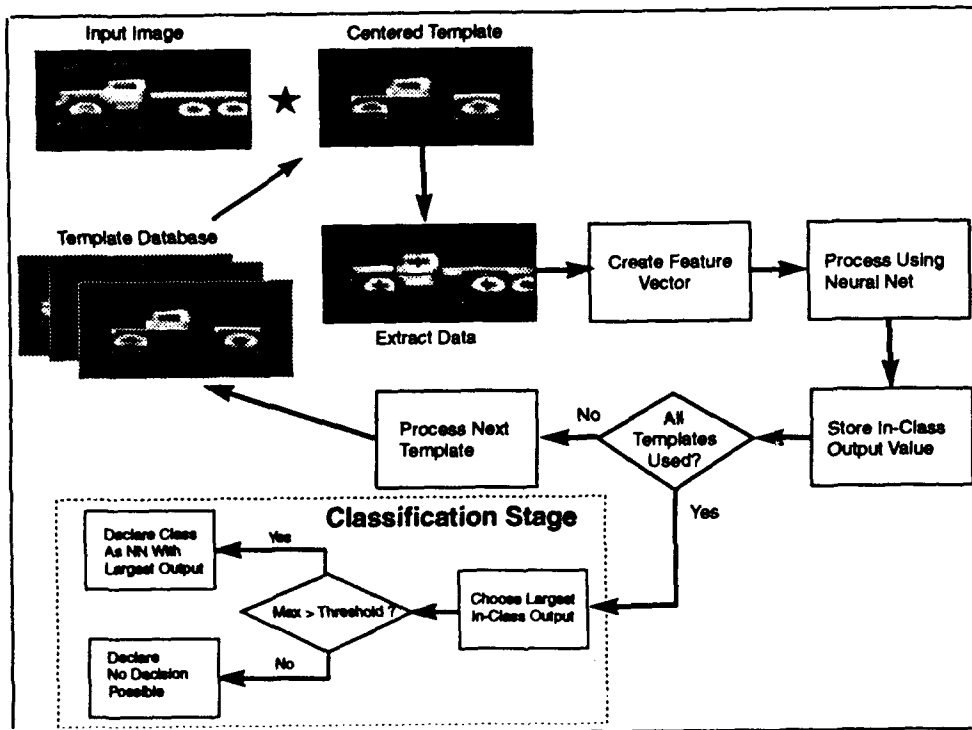


Figure 3. Neural Network Based Automatic Target Recognizer Using Relative Locations. The * indicates the correlation operator.

boundaries and pixel densities for blobs in the scene that also require large amounts of computer processing. The last step in the process, classification, usually involves large amounts of computer processing as well. To gain a better understanding of the pattern recognition process and how it is implemented in this dissertation research, each of the steps will be explained in terms of their implementation in the system architecture shown in Figure 1. The first step in pattern recognition is termed segmentation. Segmentation using Gabor functions is discussed in the next subsection.

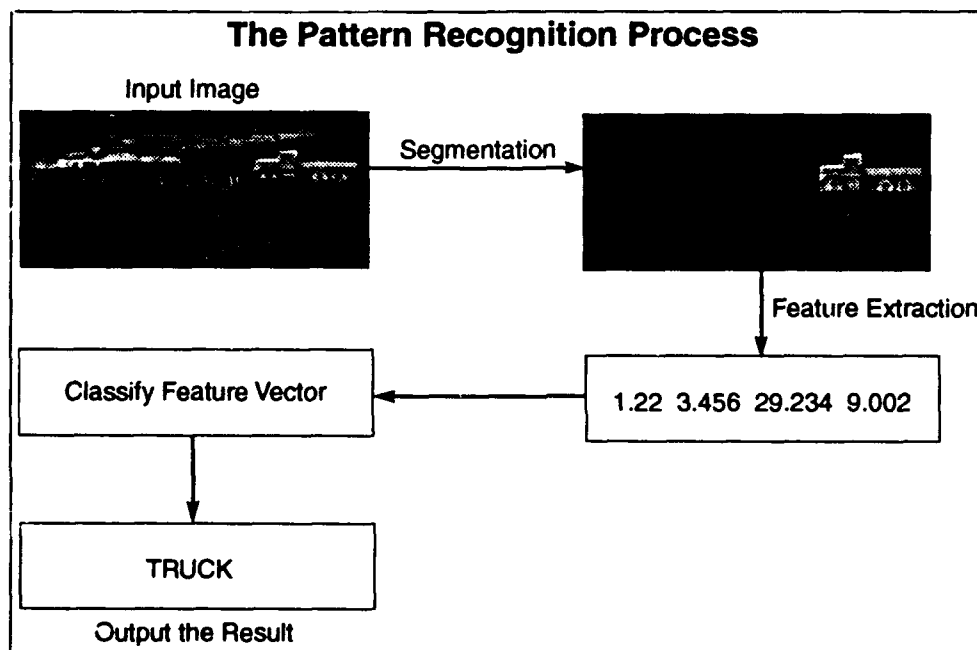


Figure 4. The Pattern Recognition Process

1.3.1 Segmentation Using Gabor Functions The process of segmentation involves choosing the items of interest from an image. There are many techniques to segment images but we will limit the discussion to those which show promise for implementation optically or via neural network algorithms. Hubel and Wiesel (21) discovered groups of neurons which preferentially respond to particular orientations of line segments in the field of view of the eye of a cat. Several authors (6, 55, 22) have shown that these "edge detectors" can be explained as behaving as Gabor filters (13) with particular Gaussian widths and spatial frequencies. Several authors (1, 54, 56) have segmented items of interest from image data

using a series of Gabor filters. Figure 5 is an example of the two-dimensional cosine Gabor function with its associated two-dimensional Fourier transform.

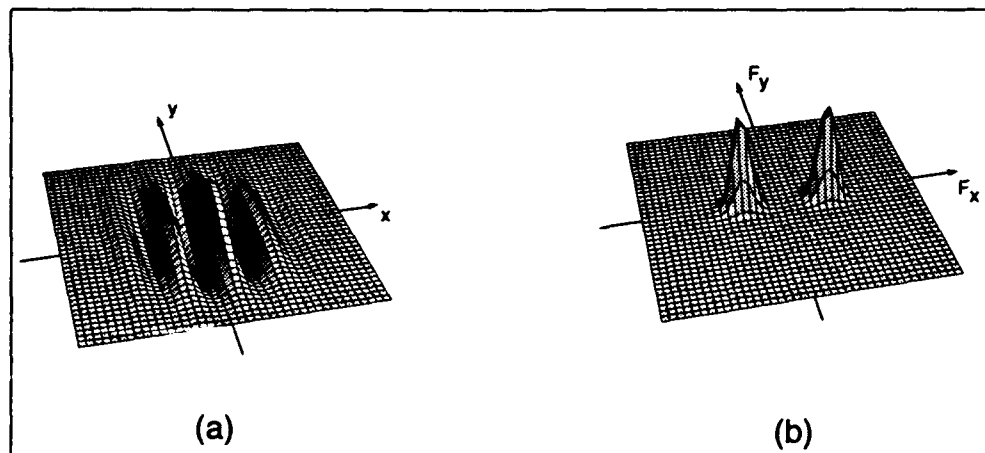


Figure 5. Two-Dimensional Cosine Gabor Function. a) 0 Degrees Rotation at Six Cycles Per Window, b) Fourier Transform of (a)

The Gabor function is represented in the Fourier domain as a pair of Gaussian profiles separated by twice the spatial frequency ($\pm f$) of the sine or cosine function. The radius of the Gaussian profiles is inversely proportional to the radius of the Gaussian window function used to window the sine/cosine function. Figure 6 is an example of a combination of two Gabor Filters, one at zero degrees rotation and the other at 90 degrees rotation and the resultant Fourier transform. In addition to the Gabor-like filtering properties of the brain, sets of pre-processing neurons in the retina perform a coordinate transformation as information is relayed from the retina to the mapping area in the striate cortex. This transformation has been shown to approximate a log-polar transformation (52). The log-polar transformation allows for changes in scale and in-plane rotation to be represented as shifts in the mapping plane. This log-polar mapping may also be useful in motion detection. Thus, the mapping area appears to be optimized for a correlator which is capable of handling shifts but not scale changes and rotations in images. Indeed, we may very well have a correlator in the inner mind which correlates the object of interest with a database of stored images. The human visual system performs a series of segmentations and transformations which makes it easier to extract useful features and

subsequently classify an object of interest. There are many other techniques available for segmenting objects from a scene. For instance, Tong (51) used sharp gradients to segment objects from ladar and FLIR images while Roggemann (37:14) used planar patches to help pick out objects from similar ladar data. As Duda and Hart (8:5) point out, many of the measurements which are taken for a particular problem depend a great deal on the problem. The reader may ask “Why are we looking at visual systems to gain knowledge about segmentation?” The simple answer, and the most convincing, is that we have not yet invented a segmenter that works nearly as well as the human visual system. This research investigated the use of Gabor functions to perform the segmentation task as well as the feature extraction and classification tasks.

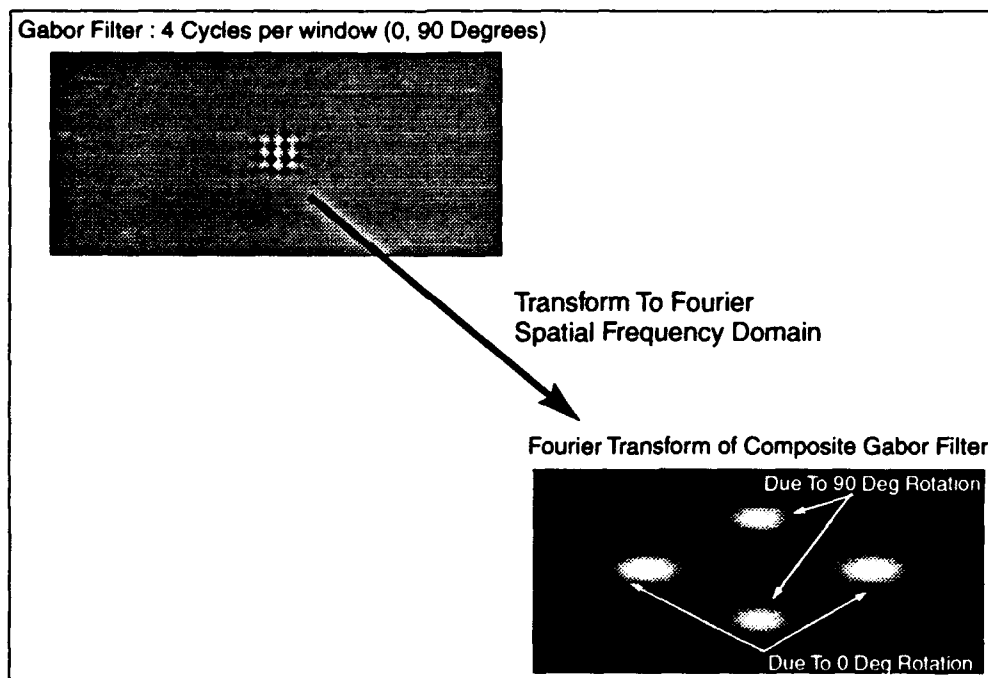


Figure 6. Transformation of Cosine Gabor Filter Between Normal Spatial Domain and the Fourier Spatial Frequency Domain

Once the segmentation has taken place the next step in the identification process is that of making meaningful measurements on the areas of interest which will allow the classifier to distinguish one type of object from another.

1.3.2 Feature Extraction The process of taking measurements on the object of interest is known as feature extraction. For instance, if we were interested in describing an electrical signal we would probably describe it in terms of its amplitude, phase, and frequency. These three attributes could be measured and used to represent the electrical signal. These measurements would be the features which, hopefully, uniquely describe the electrical signal. The brain also performs feature extraction on objects of interest. Kabrisky (23:82) has shown that the brain can perform a Fourier-based correlation. Ginsburg (14), a previous Kabrisky student, reports that many of the optical illusions we see can be explained using a low-order Fourier spatial frequency mapping. In addition, Oberndorf (35) has shown results similar to Ginsburg's work using Gabor filters. Because the brain extracts spatial information in the mapping area in a way which can be imitated using Fourier spectral analysis, the Fourier domain appears to be a place to start. In addition to Fourier spectral analysis, which has been used for decades to describe objects, other methods are also used for feature extraction. Some authors (9, 41, 50) have used invariant Zernike moments as features of an object. Other techniques for extracting features may involve line descriptors, clustering techniques, and topological measurements. Other authors (3, 18) have used Hough transform coefficients as features. We are not sure what is involved with picking out objects from a scene when the brain is involved. From Ginsburg we know that spatial features are extracted. We also know that color could be used as a discriminant, and that the scale and rotation of an object are virtually eliminated because of the approximate log-polar transformation. Perhaps the brain uses a correlator to perform much of the classification task while using the various features extracted from other methods to double check and enhance the choice of the classifier. Luria (30:134) asserts that the brain is constantly updating its perceptual model, while performing a "perceptual hypothesis", by incorporating features which support the hypothesis while rejecting those which are of little or no value. This leads us to the final task in a target identifier the classification of the target.

1.3.3 Classification Using Neural Networks Classification is the identification of a particular object based upon a set of known measurements or features. Once all of the preprocessing is finished, the results are sent to a classifier which makes the final decision based upon the feature vector presented as the input to the classifier. Several neural networks are available today for use as a classifier. Lippmann (29) has a good description of the classification properties of many of the neural network architectures available today. The classifier used in this dissertation is known as a feedforward backpropagation neural network. This neural network has the interesting property of being able to determine the underlying *a priori* probability distribution (15, 45) for each class being considered in the classification problem. This allows for direct comparison with traditional Bayesian classifiers. In addition, it will be shown in Chapter 4 in this dissertation that the saliency (usefulness) of features can be determined through exploiting the Bayesian properties of the backpropagation network. To gain a better understanding of the backpropagation neural network a short review is presented.

The backpropagation neural network is based upon the perceptron (39). The perceptron was envisioned by Rosenblatt to solve classification problems. The perceptron as shown in Figure 7 is patterned after the neuron. It has weighted inputs which are then summed in the node. As the node's bias is overcome it "fires" raising the output to maximum. Rosenblatt used nonlinear activation functions such as the signum function. Today the nonlinear activation function of the perceptron is generally the sigmoid function which is described mathematically as:

$$\text{sigmoid}(\alpha) = \frac{1}{1 + e^{-\alpha}} \quad (1)$$

where

$$\alpha = \sum_{m=0}^{m=M} x_m \cdot w_m$$

$x_m \equiv$ input to node

$w_m \equiv$ weight between input x_m and node

The perceptron was soon found to be able to solve classification problems which were linearly separable, but was inadequate (33:189) for problems in which the classes were not linearly separable. This led to the formulation of interconnected sets of perceptrons in feedforward networks. The biggest problem was finding a training rule for the weights between the interconnected perceptrons. This was solved by Werbos (58) in his PhD dissertation and led to the current backpropagation architecture (46:324)(48:236). A typical backpropagation net is depicted in Figure 8. Note the similarity between the perceptron and the hidden layer and output nodes. The weights between the nodes are trained by presenting the net with a training set of exemplars from each data class and updating them based upon the change in the output as the weights and training vectors are varied.

The backpropagation feedforward neural network has been shown to have a Bayesian character (15, 24, 36, 45) in how it processes data during training. The Bayesian character is a direct result of the use of a mean squared error (MSE) cost criterion. This enables the backpropagation network to perform Bayesian estimation (53:283) of functions as well as Bayesian classification. The apparent ease with which the backpropagation network can be used for either function estimation or classification makes it ideal for the back end of the system architecture developed in this dissertation research.

The architectures explored in this dissertation led to a number of results which are of interest to the scientific community and are summarized in the next section. The methodologies used to obtain the results and the details of the results are given in chapters 2 through 5.

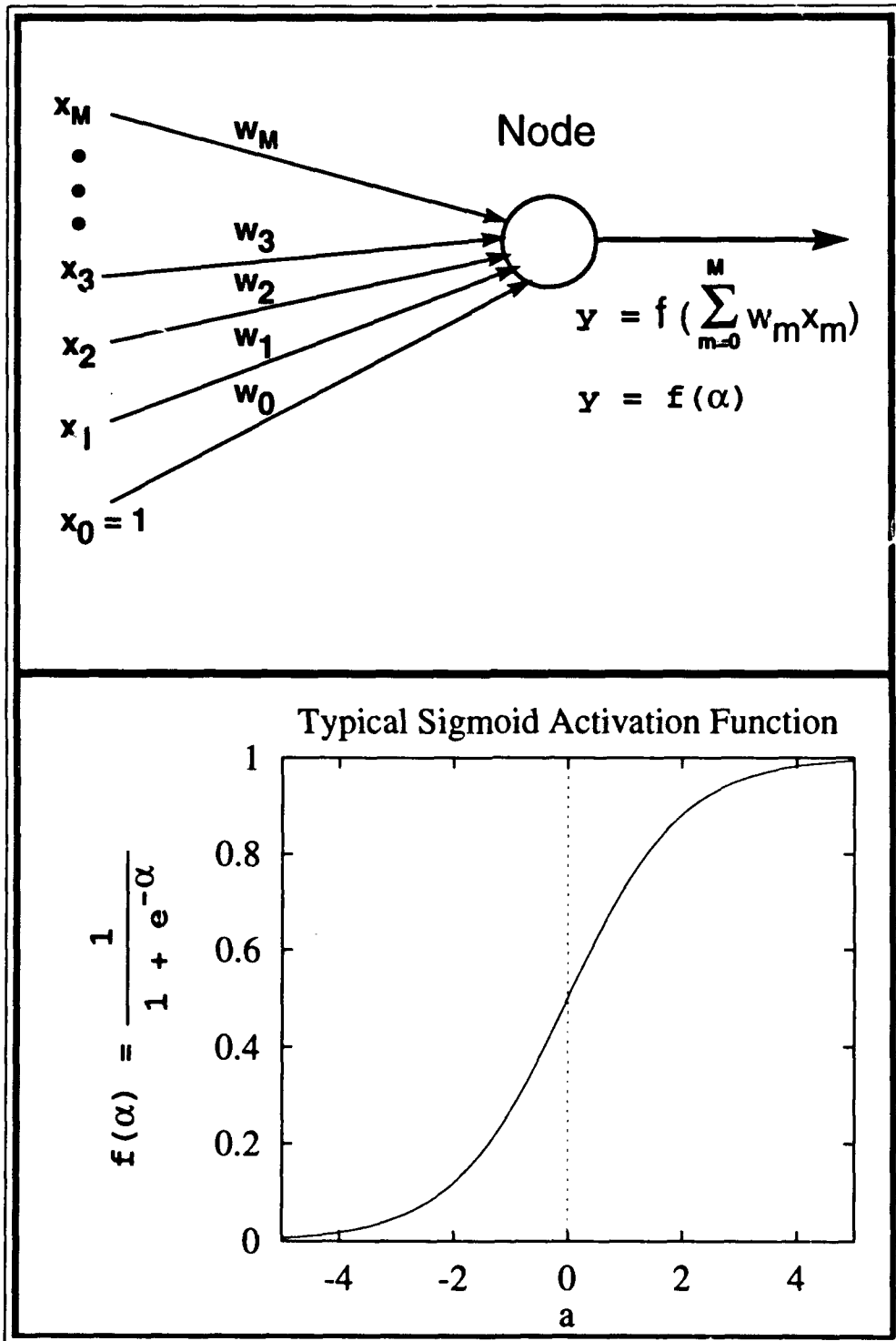


Figure 7. Diagram of Perceptron Node with A Sigmoid Activation Function

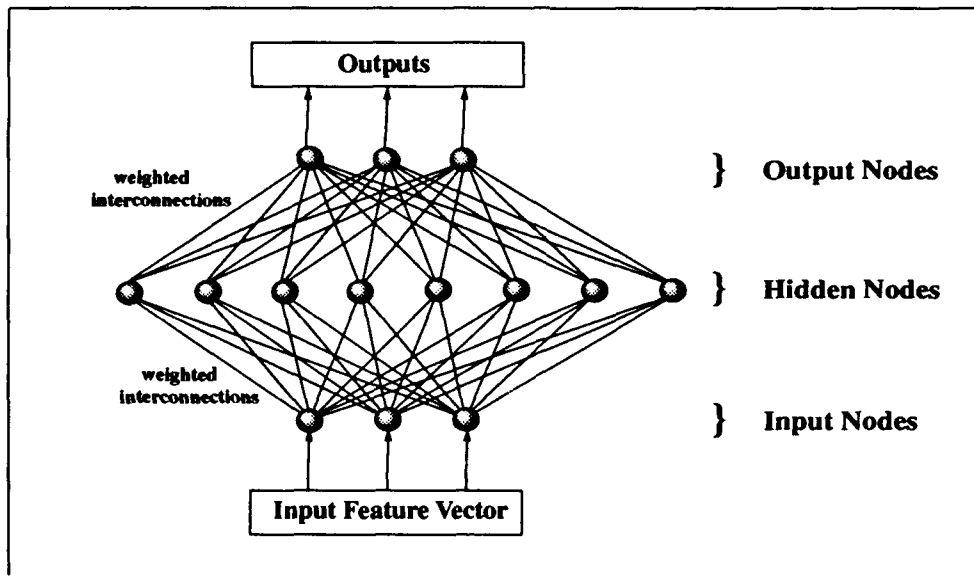


Figure 8. Diagram of Typical Backpropagation Net With A Single Hidden Layer

1.4 Significant Results

There are four major contributions to the scientific body as a result of the dissertation research: 1) A new architecture based upon the gestalt representation of objects using Gabor coefficients as inputs. 2) A new classification architecture which uses relative location information of selected textures as a method of classifying tactical targets. 3) A neural network architecture which allows for the addition of new classes without extensive retraining. 4) A feature saliency metric based upon the Bayesian nature of feedforward neural networks.

The neural network architectures used as classifiers for the target recognition problem were based upon two themes. The first was that of learning the gestalt representation (see Chapter 2) of each of the classes with the neural network performing as a function estimator with the classification task performed using a MSE measurement. The overall classification accuracy obtained using the gestalt representation was 62 percent which was an improvement over the 43 percent reported by Lazofson (27:70), but inadequate for a classifier. The second theme was to use the neural network as a Bayesian statistical

classifier for either a two-class or multi-class problem. The first use of the neural network as a Bayesian classifier was for a multi-class problem which used various distance metrics between the centroid of a cluster of the most resonant Gabor filter correlations and the locations of the resonant correlations. The distances between the centroid and each of the Gabor filter correlation peaks were used as feature vector descriptors for the target classes with the classifier performing as a Bayesian statistical classifier (see Chapter 3). The best result using the various distance metrics in Chapter 3 was a classification accuracy of 73 percent which was also inadequate. After the first two system architectures showed limited classification accuracy, a third approach using the relative locations of specific spatial features feeding a bank of two-class neural networks was developed as described in Chapter 5. The results obtained using the relative location concept were 100 percent for the three class problem used in the two previous approaches. This technique was then applied to a seven class problem which included crepuscular (twilight) conditions in the FLIR imagery. In the seven class problem the relative location technique was able to correctly classify 93 percent of the FLIR images. An outgrowth of the second approach was that it readily became apparent that a limitation on the number of features would be required to preclude memorization (4, 11) problems in the statistical classification scheme. This led to a major contribution in the area of feature saliency from a Bayesian point of view. Ruck (45) and Gish (15) proved in separate but similar analyses that the multilayer feedforward neural network is indeed a Bayesian classifier. Ruck showed that the outputs of a feedforward neural network, when trained under fairly minimal requirements, become the a posteriori class conditional probabilities when the network converges. A logical extension of the Bayes result was to explore using the probability of error as a means of determining which input features were the most influential, as described in Chapter 4. An interesting result of the analysis was that the intuitive metric proposed by Ruck (44) was actually the same (within a multiplicative scaling constant) as that obtained from the Bayesian analysis. This further removes any doubt as to the Bayesian behavior of feedforward networks. The results highlighted in this section are discussed in detail in

subsequent chapters of this dissertation.

1.5 Dissertation Overview

The general flow of the dissertation is organized along the path taken during the dissertation research. Chapter 2 discusses the gestalt representation and the associated architecture in detail along with the results obtained from the research. Chapter 3 presents alternative approaches to the gestalt method using traditional backpropagation classifiers while retaining the Gabor processing front-end and changing the way the Gabor coefficient information is processed for presentation to the classifier. Chapter 4 presents the feature saliency derivation along with the subsequent improvements in processing speed with lower computational cost while maintaining accuracy. Chapter 5 presents the relative location concept and the classification results. Chapter 6 presents conclusions and recommendations. Each chapter in the dissertation is written to be self-contained for better readability.

II. Using Gabor Filter Methods to Learn The Gestalt

This chapter explores the use of Gabor filtering methods in learning the gestalt representations of objects. The foundation for the use of Gabor filtering methods and neural network approaches to solving the pattern recognition problem are presented in the background section. The Lazofson architecture (27) is described in detail following the background. The concept of the gestalt is then introduced along with the rationale for modifying the Lazofson architecture to improve its classification performance.

2.1 Background

The basic concept (15) that will be developed and subsequently tested in this chapter is learning the "gestalt" of an image. The term gestalt is defined by Webster (34) as:

gestalt - a structure, configuration, or pattern of physical, biological, or psychological phenomena so integrated as to constitute a functional unit with properties not derivable by summation of its parts.

In an effort to learn the gestalt representation of a set of objects, some researchers (27, 28) have used the traditional backpropagation network fed by feature vectors obtained from using Gabor filter processing. Lazofson modeled much of his processing architecture after Le Cun (5) and Fukushima (12). Both Le Cun and Fukushima were solving a handwritten character problem by presenting the characters to a neural network architecture for classification. Le Cun used backpropagation networks to process the input character, while Fukushima uses a winner takes all approach in training his network nodes. Both of these approaches to solving the handwritten character problem are computationally intense. To better understand why these approaches are so demanding consider the Le Cun approach as shown in Figure 9 and the neocognitron as depicted in Figure 10. As can be seen in Figure 9 there are 256 input nodes, 768 nodes in the first hidden layer, 192 nodes in the second hidden layer, 30 nodes in the third hidden layer and 10 nodes in the output layer. Similarly, the neocognitron shown in Figure 10 uses approximately 35,000 processing

elements for its solution to the character recognition problem. Obviously the number of nodes required to solve the handwritten character task using either the neocognitron or the Le Cun architecture is significant. Lazofson (27:43) sought to combine the windowing ideas of Le Cun and Fukushima with results obtained from studies of the visual system in cats (16, 21, 22) to produce an improved recognition architecture. This was an ambitious undertaking because Le Cun and Fukushima both used characters on a pristine background while the forward looking infrared (FLIR) imagery selected by Lazofson used natural backgrounds. The Lazofson architecture is discussed in detail in the next section.

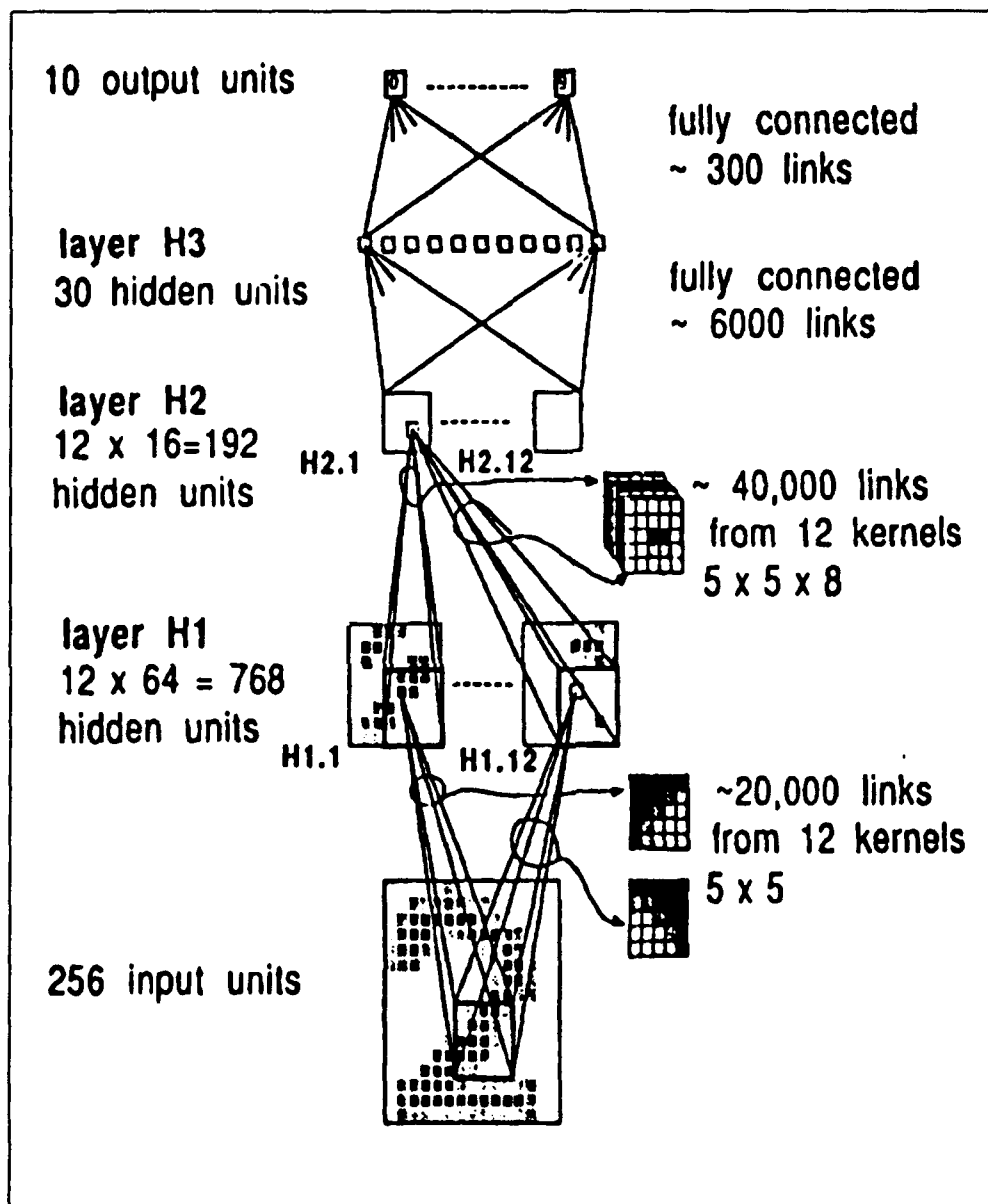


Figure 9. Architecture for the Solution of Handwritten Characters Developed by Le Cun (5:548)

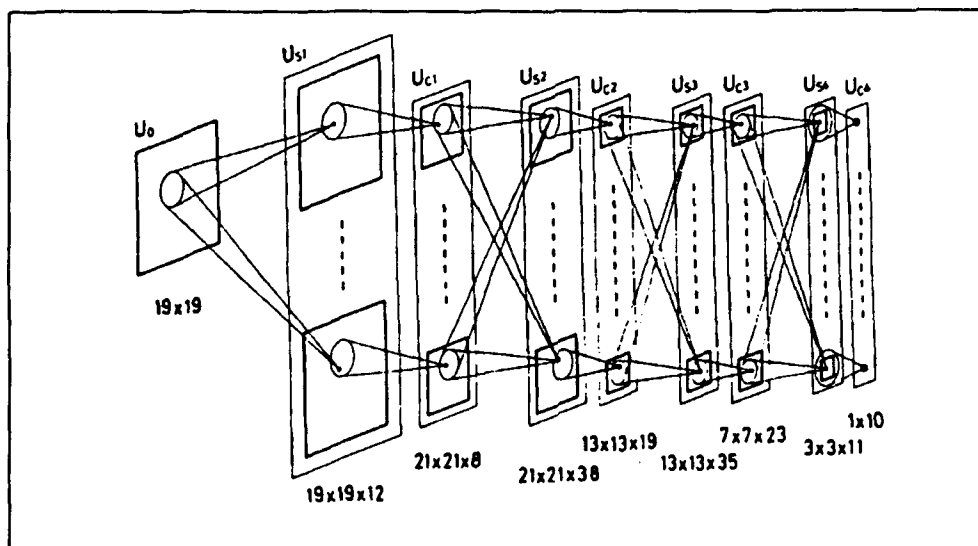


Figure 10. Neocognitron Architecture for the Solution of Handwritten Characters Developed by Fukushima (12:121)

2.2 The Lazofson Pattern Recognition Architecture

The pattern recognition system architecture used by Lazofson is shown in Figure 11. The architecture used an 8x8 roving window, termed the receptive field, which ran across the image with an overlap of one-half the window size. The portion of the image contained in the receptive field is processed through each of the four Gabor orientation group filters with the results for each of the 434 possible correlation values of each Gabor filter stored in an array. The "phase synchrony" summation nodes look at 5x5 windows shifted only one node per measurement across the Gabor orientation filter output array. Phase synchrony is a term which is descriptive of the locking up of neurons in the brain when certain stimuli are observed (27:22). Phase synchrony combines local information to express global information. Thus the phase synchrony nodes are combining groups of neurons which are tuned to the same orientation. These summation nodes result in four groups of 270 nodes or 1080 total which fed the output layer of the system network. The final output layer is a backpropagation neural network trained to solve a four class recognition problem.

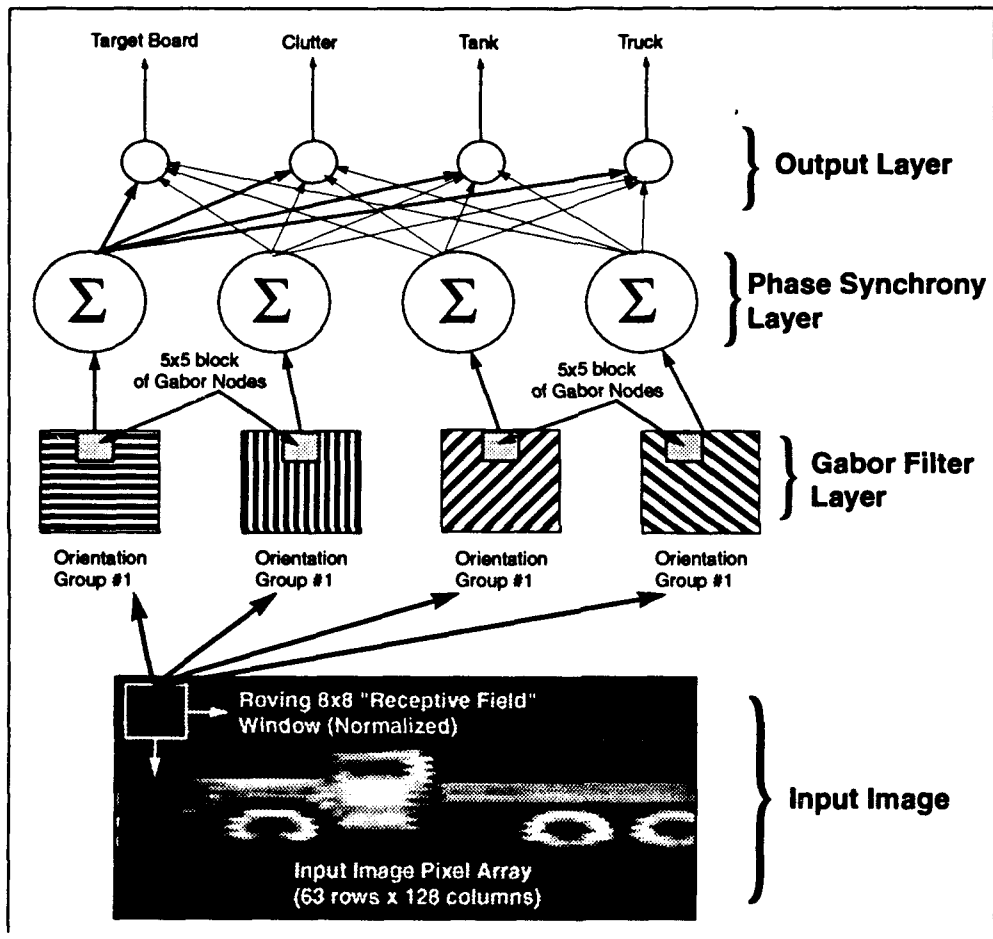


Figure 11. Biologically Inspired Pattern Recognition Architecture Used by Lazofson (27:45)

The results of the Lazofson architecture are shown in Figure 12. Lazofson was not able to obtain suitable results (3.2 percent) for the classification task. The dissertation research was started using the Lazofson architecture as a baseline with variations on the theme of localized windows and Gabor filter techniques.

One of the drawbacks to the Lazofson architecture was the large number of inputs (~1080). This presented a two-fold problem because of the computational time required to train the network and the required number of training samples to ensure that the solution obtained by the backprop output layer was unique. According to Foley (11), the number of training samples per class should be at least three times as large as the number of inputs (features) to a neural network. Because of the processing limitations and the limited number of training samples, a faster and more efficient method of extracting Gabor features for classification of FLIR imagery was developed. This new system architecture is developed in the next section and is termed the Gestalt network (GNET).

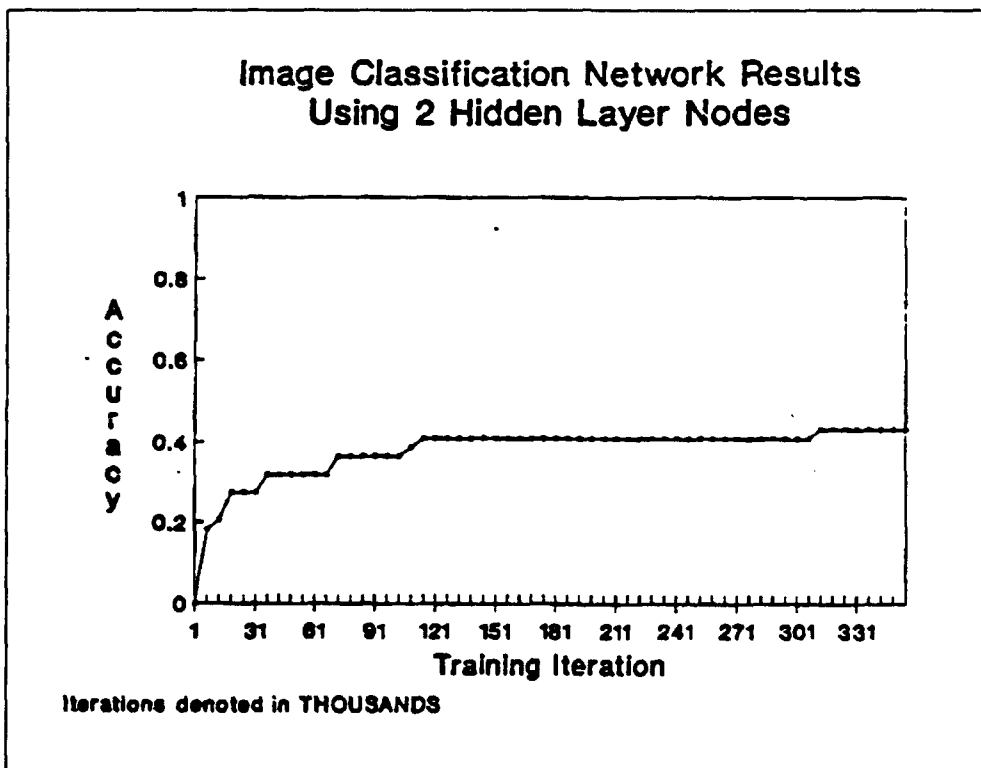


Figure 12. Image Classification Results For Lazofson Architecture (27:70)

2.3 The Gestalt Network System Architecture

The basic architecture of the Gestalt Network (GNET) pattern recognition system is depicted in Figure 13. Each portion of the GNET system will be described in detail later in this chapter. The GNET architecture consists of the input image, Gabor filters, neural networks and, finally, the classifier. Since each neural network is trained to recognize only one class, the neural networks.

The Gabor filters extract pertinent texture information from the input image which is then processed by each neural network. A windowed portion of the image is correlated with each Gabor filter with the result used to train the neural networks. The mean squared error (MSE) between the predicted Gabor correlation value and the actual Gabor correlation value is calculated for each network. The network with the smallest MSE is declared to be the winner and the input object is assigned to the class the network represents. The gestalt, "tankness", of each class is stored in the weights of its particular neural network. Thus the neural network forms a representation of the object which, hopefully, is unique based upon the Gabor data presented during training. The first component of the GNET architecture is the Gabor processing layer.

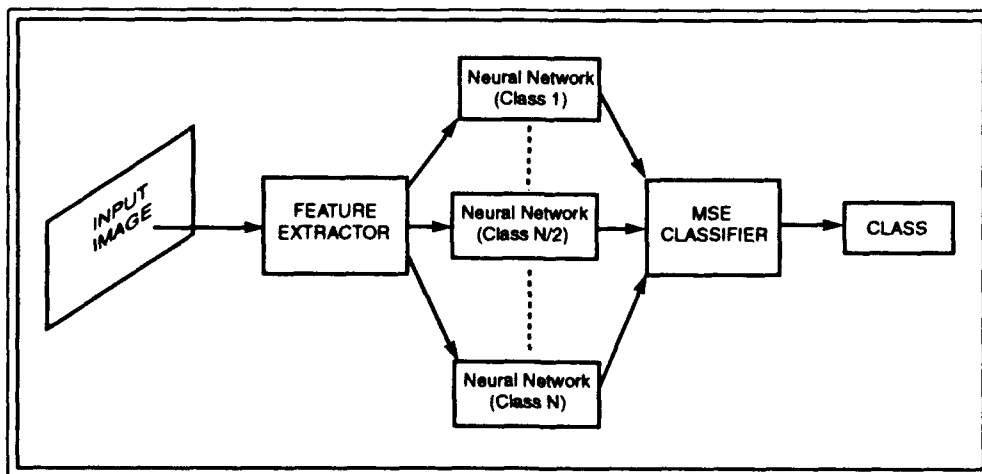


Figure 13. Neural Network Based Automatic Target Recognizer

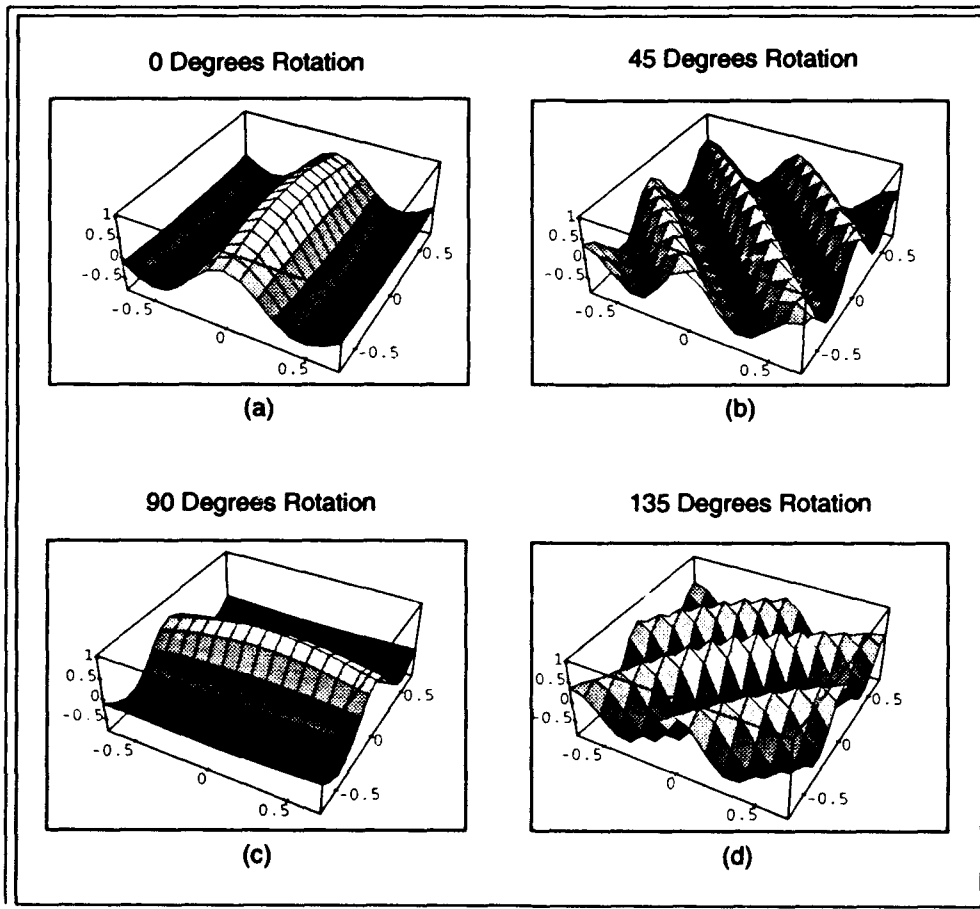


Figure 14. Two-Dimensional Cosine Gabor Function for Various Rotation Angles and Spatial Frequencies: a) 0 Degrees Rotation, b) 45 Degrees Rotation, c) 90 Degrees Rotation, and d) 135 Degrees Rotation

2.3.1 Gabor Processing A great deal of interest has been generated (2, 7) about the texture discrimination properties of the Gabor transform. The Gabor filters used in this research are two-dimensional filters such as those shown in Figure 14. They are represented mathematically by:

Two-Dimensional Cosine Gabor

$$G_c(x, y, f_x, f_y, \theta) = \frac{\cos(2 \cdot \pi(x \cdot f_x + y \cdot f_y) + \theta) \cdot e^{-\left(\frac{x-\bar{x}}{\sqrt{2} \cdot \sigma_x}\right)^2} \cdot e^{-\left(\frac{y-\bar{y}}{\sqrt{2} \cdot \sigma_y}\right)^2}}{2 \cdot \pi \cdot \sigma_x \cdot \sigma_y} \quad (2)$$

Two-Dimensional Sine Gabor

$$G_s(x, y, f_x, f_y, \theta) = \frac{\sin(2 \cdot \pi(x \cdot f_x + y \cdot f_y) + \theta) \cdot e^{-\left(\frac{x-\bar{x}}{\sqrt{2} \cdot \sigma_x}\right)^2} \cdot e^{-\left(\frac{y-\bar{y}}{\sqrt{2} \cdot \sigma_y}\right)^2}}{2 \cdot \pi \cdot \sigma_x \cdot \sigma_y} \quad (3)$$

where:

$x, y \equiv$ position in x, y direction

$\bar{x}, \bar{y} \equiv$ centroid of Gaussian envelope in x, y direction

$f_x, f_y \equiv$ x, y spatial frequency component

$\sigma_x, \sigma_y \equiv$ standard deviation of Gaussian in x, y direction

$\theta \equiv$ phase shift for sine/cosine wave

As can be seen in Eqs (2) and (3) on page 26, there is a great deal of flexibility ($x, y, f_x, f_y, \theta, \sigma_x, \sigma_y$) available to the filter designer. This flexibility allows a wide variety of Gabor filters to be constructed for implementation with just a couple of equations.

The Gabor processor is depicted in Figure 15. It consists of a series of two-dimensional perceptron (29) (57:28) arrays input image. As can be seen two-dimensional array has a receptive specific portion of the input image. sampled values of the two-dimensional of $(f_x, f_y, \sigma_x, \sigma_y, \theta)$ values which correspond to its filter.

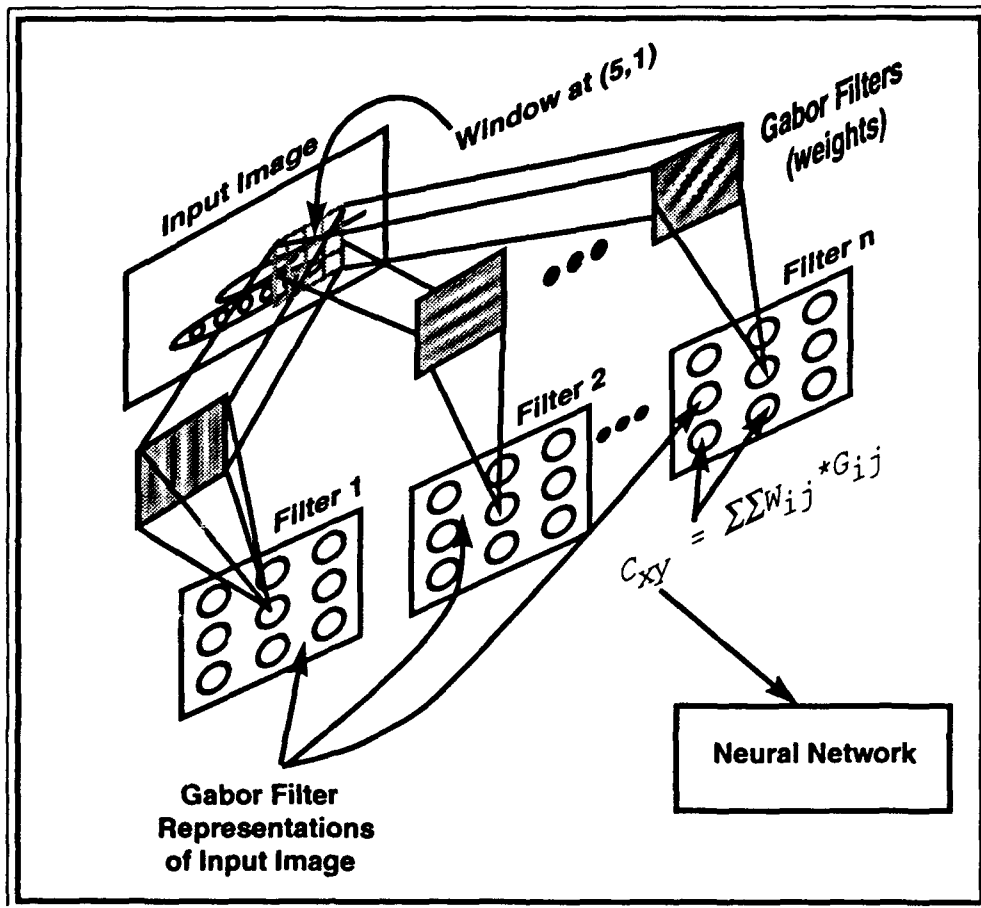


Figure 15. Simplified Diagram of Gabor Processor with Two-Dimensional Arrays

The input image is windowed and the data in each window is processed by subtracting the average pixel brightness in the window from every pixel value for the window and then each pixel value is divided by the average energy in the window to normalize the pixels relative one to another. This process, known as Lambertization (25:3-15), eliminates the dc bias present in the window and consequently edge enhances any objects in the

window. Then each window value is tied by a weight vector to a perceptron in each filter array. Because a linear output rather than a sigmoidal output is used for the perceptron, a correlation is obtained between the windowed portion of the input image and the Gabor filter. The correlation process is represented mathematically as

$$C_{xy} = \sum_i \sum_j I_{ij} \cdot G_{ij} \quad (4)$$

where:

$I_{ij} \equiv$ image value at position i, j in window

$G_{ij} \equiv$ associated Gabor weight value at position i, j in window

$C_{xy} \equiv$ correlation value at window position x, y in image

(5)

To further illustrate how a perceptron can be used as a correlator consider Figure 16 which is a diagram of the traditional perceptron. The inputs (x_i) to the perceptron for the system architecture are the sampled values of the windowed portion of the input image while the weights (w_i) are the sampled values of the two-dimensional Gabor filter which is matched to the window. The nonlinear activation function (f) of the perceptron is replaced by a constant activation function, producing a linear output.

The perceptron nodes form the unit building blocks of the network feature extractor. Because each image may need several Gabor filter descriptions to properly extract the features, several perceptrons with different sets of weights may be tied to the same window position as depicted in Figure 15.

Once the image is presented to the architecture, the correlation values are calculated by the perceptrons. Because each individual perceptron has only one receptive field, its output can be thought as being a mapping from a window in image space to a point

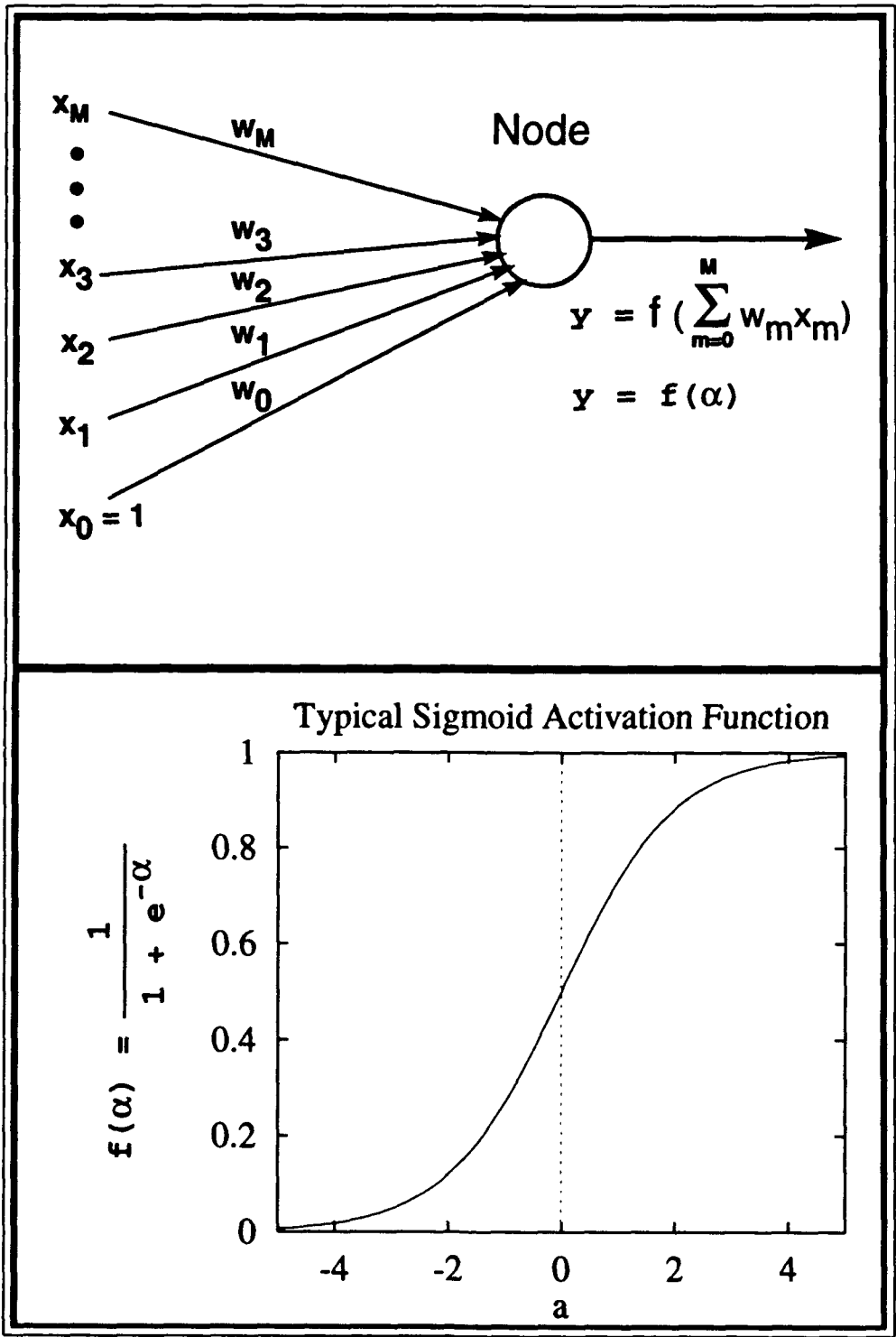


Figure 16. Diagram of Perceptron Node

in correlation space. This mapping concept allows us to consider the two-dimensional perceptron arrays shown previously in Figure 15 as correlation views of the image.

Because the mapping is for particular receptive fields, each perceptron's output is precisely determined as a function of window position (x, y) and filter with frequency (f_x, f_y) . The inputs (x, y, f_x, f_y) and their respective correlation value, $C(x, y, f_x, f_y)$, can be used to train neural networks in an effort to learn the "gestalt" of an image.

2.3.2 The Neural Network Processor The neural network processor is actually a collection of specific neural networks each trained on only one class. The inputs to the network are (x, y, f_x, f_y) with the correlation value (perceptron output) used to train the network and later test the network.

The network used in this research is a three-layer (two hidden, one output) back-propagation network (47) as shown in Figure 17. inputs with twenty nodes in the first sigmoidal the $C(x, y, f_x, f_y)$ value given the input (x, y, f_x, f_y) . Once training is complete, $\sim 100,000$ iterations, no longer be trained. inserted the $C(x, y, f_x, f_y)$ value based upon over the entire input data set. The output of the network is then a MSE value which represents how well the input images matches its gestalt of the image. The outputs of each class neural network are then fed to the MSE classifier.

2.3.3 The Mean Squared Error Classifier The MSE classifier shown in Figure 13 compares the outputs of each of the neural network processors and assigns the input image to the class represented by the neural network processor with the smallest MSE. The mean squared error is calculated using Eq (6).

$$\text{MSE} = \frac{\sum_{(f_x, f_y)=1}^K \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} (C(x, y, f_x, f_y) - \tilde{C}(x, y, f_x, f_y))^2}{K \cdot M \cdot N} \quad (6)$$

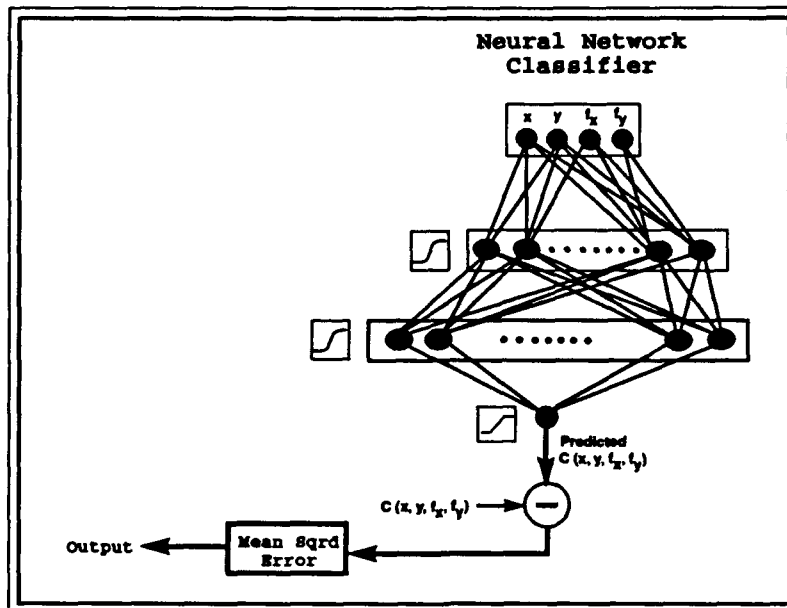


Figure 17. Diagram of Single Neural Network Processor

where

$MSE \equiv$ Mean Squared Error

$f_x, f_y \equiv$ spatial frequency pair for Gabor filter

$x \equiv$ window location in x direction within the image

$y \equiv$ window location in y direction within the image

$K \equiv$ total number of spatial frequency pairs

$\tilde{C}(x, y, f_x, f_y) \equiv$ predicted correlation coefficient

$C(x, y, f_x, f_y) \equiv$ actual correlation coefficient

After the mean squared error is calculated for each “gestalt” neural network for the image in question, the class is declared to be the class of the network with the smallest

MSE. Once the class assignment is made classification is complete.

2.4 Understanding the Gestalt Representation

The gestalt representation of the input image is represented by a five-dimensional surface (x, y, f_x, f_y) comprised of $C(x, y, f_x, f_y)$ values. An analog to this five-dimensional problem which is easy to see in three dimensions is that of a cylinder which is heated on one end and cooled on the other. For each (x, y, l) position, where l is length, there is a corresponding temperature value $T(x, y, l)$. Hence, in the five-dimensional space there is a corresponding value for the correlation coefficient $C(x, y, f_x, f_y)$.

In an effort to better visualize this representation process we will convert from (f_x, f_y) to the polar coordinate system with the following definitions:

$$f_x = \rho \cdot \cos(\phi) \quad (7)$$

$$f_y = \rho \cdot \sin(\phi) \quad (8)$$

$$\rho = \sqrt{f_x^2 + f_y^2} \quad (9)$$

$$\phi = \arctan\left(\frac{f_y}{f_x}\right) \quad (10)$$

where:

ρ = modulation in cycles per window

ϕ = angle of rotation in clockwise direction

Once the polar conversion is made, a fixed value for ρ will yield different filters as ϕ is allowed to vary as shown in Figure 18. If all possible values for ϕ are plotted a circle of filters would result as indicated by the dashed circle in Figure 18. By allowing ρ to

vary as well we would simply be defining an area of possible filter values as depicted in Figure 19.

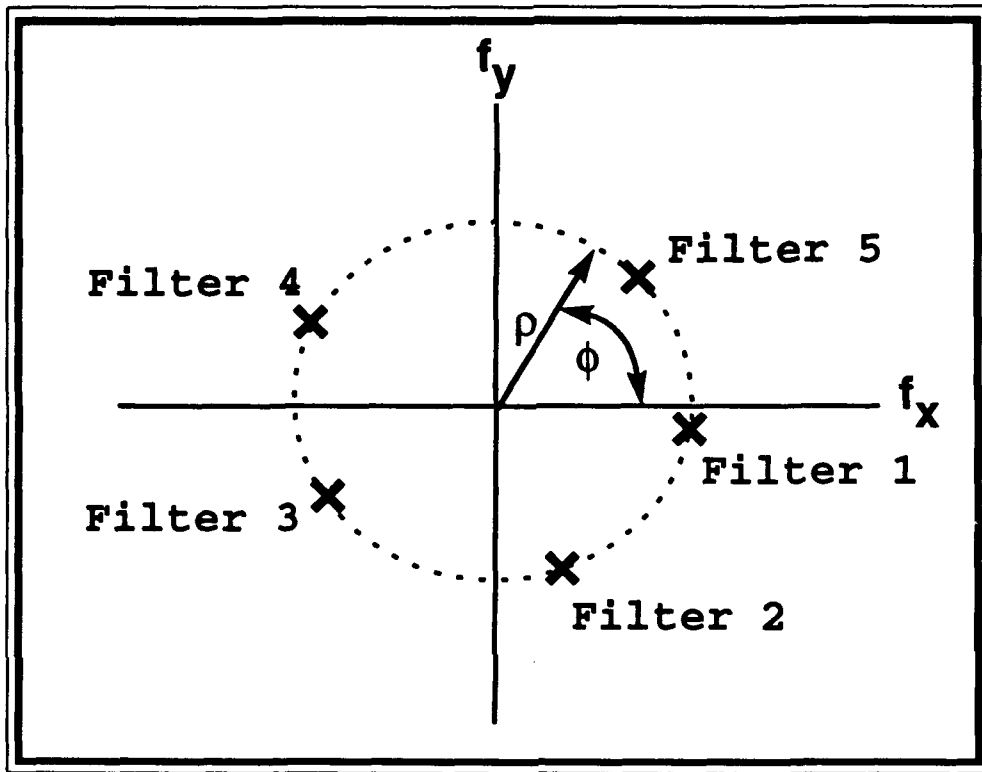


Figure 18. Filters With Fixed (ρ) Variable (ϕ) in (f_x, f_y) Coordinate System

Now extending this concept to higher dimensions we would note that in the case of three dimensions volumes of possible filter values can be defined and in even higher dimensions, hypervolumes of possible filter values could be defined. Because each perceptron in Figure 15 has a unique receptive field and *assuming* that adjacent nodes don't yield an appreciable amount of information about the input image, we could use linear arrays of perceptrons rather than two-dimensional arrays of perceptrons in the Gabor processor. This change is depicted in Figure 20.

Expanding our representation to three dimensions, with the linear arrays, we can represent the five-dimensional space (x, y, f_x, f_y) as a three dimensional space (ρ, ϕ, l),

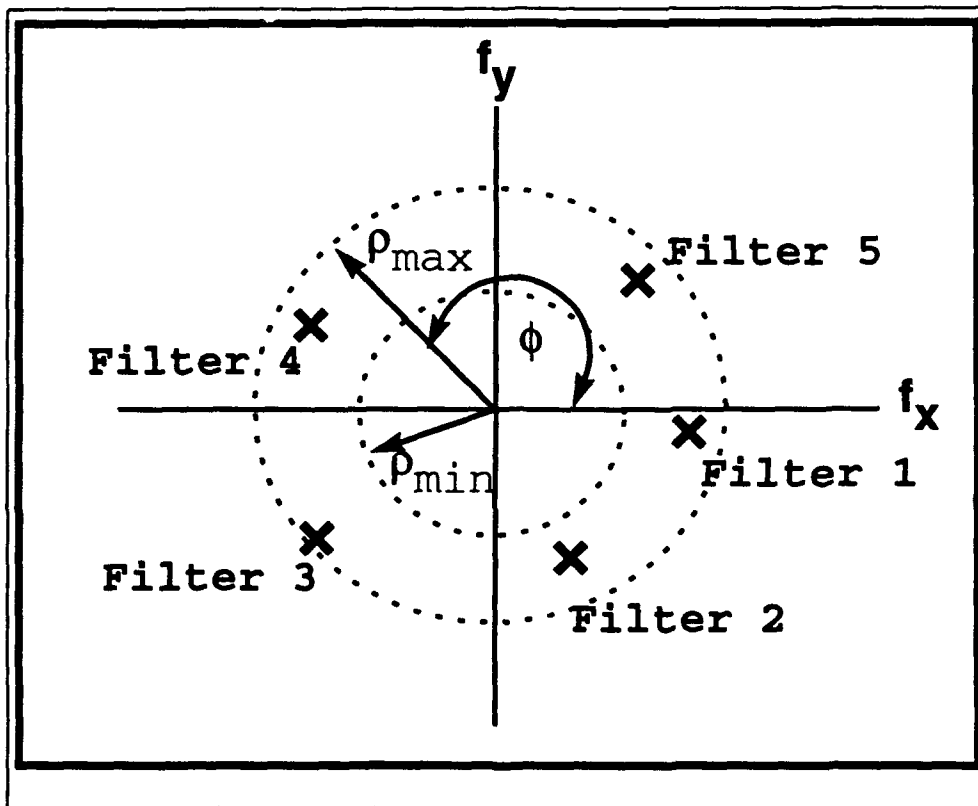


Figure 19. Filters With Variable (ρ, ϕ) in (f_x, f_y) Coordinate System

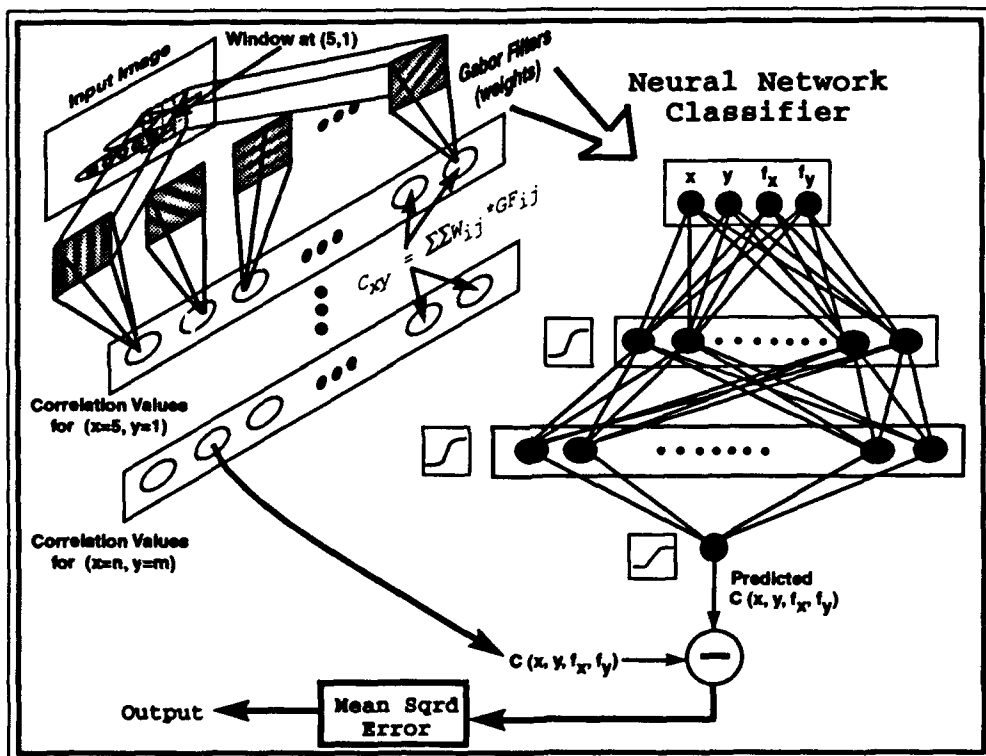


Figure 20. Simplified Diagram of Gabor Processor with Linear Arrays

where l is the window, as shown in Figure 21. Each window would be represented as a discrete position l along the axis of the cylinder, with the particular filter for that window represented by (ρ, ϕ) or (f_x, f_y) .

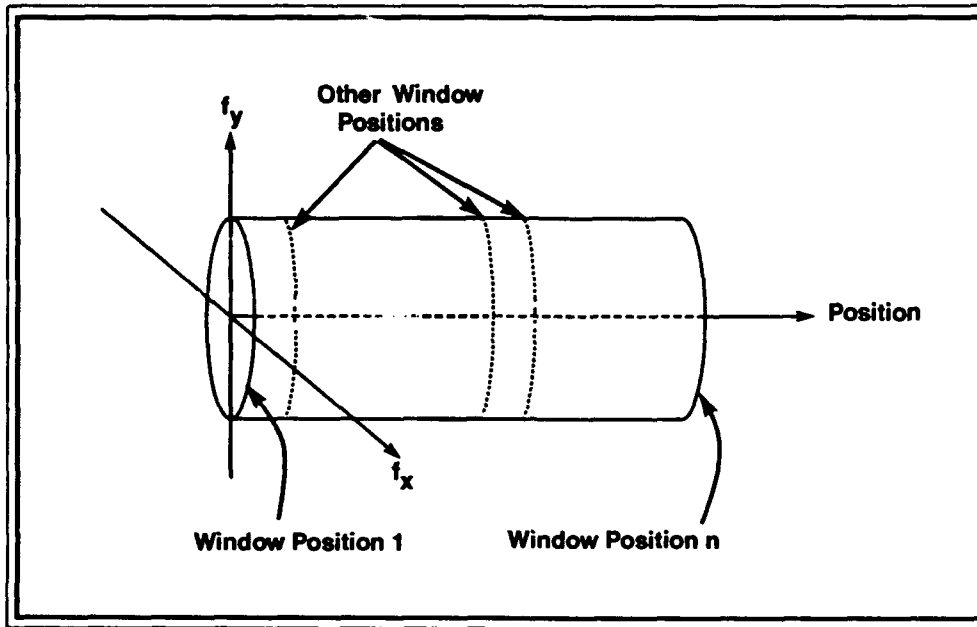


Figure 21. Three Dimensional View of Correlation Space

For constant ρ and all possible values of ϕ and l we would have filter data for the outer surface of the cylinder. As we begin to vary ρ the data points within the cylinder begin to fill. Because each filter in the current architecture is tied to every window a particular (f_x, f_y) pair would appear as a line segment through the cylinder along the position axis. The $C(x, y, f_x, f_y)$ values for each filter/window combination could then be represented by the temperature in the three-dimensional analogue explained earlier. Hence, the cylinder describes the known filter data points, while the actual correlation values are the outputs of the particular perceptrons which have the proper (x, y, f_x, f_y) identification.

Now that we can visualize the correlation surface as the temperature as a function of position in the cylinder, we can begin to see how a, hopefully unique, description of each

class is possible. Charged with the zeal of our new-found knowledge we proceed to test our system with the results presented in the next section.

2.5 *Testing the Notion of Learning the Gestalt*

The next step in the process was to test the GNET architecture on a simple classification problem which is depicted in Figure 22. The gestalt concept is dependent upon the backprop network's ability to reduce the error between its estimate of the $C(x, y, f_x, f_y)$ coefficient and the actual $C(x, y, f_x, f_y)$ to an acceptable margin before attempting the classification task. Typical mean-squared error (MSE) versus training iteration curves for each of the classes are given in Figure 23. Note that the MSE for each class is small versus the range of the data. Typical $C(x, y, f_x, f_y)$ values range between -2 and 2. The actual error between the predicted $C(x, y, f_x, f_y)$ and the actual $C(x, y, f_x, f_y)$ value is only 0.03. Thus the backprop network was doing a good job of estimating the gestalt of each class. Tables 1 and 2 show the results of the MSE classifier for the two four class problems depicted in Figure 22. The columns in the tables represent the actual class of each network (the class used to form the weights), while the rows represent the class of the input image. The figures of merit (FOMs) in the tables are given by

$$\text{FOM} = \frac{\text{MSE}}{\text{smallest MSE of all Nets}} \quad (11)$$

As can be seen in the tables all of the low FOM values correspond to the proper class. The results presented here show that the combination of Gabor filters and neural networks can solve pattern recognition problems.

Armed with the success of the "toy" problem, the next problem attempted was identification of tactical targets obtained from FLIR imagery as shown in Figure 24. The tactical target data set constrained a total of 53 targets. One of the targets from each class was chosen as a template and used to train a gestalt network. Once the three gestalt networks were trained, the entire data set was tested with the results given in Table 3. The

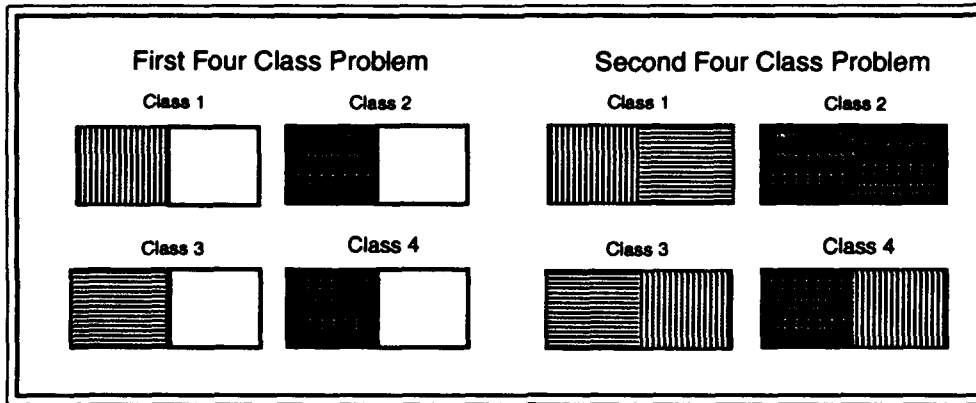


Figure 22. Class Problem for Network Training

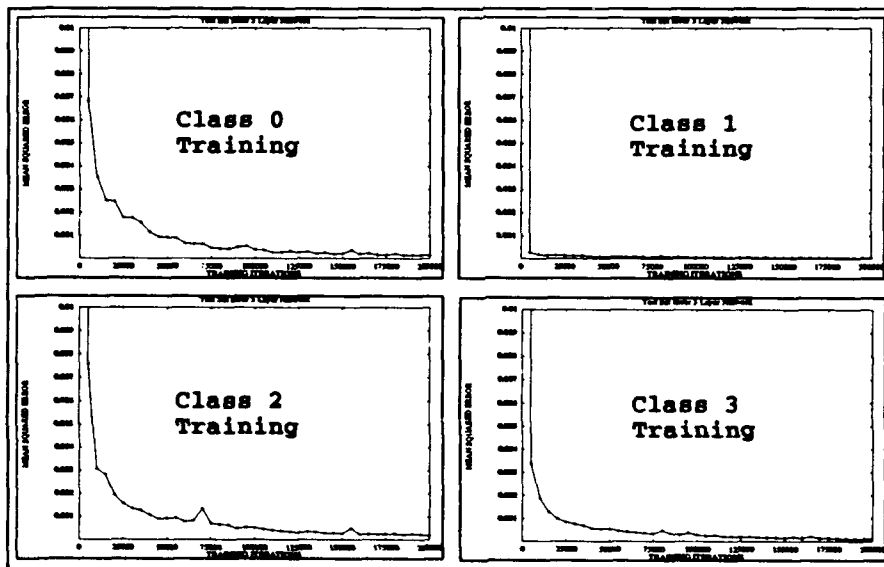


Figure 23. Mean Squared Error as a Function of Training Iteration For Each of The Classes in The Four Class Problem.

<i>First Four Class Problem</i>				
	Class 1	Class 2	Class 3	Class 4
Image 1	1	75.7	10.2	50.4
Image 2	49.1	1	10.3	12.2
Image 3	570.8	894.7	1	634.3
Image 4	46.7	23.1	10.4	1

Table 1. Figures of Merit for First Four Class Problem

<i>Second Four Class Problem</i>				
	Class 1	Class 2	Class 3	Class 4
Image 1	1	89.0	3.4	7.1
Image 2	9.0	1	5.8	4.3
Image 3	6.3	71.6	1	3.2
Image 4	9.7	64.0	4.6	1

Table 2. Figures of Merit for Second Four Class Problem

tactical target result of 62.3 percent is inadequate for a classifier. To improve upon the results, alternative architectures were used to classify the tactical target data set.

<i>Tactical Target Results</i>			
	Total Targets	Total Correct	Percent Correct
Tank	21	11	52.4 %
Truck	23	19	82.6 %
Target Board	9	3	33.3 %
Overall	53	33	62.3 %

Table 3. Results For Tactical Target Problem

The gestalt representation scheme presented in this chapter may at first appear to be complicated but in practice it is fairly straightforward to implement. The backpropagation neural network with linear output nodes was shown to have the capacity to learn the gestalt representation of input images based upon spatial features obtained from the use

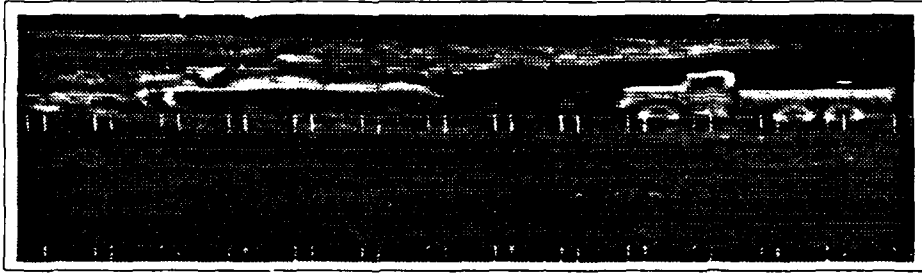


Figure 24. FLIR Image of Tank and Truck in Clutter

of Gabor functions as filters. The concept was tested upon simple spatial patterns and shown to be adequate for recognition tasks. The real test of any recognition system or technique is its ability to perform on real world data. Because of the low performance of the gestalt network on the tactical target data, other classification schemes were developed to improve performance. The next chapter examines the use of alternative classification schemes using the Gabor front end tied to a roving window across the scene of interest.

III. Alternative Classification Schemes Using Distance Metrics

In the previous chapter the gestalt network (GNET) architecture was presented along with results for a simple problem and a tactical target problem. This chapter will present the research efforts used to overcome some of the problems associated with the GNET architecture. As a baseline for comparison a traditional backpropagation neural network solving a three class problem (tank, truck, target board) was trained using the $C(x, y, f_x, f_y)$ coefficients for each window position as features. The typical window used for processing the scene was (8x8) pixels with one-half of a window overlap for each successive receptive field within a (64x128) FLIR image. This resulted in a total of 465 $C(x, y, f_x, f_y)$ values for each filter. Typically four filters were chosen for the segmentation task. This resulted in 1860 features to be sent to the backpropagation network. With 1860 features and only 53 targets available, the network would be able to memorize the test data (4, 11, 31). Because Foley's (11) rule is violated with this arrangement, the number of features presented to the network must be reduced while maintaining the relative locations and Gabor textures as inputs to the neural network classifier. The first approach attempted limited the number of features used by choosing the top N Gabor coefficients and their locations and then processing the data using a variety of distance metrics.

3.1 Choosing the top N Gabor Coefficients and Their Locations

The initial feature vector data set was generated by choosing an (8x3) window with a two cycles per window cosine Gabor function at angles of (0, 15, 30, . . . 150, 165) degrees which mimics the biology (6, 55, 16, 22). Once these windows are mapped over the input a composite image is formed from the various filter representations depicted in Figure 25. The composite image is created using the rule below:

$$\text{if } |Gabor[k][i][j]| > |Composite[i][j]|$$

$$Composite[i][j] = Gabor[k][i][j] \quad (12)$$

where

$k \equiv$ Gabor filter image number (one per filter)

$i \equiv$ x window location

$j \equiv$ y window location

$Gabor[k][i][j] \equiv$ coefficient of Gabor filter image k at location i, j

$Composite[i][j] \equiv$ Composite image coefficient at location i, j

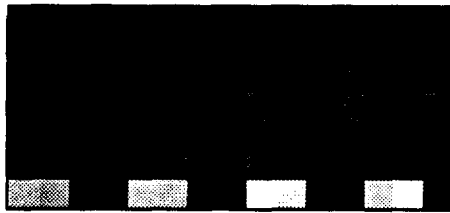
Once the composite image is formed the top N (by magnitude) Gabor values and their locations are selected. These N values and their locations form the 3·N features ($Composite[i][j], x, y$) of the feature vector which represent the gestalt of the image. This process is continued for all of the images in the data set. Once these feature vectors are calculated, they are then processed for input to a multilayer perceptron classifier. The final results of this process are given in Figure 26. As can be seen in Figure 26, the location of the Gabor coefficient with the largest magnitude is marked with an "X" while the next nine largest magnitude locations are marked with a number 0..9. These locations are the reduced representation of the input image.

This method produced moderately successful results with fairly good training. The typical run had an overall training accuracy of 85 percent while averaging 68 percent on test set accuracy. The overall results were not satisfactory, although for the same test set as Lazofson (27:69) the test set accuracy was better (68 percent vs 43.2 percent) using fewer features (30 vs 1080). One of the key ideas for the research effort was not yet invoked at this point and that idea was to use the relative locations of associated textures as features to the neural network classifier. The initial hypothesis was that the network would learn the importance of these relative textures such as wheels or cabs and their relationship one to another from the data set. Unfortunately the neural network was unable to learn the

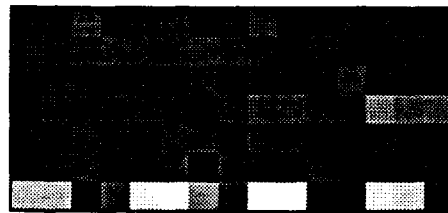
Original FLIR Image



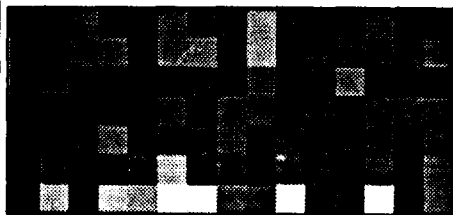
Correlation Images



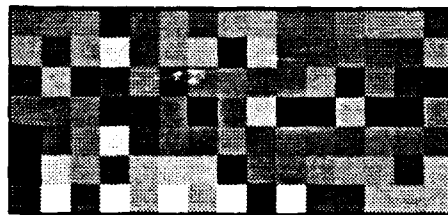
Filter = 0 Degrees



Filter = 15 Degrees



Filter = 30 Degrees



Filter = 45 Degrees

Figure 25. Truck With Associated Gabor Filter Images (16x16) Window.

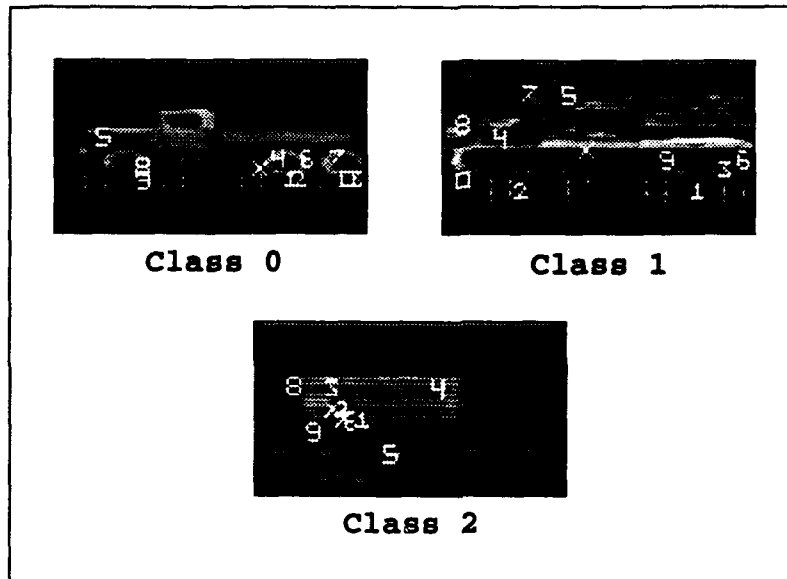


Figure 26. Three Class Problem Overlaid With Top Ten Gabor Locations.

spatial relationships from the data set. Because of the failure of the GNET to learn the spatial relationships, several sophisticated data representations were attempted. The first approach was that using the change in the (x, y) positions between the N maxima. This method is termed the Delta method.

3.2 Delta Method

The Delta method was simply a measure of the difference in the x and y positions between each of the N maxima. Once the maxima are found they are sorted from highest to lowest value. The position of the maximum magnitude is then set as the origin for the set. Then the deltas are calculated for each of the other $N-1$ maxima from the origin. The feature vector is made up of the origin and then the consecutive delta pairs from the origin. A typical run for the Delta method is given in Figure 27. As can be seen in Figure 27 the test set accuracy is 45 percent after 20,000 iterations.

The limited success obtained with this method led to the use of a Euclidean distance metric.

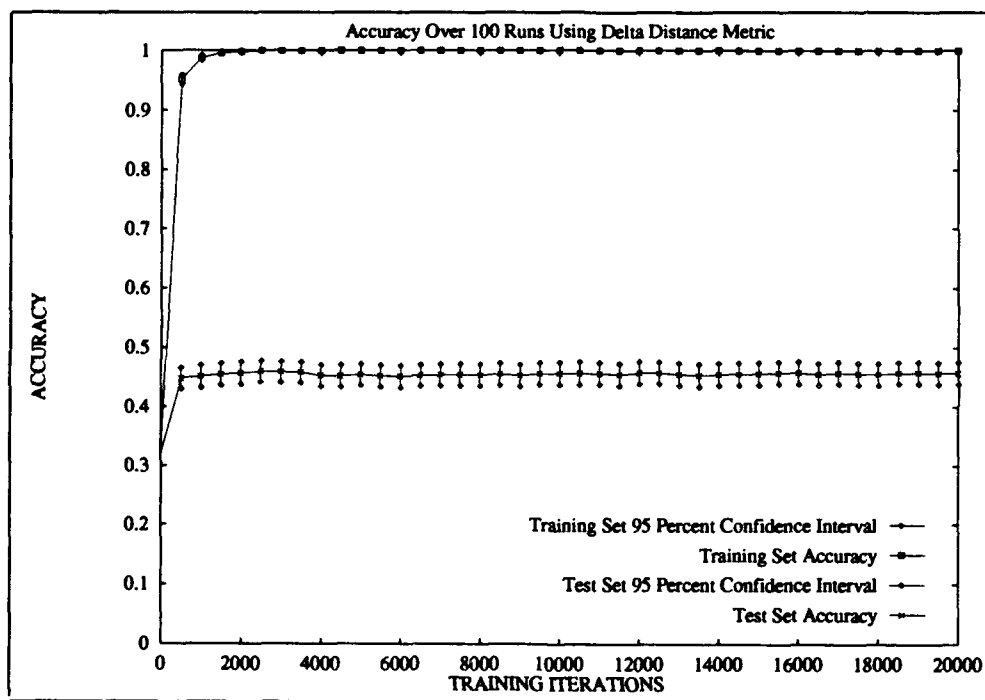


Figure 27. Comparison of Training Accuracy and Test Set Accuracy For Delta Rule

3.3 Euclidean Distance Metric

The Gabor coefficients were again rank ordered from the composite image with the top N coefficients selected. Then the Euclidean distance from the maximum to each of the N-1 Gabor coefficients was then calculated. The results of this method are given in Fig 28.

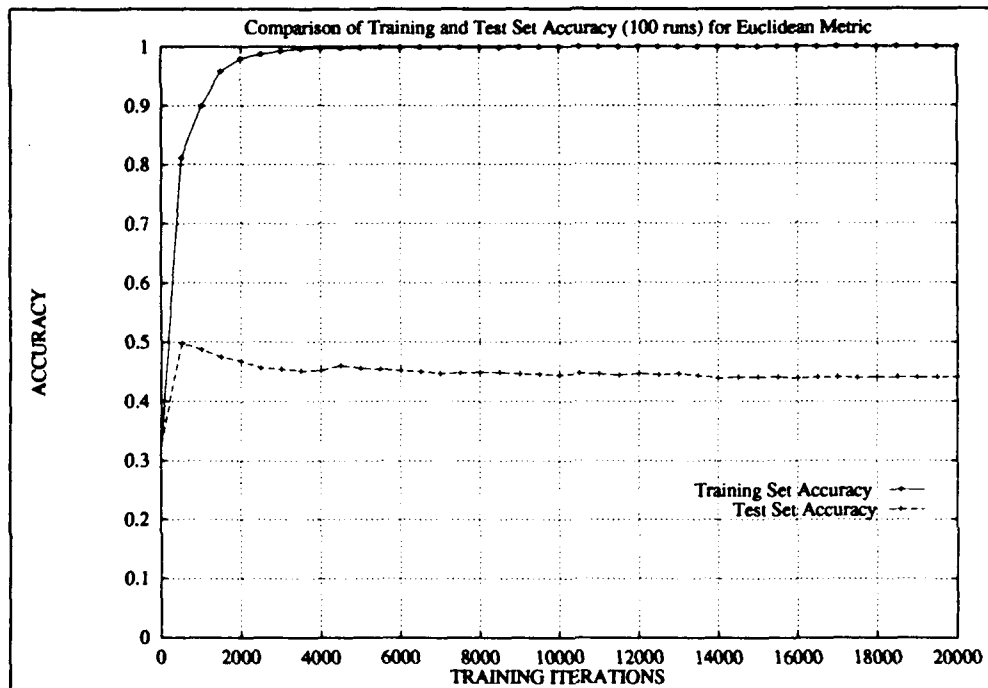


Figure 28. Comparison of Training Accuracy and Test Set Accuracy For Euclidean Metric

As can be seen in Figure 28 the overall test set accuracy was only 45 percent. This result led the research in a direction to improve performance while maintaining the character of relative locations of spatial textures. The obvious place to look for clues lies in the locations of the maxima themselves. The biggest problem was due to the shifts of targets within the (64x128) picture window. The target shifts resulted in different locations for the top N Gabor maxima. The sensitivity of the Gabor filters to the texture they were correlated with caused the values of the correlation to vary widely even if the target had shifted only a few pixels in the scene. This is best illustrated in Figure 29 where the

shift of the maxima can be observed while the target is shown to be nearly identical in each picture. The next section describes a series of metrics which eliminated most of the problems due to the overall shift of a target within the (64x128) picture window and which provided scale invariance as well. These metrics are all based on using the centroid of the N maxima as the origin and then performing traditional distance metrics. The class of metrics can be collectively termed centroid metrics.

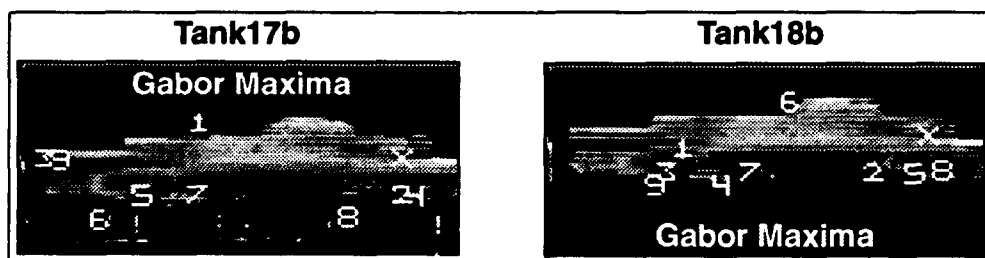


Figure 29. Shift of Gabor Maxima For Two Similar Images

3.4 Centroid Metrics

The previous sections described two metrics which were based upon the distance from the maximum to each of the other N-1 Gabor coefficients. This process while an interesting method was very susceptible to changes in the input image. For instance the method is very dependent upon the position of the object in the scene. The result of these dependencies was that the position of the maximum Gabor coefficient would change for different FLIR images for the same class. This is best understood by observing Figure 29. Even when the Gabor maximum is in the same approximate location, the other maxima are changing positions. Because of this problem, the next step employed to improve performance was to use the relative distances from the centroid of the ensemble of N locations corresponding to each of the Gabor coefficients as given in Eq (13).

$$Centroid[x][y] = \sum_{i=1}^N \left[\frac{x_i}{N} \right] \left[\frac{y_i}{N} \right] \quad (13)$$

where

$Centroid[x][y] \equiv$ the x,y position of the centroid

$x_i \equiv$ x position of Gabor maxima i

$y_i \equiv$ y position of Gabor maxima i

$N \equiv$ total number of Gabor maxima

Once the centroid was selected the distances from each of the maxima to the centroid were rank ordered from longest to shortest with the longest distance used to normalize the data vector. The first element in the data vector was then discarded because it was unity for all data classes due to the normalization scheme. The normalization provided invariance to scale while the rank ordering provides rotation invariance. Once the centroid was determined, various distance metrics were used to create feature vectors. The first metric used was the Euclidean distance metric.

3.4.1 Centroid Euclidean The Euclidean distance from the centroid calculated by Eq 13 to each of the N maxima locations was calculated with the longest distance occupying the first position in the feature vector and the other N-1 distances in descending order. After the distances were rank ordered then each distance feature in the vector was divided by the longest distance. The normalized features were then stored in a feature vector which was sent to a backpropagation network solving a three class problem. The results obtained using this method were much better than those previously obtained from the standard backprop classifier trained on the locations and their Gabor coefficient. The results of using the centroid Euclidean metric are given in Figure 30. Figure 30 shows the results for 100 runs using different seeds for initial weight values, training presentation and partitioning the training and test set. The results are much better with a peak in performance of approximately 62 percent at 2000 iterations. Because the number of vectors for two of

the classes are roughly equal at 40 percent apiece and the remaining class at 20 percent, the reported accuracy is well above that obtained from guessing a class at random (33 percent) or that obtainable from only guessing the class with the largest number of vectors (44 percent).

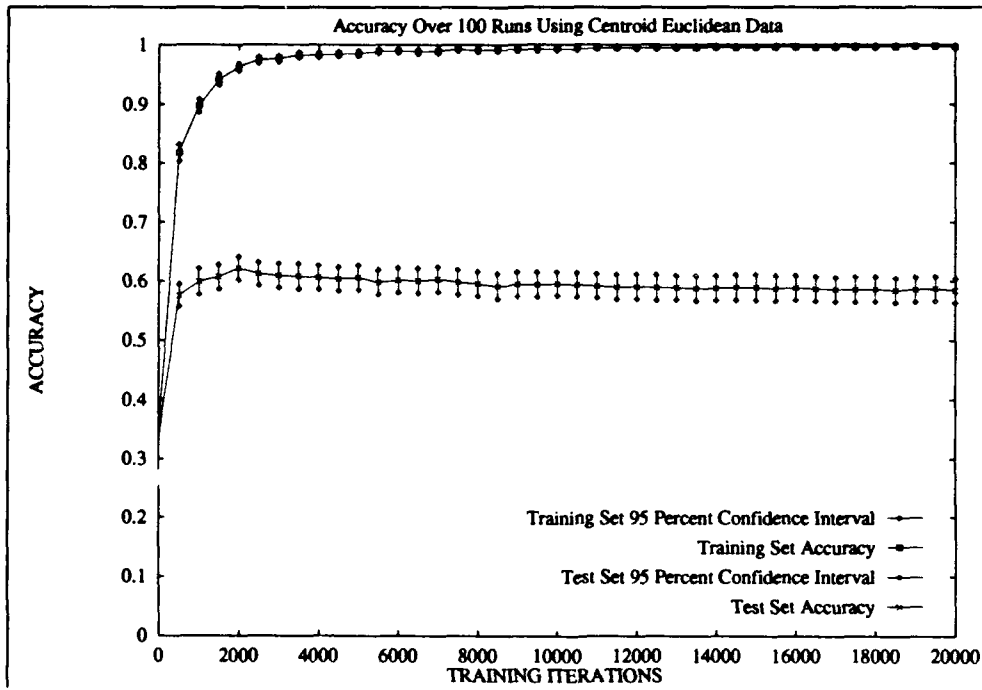


Figure 30. Comparison of Training Accuracy and Test Set Accuracy For Centroid Euclidean Metric

While this was very encouraging, other methods still remained for testing the concept of relative locations of spatial textures.

3.4.2 Centroid Taxi Distance The next distance metric for measuring between maxima was that of the taxi-cab distance, as given in Eq 14, from the centroid to each of the N locations and then ordering the distances from longest to smallest. The results obtained using the taxi metric were similar to those obtained using the centroid Euclidean metric and are given in Figure 31.

$$taxi\ distance = \sum_{i=1}^N |Centroid[x] - Max[i][x]| + |Centroid[y] - Max[i][y]| \quad (14)$$

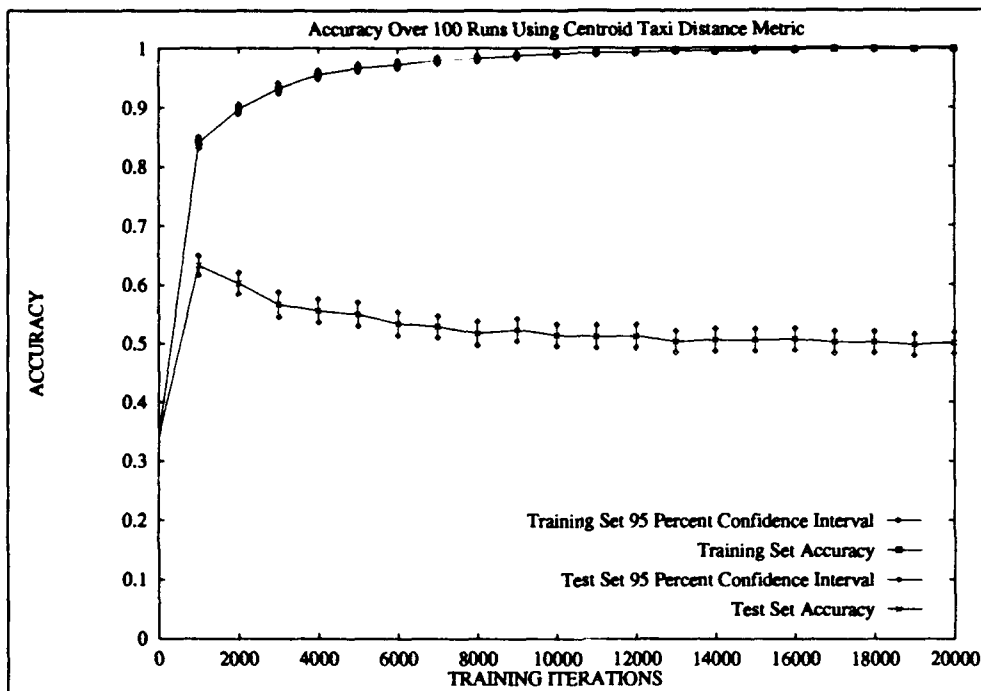


Figure 31. Comparison of Training Accuracy and Test Set Accuracy For Centroid Taxi Metric

3.4.3 *Centroid Delta* The centroid delta metric was extracted as shown in Figure 32. After each of the top N Gabor coefficients and their Δ_x, Δ_y locations were recorded they were presented to the neural network classifier for training and classification. The results obtained using the centroid Delta metric were similar to those obtained using the delta metric.

The centroid delta metric is calculated using

$$\text{delta } x[i], y[i] = | \text{Centroid}[x] - \text{Max}[i][x] | , | \text{Centroid}[y] - \text{Max}[i][y] | \quad (15)$$

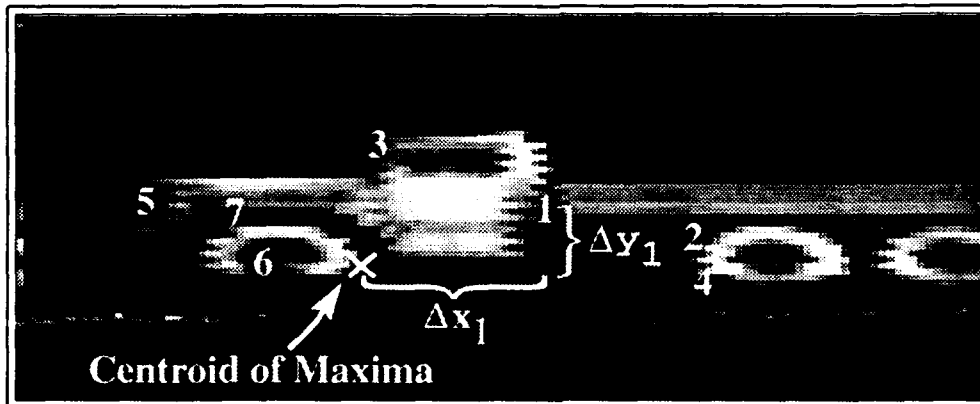


Figure 32. Extraction of Centroid Delta Position Data

3.5 Relative Locations of Spatial Textures

The final method used for this portion of the research was to use the relative locations of spatial textures such as tire-like objects for the classes. The centroid Euclidean metric was used to create the feature vectors for each class. The training and test files were created from correlation data obtained from correlating each tank with a drive sprocket from a representative tank and from correlating each truck with a tire from a representative truck. Once the correlation files were created, the top five maxima and their locations were used as inputs to the centroid Euclidean distance estimator. The resultant feature vectors were then fed to a standard backprop classifier with one class representing tanks and the other class representing trucks. Figure 34 is a representative sample of the output from the correlator for the truck data which has been correlated with a tire while Figure 35 is representative of the tanks which have been correlated with a cog. The overall performance was better

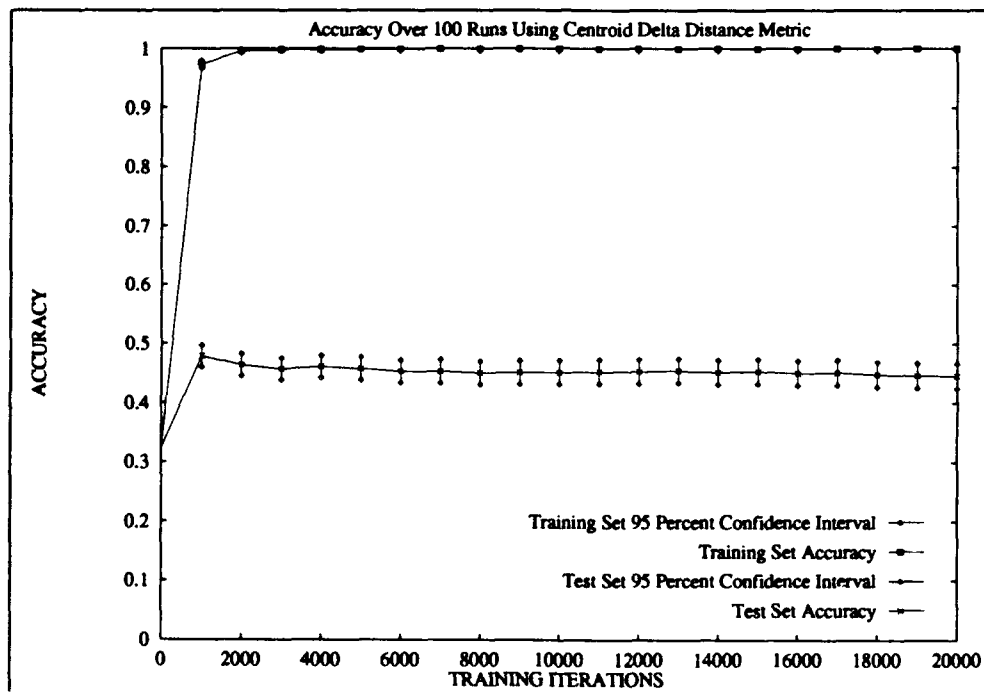


Figure 33. Comparison of Training Accuracy and Test Set Accuracy For Centroid Delta Metric

than that previously obtained. The data was presented to the classifier as a two-class (tank or truck) problem. The results in Figure 36 are considerably better than chance and demonstrate the viability of relative locations of spatial textures as a means of identifying man-made objects.



Figure 34. Result of Correlating a Truck With a Tire From a Truck in the Data Set.

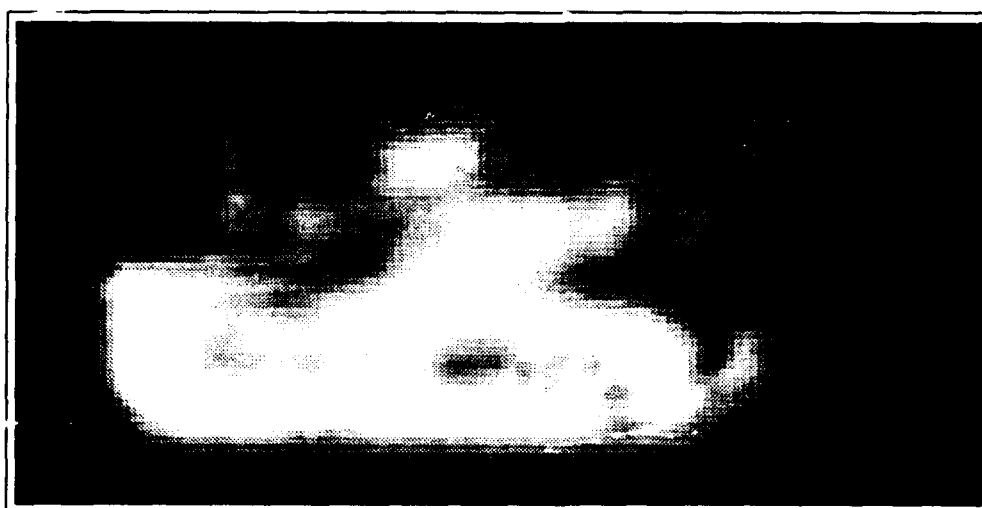


Figure 35. Result of Correlating a Tank with a Cog From a Tank in the Data Set.

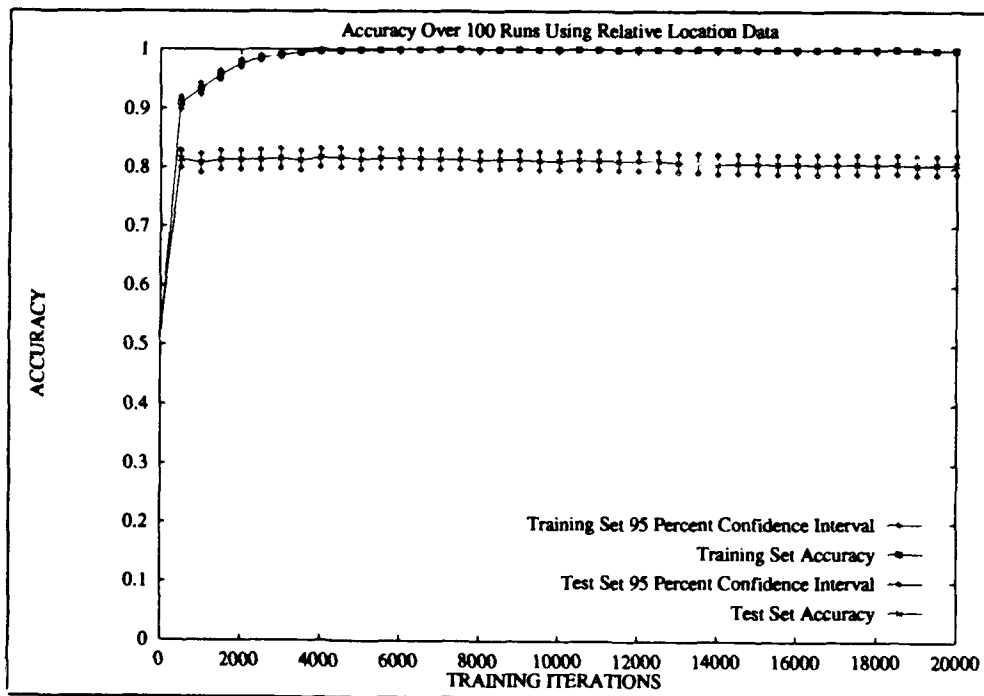


Figure 36. Comparison of Training Accuracy and Test Set Accuracy For Relative Location Data Set

The results in Figure 36 were very encouraging and the research was directed along a path which would allow a machine to find the best relative locations for segmentation and subsequent classification. The segmentation problem is non-trivial and a considerable amount of research was conducted in an attempt to make the concept work. The research centered around using a Gabor filter template of the desired correlation object, such as the tire for a truck, which would produce a similar result as that obtained with correlation. The template is a feature vector made up of Gabor filter coefficients of the object to be correlated with the input scene. The output from a roving window in the input image is processed using the Gabor filter outputs as features. The roving window feature vector is then correlated with the stored template and a correlation image is produced for all shifts of the roving window. The results of this correlation process, while easy enough for a human to assimilate, are not good enough for machines. An example of a truck with the resulting correlation picture is given in Figure 37. Depending upon the number of cycles/window, Gaussian roll-off, filters and orientations very different results could be obtained. The ability of machines to use Gabor coefficient maxima and their location is imprecise with approximately 60-70 percent accuracy possible.

The failure of the Gabor maxima approach to obtain adequate classification accuracy led the research to the relative location idea. The results obtained in Figure 36 were due in large measure to user inputs as to which peaks in the correlation images, such as Figure 35, were the brightest. The large amount of user input into the generation of the relative location made it difficult to transition the results to an autonomous process. The relative location results presented in this chapter were encouraging and the research was directed at making the relative location process as autonomous as possible. Chapter 5 discusses how the final relative location architecture was implemented and presents the results for the three class problem presented in this chapter as well as a seven class problem. Because of the large number of features generated using the Gabor filter coefficient approach a method of determining which features were important and which were useless was needed. The next chapter discusses feature saliency from a Bayesian perspective.

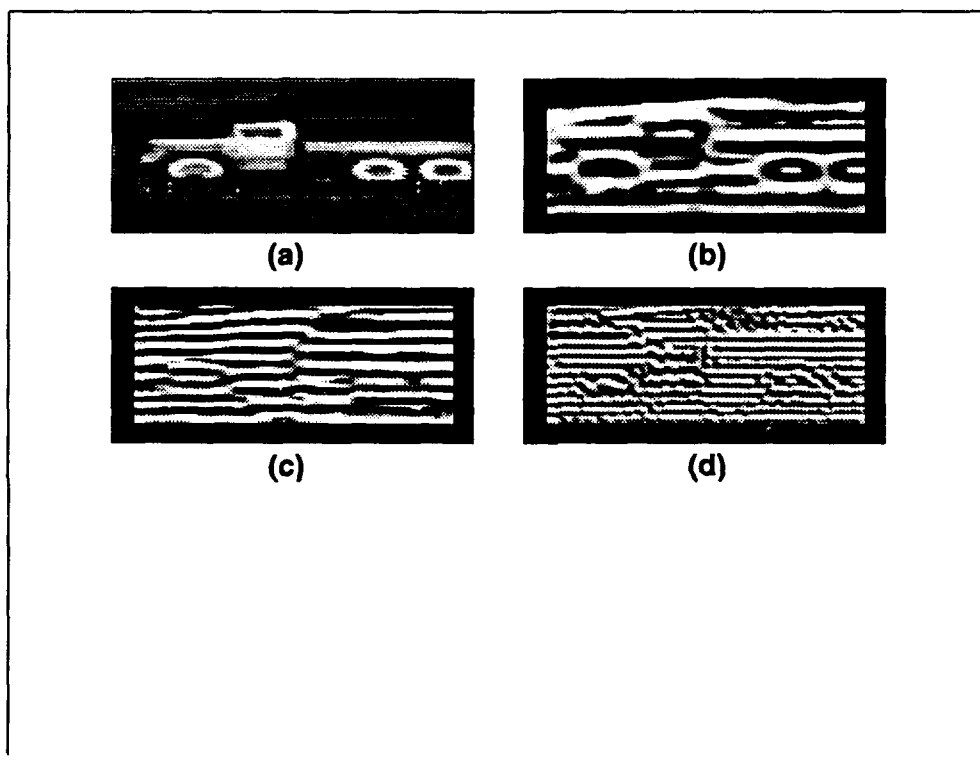


Figure 37. Output Images obtained from Correlating Gabor Templates with Original Image. (a) Original Image. (b) Correlation Result for 4 Filters, 2 cycles/window, $\sigma = 0.3$. (c) Correlation Result for 4 Filters, 3 Cycles/Window, $\sigma = 0.3$. (d) Correlation Result for 4 Filters, 4 Cycles/Window, $\sigma = 0.3$.

IV. Searching the Input Space for Optimal Features

The last chapter described the gestalt neural network architecture which is implemented in this research. This chapter describes the path taken in the quest for the optimal feature set. I have heard it said many times by my thesis committee: "*find a good feature set and I'll give you a good classifier*". The solution to this dilemma is not easy to find and is as elusive as the snail darter. This chapter describes the methods used to solve this issue of what makes a good feature as well as introduce a foundation for using neural networks to find the appropriate features for the classification task.

4.1 The Curse of Dimensionality

One of the easiest traps to fall into when performing pattern recognition is the trap of taking too many measurements. This is known as the curse of dimensionality. Duda and Hart (8:136) point out that the increase in computation is not sufficient enough to justify the gain in classification accuracy. Not only that, poor measurements can skew the actual results by introducing confusing and overlapping data sets. This leads us to the question of how do you select the best features without first trying them out? The answer is that you can't just pick out the optimal features. You either must use clustering algorithms such as the Karhunen-Loève transform (8:246) (53:269) to find important features or test for the salient features after the classifier has been trained (36, 44). The next section describes how feature saliency can be determined from a Bayesian perspective.

4.2 Saliency of Features

Saliency can be defined as a measure of a feature's ability to influence the classifier. If the feature, when varied, has little effect on the decision of the classifier then it is of little use and will have a low saliency value. On the other hand, when minor perturbations of a feature have a dramatic effect on the output of the classifier, then the feature is highly desirable and has a high saliency value. The question is to come up with a methodology

for determining the saliency of a feature. Ruck (43) proposed a method for computing the saliency of a feature on the input to a feedforward neural network. This section will establish the same measurement from a statistical point of view. Ruck (45) also showed that when the sample data accurately represents the underlying probability density functions and the output nodes are trained to be 1 and -1 respectively, the final state of the output nodes for a feedforward neural network actually represent the *a posteriori* probabilities of a given class, j , given an input feature vector, \vec{x} . This is shown below:

$$z_j = P(C_j|\vec{x}) \quad (16)$$

where

$z_j \equiv$ output of node j on output layer

$P(C_j|\vec{x}) \equiv$ the *a posteriori* probability of class C_j given input vector \vec{x}

$\vec{x} \equiv$ input feature vector

A relationship between the *a posteriori* probability and an error criterion which will be used to measure the saliency of a given input feature is now developed. From Bayesian statistics we know that (32:37):

$$\sum_j P(C_j|\vec{x}) = 1$$

The probability of error for a particular output node can be defined as the likelihood that the input feature vector belongs to a class other than that indicated by the value of that particular output node. The probability of error for an output node (j) given an input vector \vec{x} can be computed as:

$$P_{error}(j, \vec{x}) = 1 - P(C_j | \vec{x})$$

where

$P_{error}(j, \vec{x}) \equiv$ the likelihood that the input feature vector \vec{x} belongs to a class other than that indicated by

or in equivalent form:

$$P_{error}(j, \vec{x}) = \sum_{k \neq j} P(C_k | \vec{x}) \quad (17)$$

Based upon (16) and (17), when we take the partial derivative of the summed outputs $\sum_{k \neq j} z_k$ of the feedforward network we are in actuality taking the partial derivative of $P_{error}(j, \vec{x})$. By taking the partial derivative of the summed outputs $\sum_{k \neq j} z_k$ with respect to an input feature we can measure the sensitivity of the output to the input feature. This provides a meaningful metric for feature saliency. Now take the derivative of the probability of error, P_{error} , with respect to a given input feature, x_i , to measure its saliency.

$$\begin{aligned} \frac{\partial P_{error}(j, \vec{x})}{\partial x_i} &= \frac{\partial}{\partial x_i} \sum_{k \neq j} P(C_k | \vec{x}) \\ &= \frac{\partial}{\partial x_i} \sum_{k \neq j} z_k \end{aligned} \quad (18)$$

The output of a three layer neural network as depicted in Figure 38 is typically represented by (46:329):

$$z_j = f_h\left(\sum_m x_m^2 \cdot w_{mj}^3 + \theta_j^3\right) \quad (19)$$

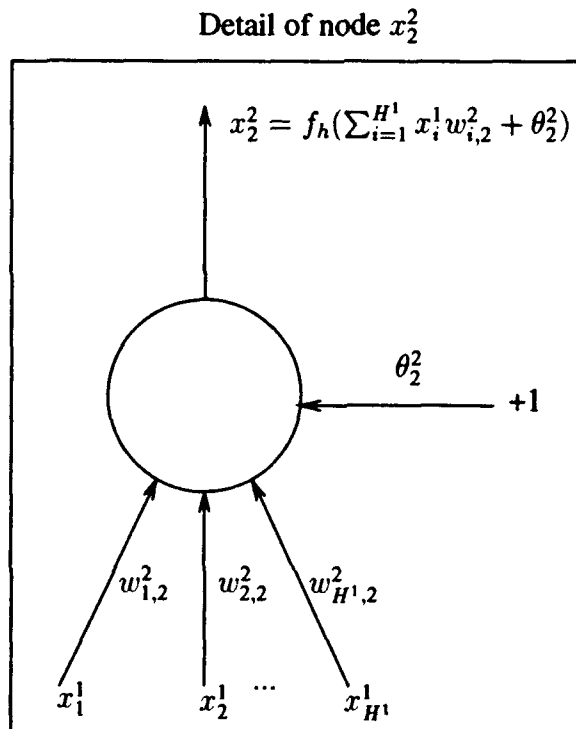
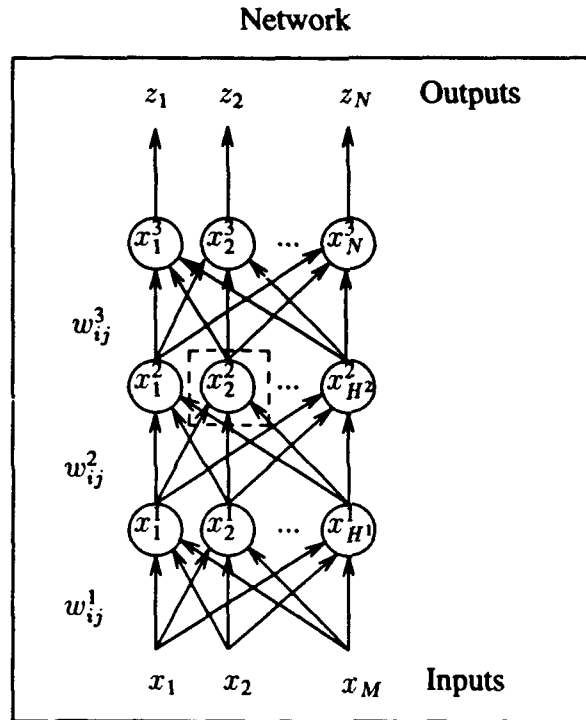


Figure 38. Three-Layer Feedforward Network With Accompanying Expansion of a Hidden Node (44)

where

$z_j \equiv$ output of node j on the third layer

$w_{mj}^3 \equiv$ weight value from node m on second layer to node j on third layer

$x_m^2 \equiv$ output of node m on second layer

$\theta_j^3 \equiv$ threshold for node j on layer 3

$f_h \equiv$ sigmoid function

Now substitute for z_k by inserting (19) into (18) we obtain

$$\frac{\partial P_{error}(j, \vec{x})}{\partial x_i} = \frac{\partial}{\partial x_i} \sum_{k \neq j} f_h \left(\sum_m x_m^2 \cdot w_{mk}^3 + \theta_j^3 \right)$$

Using the fact that the derivative of the sigmoid function, f_h , is simply the sigmoid times one minus the sigmoid leads to

$$\frac{\partial P_{error}(j, \vec{x})}{\partial x_i} = \sum_{k \neq j} z_k \cdot (1 - z_k) \cdot \frac{\partial}{\partial x_i} \left(\sum_m x_m^2 \cdot w_{mk}^3 + \theta_j^3 \right) \quad (20)$$

where

$z_k \equiv$ output of node k on layer 3

$x_i \equiv$ feature i on input layer

Applying the derivative yields

$$\frac{\partial P_{error}(j, \vec{x})}{\partial x_i} = \sum_{k \neq j} \delta_k^3 \cdot \sum_m w_{mk}^3 \cdot \frac{\partial}{\partial x_i} (x_m^2)$$

where

$$\delta_k^3 = z_k \cdot (1 - z_k)$$

Continuing the process yields:

$$\begin{aligned} \frac{\partial P_{error}(j, \vec{x})}{\partial x_i} &= \sum_{k \neq j} \delta_k^3 \cdot \sum_m w_{mk}^3 \cdot \delta_m^2 \cdot \frac{\partial}{\partial x_i} \left(\sum_n x_n^1 \cdot w_{nm}^2 + \theta_m^2 \right) \\ &= \sum_{k \neq j} \delta_k^3 \cdot \sum_m w_{mk}^3 \cdot \delta_m^2 \cdot \sum_n w_{nm}^2 \cdot \delta_n^1 \cdot w_{in}^1 \end{aligned}$$

where

$$\delta_m^2 = x_m^2 \cdot (1 - x_m^2)$$

$$\delta_n^1 = x_n^1 \cdot (1 - x_n^1)$$

Note that the derivative of an output node (z_j) of a feedforward network with respect to a given input node (x_i) can be generalized to any number of layers as shown below:

$$\frac{\partial z_j}{\partial x_i} = \delta_j^N \cdot \sum_m w_{mj}^N \delta_m^{N-1} \sum_n w_{nm}^{N-1} \delta_n^{N-2} \dots \sum_p w_{po}^3 \delta_p^2 \sum_q w_{qp}^2 \delta_q^1 w_{iq}^1 \quad (21)$$

Also note that if the feedforward network uses linear transfer functions on the output nodes rather than sigmoidal transfer functions then $\delta_j^N = 1$.

Now that a relationship has been established between an input feature and the probability of error, we can define a Bayesian-based saliency metric Ω_i for feature x_i of the input feature vector \vec{x} . The saliency metric presented is based on the magnitude of the changes in P_{error} as the feature x_i is varied over its range of values.

Because we are measuring the sensitivity of the P_{error} to changes in x_i , a feature which causes little or no change in P_{error} is of little relevance while a feature which causes substantial changes in P_{error} is very relevant and should be retained.

The saliency (Ω_i) can be calculated as follows: Take each training vector (\vec{x}) in the training set S. For the i th feature, sample at various locations over the expected range of the feature while holding all other features in \vec{x} constant. Now compute the magnitude of the partial derivative of the output z_k for the sample values. Sum the magnitude of the partial derivative of P_{error} over the outputs (j), the sampled values of x_i (D_i), and the training set S. See (44) for a discussion of this method of forming Ω_i .

$$\Omega_i = \sum_j \sum_{\vec{x} \in S} \sum_{x_i \in D_i} \left| \frac{\partial P_{error}(j, \vec{x})}{\partial x_i} \right|$$

which can be rewritten as:

$$\Omega_i = \sum_j \sum_{\vec{x} \in S} \sum_{x_i \in D_i} \left| \sum_{k \neq j} \frac{\partial z_k}{\partial x_i} \right| \quad (22)$$

where

$D_i =$ Set of sample points for feature x_i

The variable γ_i , which is the absolute value term in Eq (22) is introduced.

$$\gamma_i = \left| \sum_{k \neq j} \frac{\partial z_k}{\partial x_i} \right| \quad (23)$$

Now show that γ_i is bounded.

Because z_k is a sigmoid function, each $\frac{\partial z_k}{\partial x_i}$ exists and is finite. Therefore

$$\sum_{k \neq j} \left| \frac{\partial z_k}{\partial x_i} \right| \leq C \quad (24)$$

and by the triangle inequality (26:70) γ_i is bounded by Eq (24)

$$\gamma_i \leq \sum_{k \neq j} \left| \frac{\partial z_k}{\partial x_i} \right| \leq C \quad (25)$$

Combining Eqns (22 - 25) yields

$$\Omega_i = \sum_j \sum_{\bar{x} \in S} \sum_{x_i \in D_i} \left| \sum_{k \neq j} \frac{\partial z_k}{\partial x_i} \right| \leq \sum_j \sum_{\bar{x} \in S} \sum_{x_i \in D_i} \sum_{k \neq j} \left| \frac{\partial z_k}{\partial x_i} \right| \quad (26)$$

Rearranging the order of summation and choosing the right hand side, yields the metric Λ_i which is a simplification of (22).

$$\Lambda_i = \sum_j \sum_{\bar{x} \in S} \sum_{k \neq j} \sum_{x_i \in D_i} \left| \frac{\partial z_k}{\partial x_i} \right| \quad (27)$$

The proposed metric, Λ_i is more sensitive to changes in the P_{error} as x_i is varied than the Ω_i .

The saliency metric can be readily calculated by substituting the result from (21) into (27). The required weights and outputs are available in the normal feedforward algorithm and the required storage for intermediate results is quickly inserted into the feedforward algorithm.

4.3 Comparison With A Previous Saliency Metric

Now that we've established saliency from a probability perspective, how does it compare with a previously used saliency metric? Ruck (44) proposed a saliency metric which is given below:

$$\tilde{\Lambda}_i = \sum_{\bar{x} \in \mathbf{S}} \sum_k \sum_{x_i \in D_i} \left| \frac{\partial z_k}{\partial x_i} \right| \quad (28)$$

Note the similarity between the Ruck saliency metric and the metric defined in (27). Ruck's metric is actually the same as that obtained from the probability of error derivation to within a factor of $n-1$ where n is the number of output nodes for the feedforward network. To illustrate this relationship, consider the following:

$$\begin{aligned} \Lambda_i &= \sum_j \sum_{\bar{x} \in \mathbf{S}} \sum_{k \neq j} \sum_{x_i \in D_i} \left| \frac{\partial z_k}{\partial x_i} \right| \\ &= \sum_{k \neq 1} \sum_{\bar{x} \in \mathbf{S}} \sum_{x_i \in D_i} \left| \frac{\partial z_k}{\partial x_i} \right| + \sum_{k \neq 2} \sum_{\bar{x} \in \mathbf{S}} \sum_{x_i \in D_i} \left| \frac{\partial z_k}{\partial x_i} \right| + \dots + \sum_{k \neq n} \sum_{\bar{x} \in \mathbf{S}} \sum_{x_i \in D_i} \left| \frac{\partial z_k}{\partial x_i} \right| \\ &= (n-1) \sum_{\bar{x} \in \mathbf{S}} \sum_k \sum_{x_i \in D_i} \left| \frac{\partial z_k}{\partial x_i} \right| \\ &= (n-1) \cdot \tilde{\Lambda}_i \end{aligned}$$

The results given in this chapter show that the Ruck saliency metric, while chosen intuitively, was actually a metric which measured the sensitivity of the probability of error to a given input feature. With the foundation of the saliency metric developed from a Bayesian point of view, it can be used to determine meaningful features.

4.4 Application of the Saliency Metric For Feature Selection

The previous section developed the concept of saliency from a Bayesian perspective. This section uses the saliency measure to reduce the dimensionality of the input feature vectors to a neural network. With this concept firmly in hand we will look at how the metric classifies useful features for the three class problem given in Figure 39.

The initial feature vector data set was generated by choosing an (8x8) window with a two cycles per window cosine Gabor function at angles of (0, 15, 30, ... 150, 165) degrees.

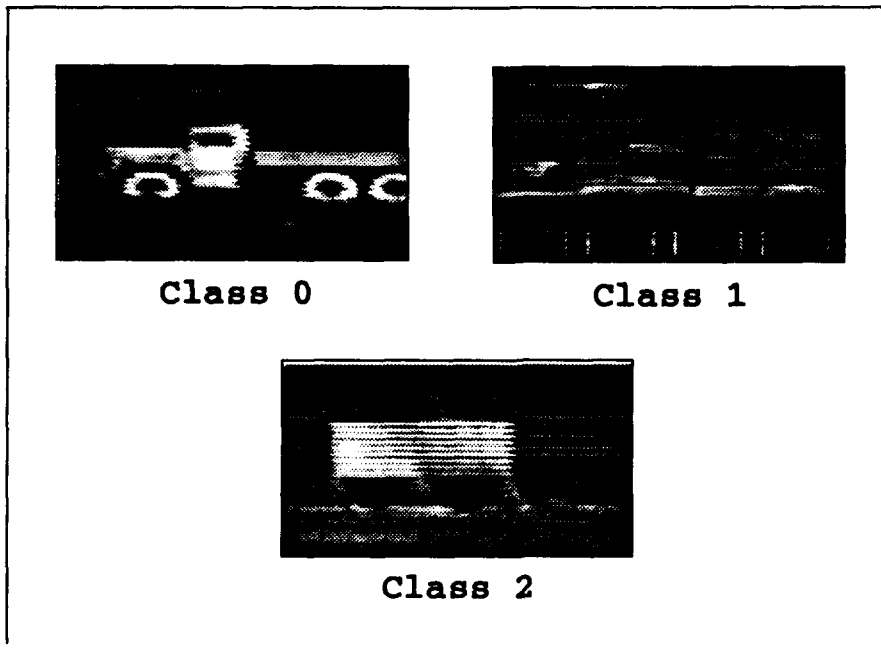


Figure 39. Three Class Problem For Classification

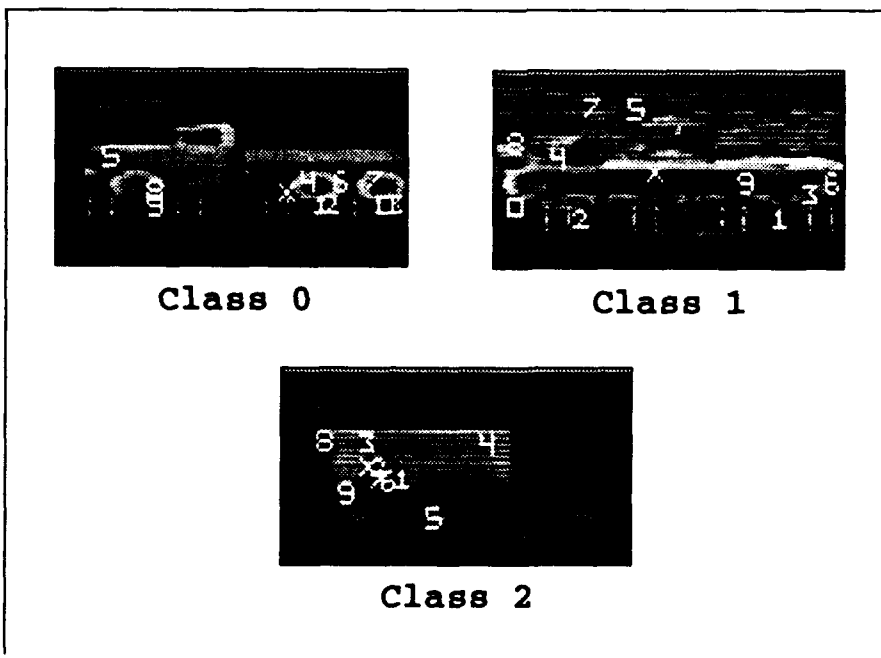


Figure 40. Three Class Problem Overlaid With Top Ten Gabor Locations

Once these windows are mapped over the input a composite image is formed according to the rule below:

$$\begin{aligned} \text{if } |Gabor[k][i][j]| &> |Composite[i][j]| \\ Composite[i][j] &= Gabor[k][i][j] \end{aligned} \quad (29)$$

where

- $Gabor[k][i][j] \equiv$ Gabor coefficient of Gabor filter k at location i, j
- $Composite[i][j] \equiv$ Composite image coefficient at location i, j
- $k \equiv$ Gabor filter image number (one per filter)
- $i \equiv$ x window location
- $j \equiv$ y window location

Once the composite image is formed the top N (by magnitude) Gabor values and their locations are selected. These N values and their locations form the 3-N features of the feature vector which represent the gestalt of the image. This process is continued for all of the images in the data set. Once these feature vectors are calculated, they are then processed for input to a multilayer perceptron classifier. The final results of this process are given in Figure 40. As can be seen in Figure 40, the location of the maximum Gabor coefficient is marked with an "X" while the next nine locations are marked with a number 0..9. These locations are the reduced representation of the input image.

Once the N locations have been determined then more processing is used to reduce the feature vector even further. In the final result the input feature vector is reduced to nine features. The results shown in Figure 41 are obtained by training 100 separate neural

networks using the nine features as inputs to the neural network classifier.

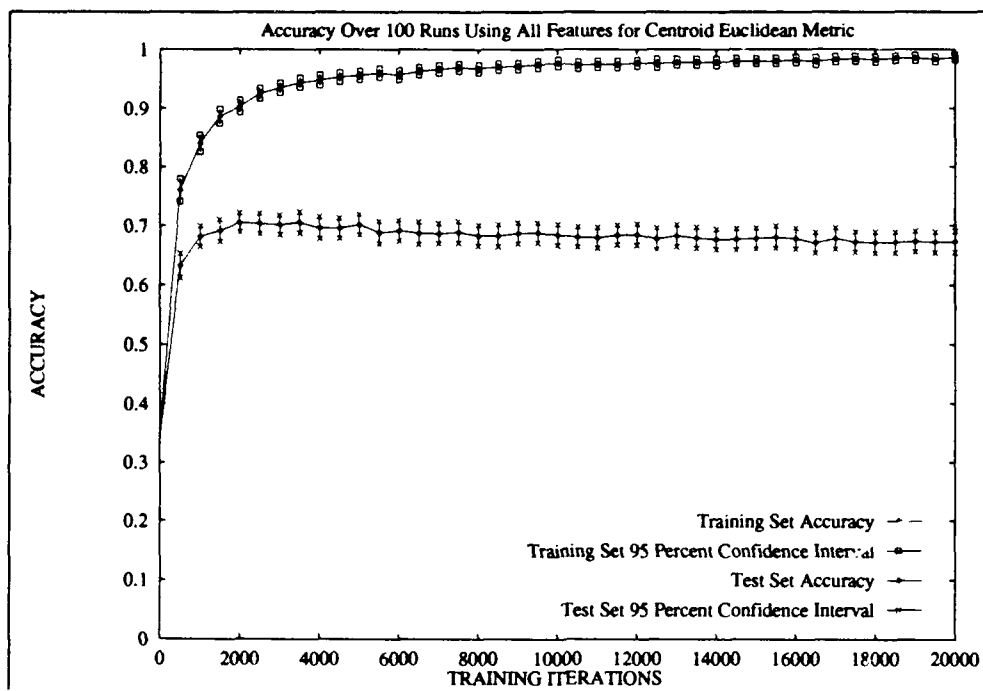


Figure 41. Accuracy Using All Nine Features from Centroid Euclidean Distance Metric Averaged over 100 Runs

The Bayesian saliency metric is then applied for each of the 100 runs and the results are recorded in Table 4. As can be seen in Table 4, a histogram of each feature is formed. For instance, feature 0 was the most important 75 times, second most important 19 times, third four times and so on. After the histogram is formed the features are ranked using the formula

$$\text{rank} = \sum_{n=1}^N K_n \cdot (N - n - 1) \quad (30)$$

where

$n \equiv$ Importance of Feature

$N \equiv$ Total Number of Features

$K_n \equiv$ Number of times feature had importance n

The saliency computed using Eq (30) is given in Table 5

Once the top three features for the centroid Euclidean metric were chosen, a new run using 100 separate neural networks for training was conducted. Figure 42 shows virtually identical results as those obtained using all of the features. The Bayesian saliency metric has helped eliminate features which were ambiguous and detrimental to finding a good solution space for the neural network classifier. In addition, by reducing the number of features required to solve the pattern recognition problem computational time to train the neural networks is reduced and the likelihood of violating Cover's (4) or Foley's (11) rules is greatly diminished.

In this chapter a new method based upon the Bayesian properties of a feedforward neural network was used to determine the importance of features fed to a neural network. The new method was found to be compatible with the saliency metric previously used by Ruck (44). The consistency of the features chosen by the Ruck saliency metric was previously demonstrated by Ruck (43) for backpropagation, k-nearest neighbor, and Parzen window classifiers. As a result of the application of the new metric, virtually identical results were found between using all of the features and only the top three features in a classification problem. The new saliency metric further supports the proofs provided by Ruck (45) and Gish (15) that the feedforward neural network is performing as an optimal Bayesian classifier. The next chapter expands upon the idea of reducing the number of features by utilizing the relative locations of spatial textures to form a greatly reduced set of features to classify FLIR imagery.

Importance	9th	8th	7th	6th	5th	4th	3rd	2nd	1st
Feature 0	0	0	0	0	0	2	4	19	75
Feature 1	15	17	13	8	12	20	9	5	1
Feature 2	4	4	8	4	19	20	24	15	2
Feature 3	24	9	11	23	15	10	7	0	1
Feature 4	16	24	12	19	8	11	8	2	0
Feature 5	24	17	23	15	10	9	0	2	0
Feature 6	14	18	19	10	21	6	10	2	0
Feature 7	2	6	13	16	8	11	19	24	1
Feature 8	1	5	1	5	7	11	19	31	20

Table 4. Importance of Each Centroid Euclidean Feature Over 100 Runs. A histogram is given of how many times the given feature was ranked in importance.

Feature	Saliency Value
Feature 0	767
Feature 1	312
Feature 2	473
Feature 3	260
Feature 4	254
Feature 5	207
Feature 6	274
Feature 7	457
Feature 8	596

Table 5. Saliency of Each Centroid Euclidean Feature over 100 runs. Features 0, 2, and 8 are the most salient.

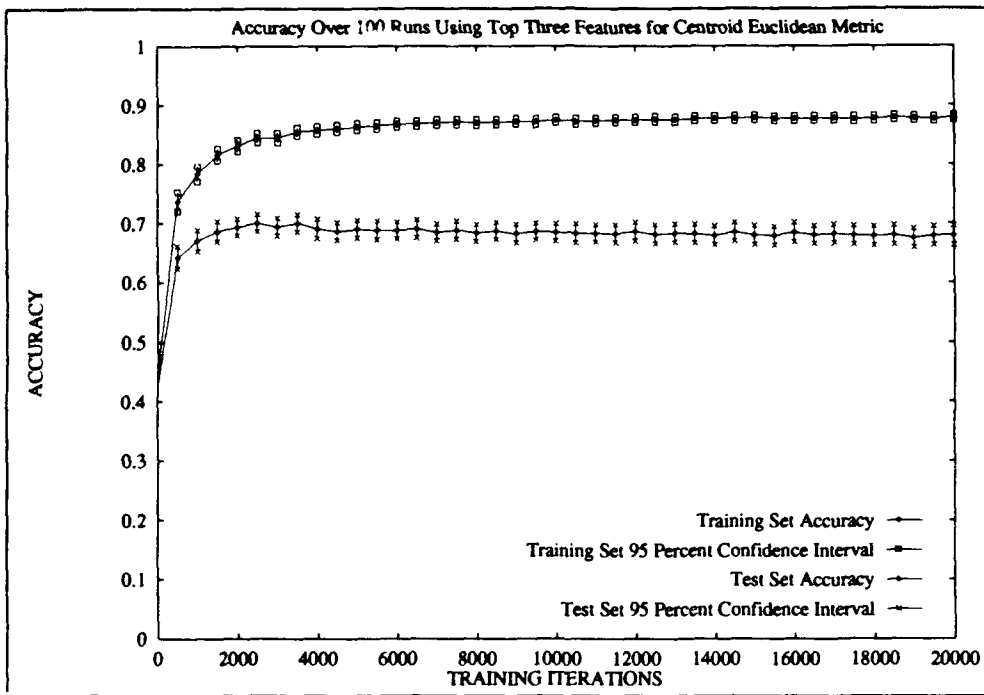


Figure 42. Accuracy Using Top Three Features from Centroid Euclidean Distance Metric Averaged over 100 Runs

V. Relative Locations and Their Importance in Object Recognition

During the course of the research, it became apparent that the roving window architecture would not yield the desired classification accuracy. The major drawback to the roving window architecture with its fixed window locations was that the objects of interest were not consistently in the same place in the scene. This observation led the research in a direction which could overcome the limitations due to the fixed data locations for the roving window architecture. The process described in this chapter overcomes the problems encountered in the roving window architecture and is able to extract features in a consistent manner. The heart of the new system lies in the concept of using the relative locations of non-homogeneous regions to solve the feature extraction, and classification problems nearly simultaneously. One of the concepts which was always of interest during the course of the research was that of using the relative locations of distinguishing features or characteristics to aid in the classification task. Indeed, many of the qualified successes found with the Gabor front-end were due to using various distance metrics from the centroid of the Gabor orientation maxima to each maxima's location. This concept was very useful but had severe limitations due to the fixed measurement locations for the roving window. Thus a better and more consistent method of data retrieval was required for the feature extraction and classification tasks. The relative location concept allows the classifier to retain the global characteristics of the image in question while extracting specific information from localized regions within the image. The architecture presented in this chapter is based upon the relative location concept for object recognition. The architecture uses an operator's input to define what features in an image are important for classification. Perhaps for a tank the tracks, turret and barrel are important. The operator would simply use a mouse to select the areas of the tank to be used for the classification task. The algorithm then stores each of the desired regions and their relative locations with respect to each other for feature extraction and subsequent classification. The concept

of using relative locations for data extraction and classification is introduced in the next section.

5.1 *The Concept of Relative Locations*

As mentioned previously, the results of the research pointed to the relative locations of distinguishing features or characteristics as a good point to improve system performance. This hypothesis is based on the research done by Luria (30) on the perception of objects and eye movement. Luria asserts that the brain makes a "perceptual hypothesis" based upon key signs (features) of an object. This is accomplished in the human visual system via saccadic eye movement with subsequent processing in the brain. A good example of the saccadic process can be seen in Figure 43. As can be seen in the Figure, the eye movements seem to be clustered about the eyes and the mouth with some movement around the perimeter of the face.

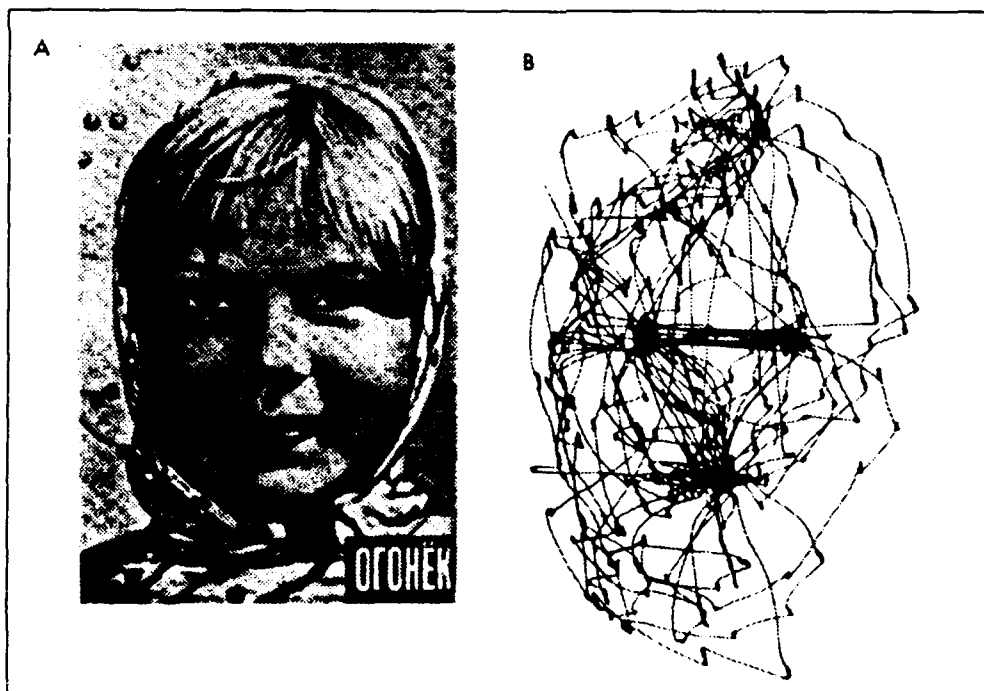


Figure 43. Eye Scanning Pattern For Girl Used by Luria (30:135)

The relative locations of the eyes and nose and their particular characteristics are extremely important for face recognition (25). The picture of the girl with the corresponding saccadic eye movement shows how the eye position is used by the brain to process the position of the eyes and mouth in developing a recognition model as pointed out by Luria (30). The concept of relative locations of distinguishing spatial features is implemented using the system architecture presented in the next section.

5.2 The Relative Location System Architecture

The system architecture presented in this section uses a combination of templates which contain relative locations of distinguishing characteristics to preprocess the data fed to a feedforward neural network for training. In addition, a user interface was developed to assist in the task of identifying which characteristics are important in performing the recognition task. A block diagram of the relative location architecture is given in Figure 44. The architecture consists of a series of independent templates along with their associated neural network classifier. The image is first presented to the system. Each template is used to find the best match in the scene and extract data from the relative locations of selected windows. The output of the feature extractor is then fed to the neural network associated with the template. Each neural network has been trained to recognize objects that are of the same type as the template and to reject all others. Once all of the templates have been applied to the input image, the class is declared to be the class associated with the neural network with the largest in-class output. With this in mind the NeuralGraphics (49) computer code was modified to incorporate the relative location concept.

To illustrate the concept consider Figure 45 which is a snapshot of the revised NeuralGraphics interface. The user uses a mouse to select the desired portions of the image in the "Template Creation" window. Once the user is finished selecting the relative location windows, they stored as a template in the "Template Storage" window. The process is continued until all of the desired templates are created. The user stores the template(s) by selecting the proper menu option with the mouse. Once the desired number

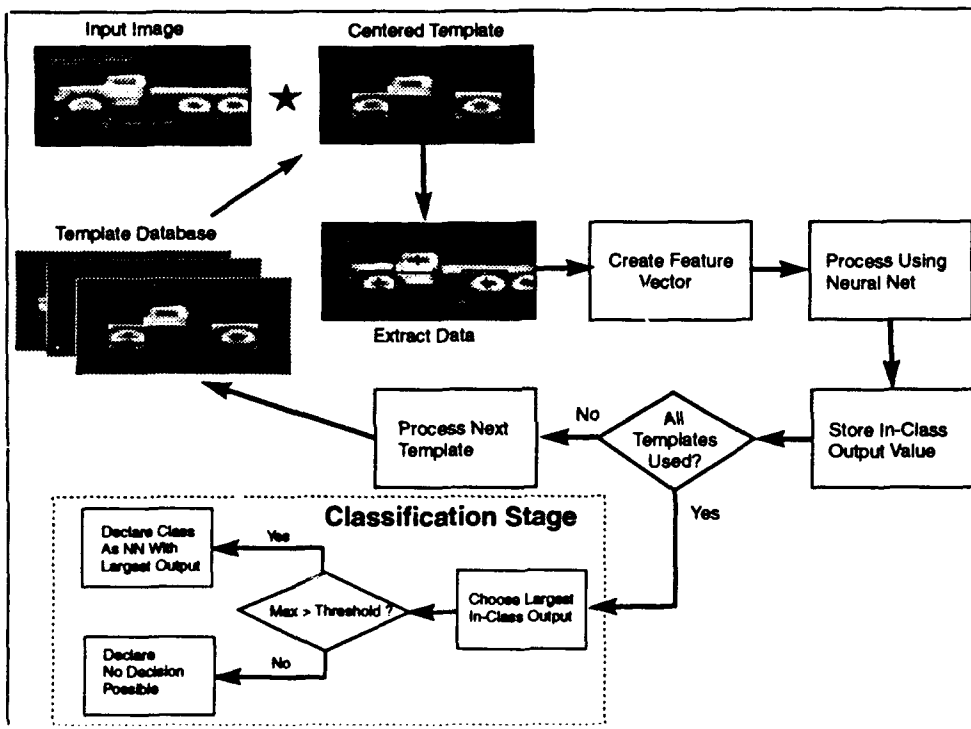


Figure 44. Neural Network Based Automatic Target Recognizer Using Relative Locations

of templates are selected and stored, the templates are used to create feature vectors for subsequent training of a neural network. The feature vectors are created by first centering the template as shown in the "Centered" window in Figure 45. The centered template is correlated with the input image and the result is stored in the "Correlated" window in Figure 45. This correlation uses the global characteristics of the template to find the best match for the template in the scene. Once the best match is found, the centroid of the template is placed at the correlation peak and data is extracted from the input image using the relative location windows defined in the template. This process allows the local variations within each window to be used in the modeling process as well. The relative location data is extracted from the template in the "Template Rel Locations" window and from the image in the "Image Rel Locations" window. One of the relative location windows for the template is shown in the "Template Var Window" with its corresponding window in the image shown in the "Image Var Window". The windows are processed to create the feature vectors for training and classification of the input image. One of the benefits of this architecture is that new classes can be added without requiring extensive retraining because each neural network classifier is solving an in-class/out-of-class problem. Hence only the new neural network representing the new class needs to be trained over the entire database. To further illustrate how the system works, each portion of the system architecture will be presented in detail.

5.3 Template Creation

The user defined template is crucial in obtaining a consistent location for extracting data from scenes in the database. The selection process is depicted in Figure 46. The user can select either the entire scene as a template or up to twenty variably-sized windows from the scene for the template. When the user is finished creating the desired template, the data associated with each window (xsize, ysize, delta_x, delta_y, xmean, ymean) are stored along with all of the values in the template.

While the word description of the template creation process is easy to understand,

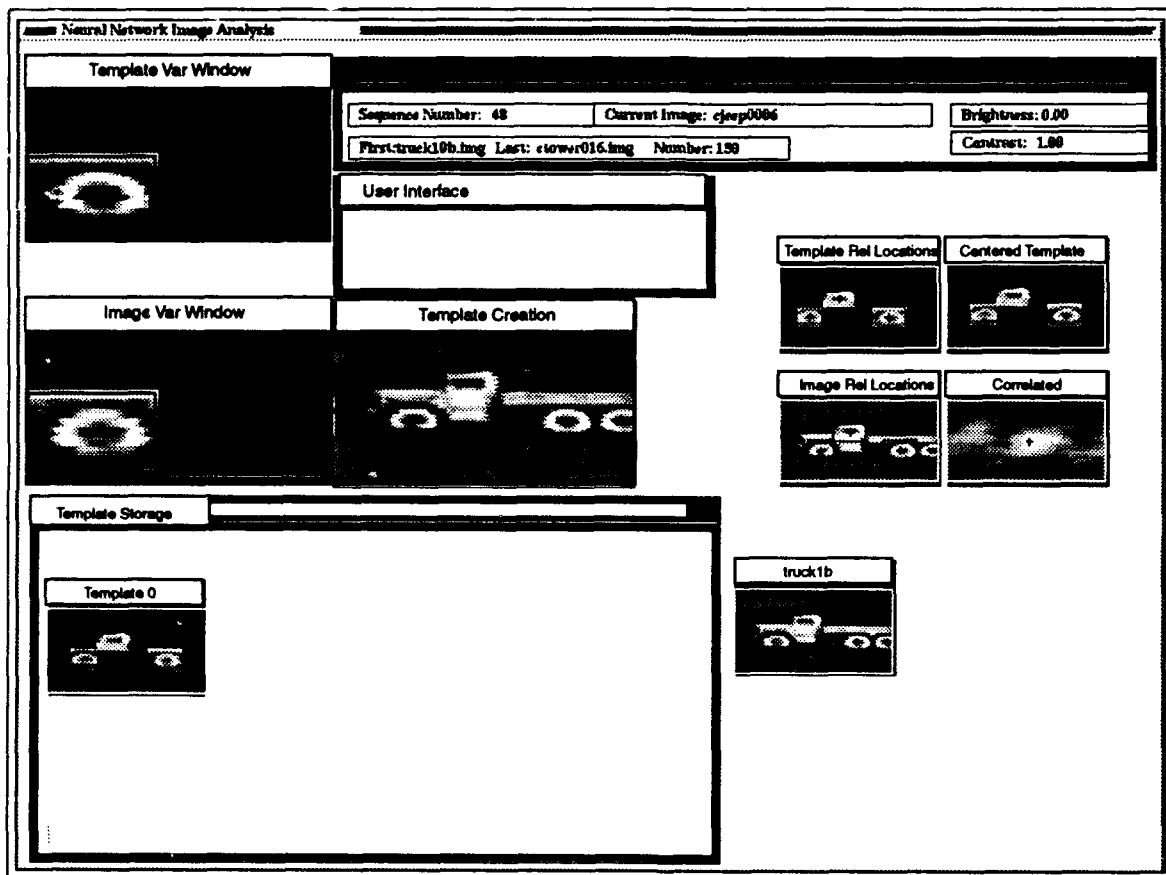


Figure 45. The Relative Location Screen running within NeuralGraphics. The various windows are used to extract desired features, find objects in the image and process relative location windows.

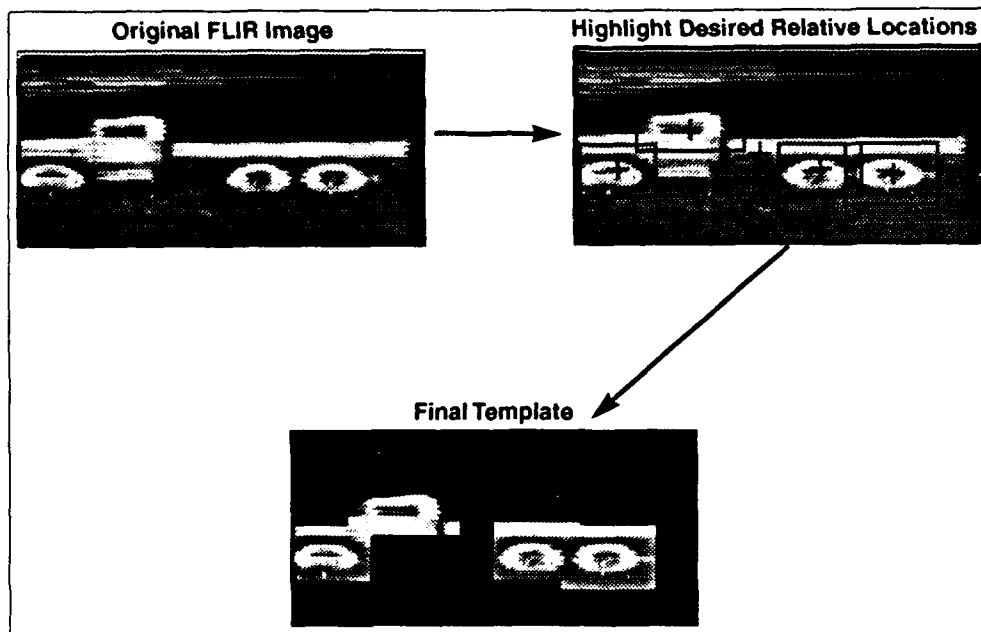


Figure 46. Creation of a Template Via Highlighting The Desired Portions of The Input Image and Storing Only The Highlighted Portions As The Template.

there is a considerable amount of processing which is going on as the template is created. As each new window in the image is selected for inclusion in the template, several parameters are calculated and stored. The first parameter calculated in the relative location process is the centroid (x_mean , y_mean) of the ensemble of variable windows selected for inclusion in the template. Next comes the δ_x and δ_y values from the centroid to each variable window in the template. The size of each variable window is stored as well. After the template is stored, it is used as a matched filter to find the best match in the input image for the template. The next section describes how the template is used for registering on the best match in the input image and then how the centroid is used to insure the best fit between the image and the template.

5.4 *Template Registration*

Once the template is created and stored, it is used for processing the input image as well as finding the desired object in the scene of interest. The process is best described by viewing Figure 47. The input image and the centered template are correlated together with the result displayed in the correlation window. The location of the maximum value in the correlation plane represents the best match between the template and the input image. The maximum in the correlation plane is then used as the registration point for the data extraction process. The data is then extracted from variably sized windows from the input image based upon the Δx and Δy locations from the maximum. It should be noted here that if the input image is the reverse of the template, black-on-white versus white-on-black, that the minimum in the correlation plane would be the best match. This is a definite problem with FLIR imagery because of crepuscular conditions in dawn and dusk viewing situations where objects such as tanks may be cooling or heating rapidly while the background has not changed temperature appreciably. This condition is often termed the diurnal shift and is depicted in Figure 48. To overcome this problem for the purposes of data extraction, the location of correlation peak for the template representing the class for each image in the database was stored and recalled. Using this technique, the images could be mixed at will and the features extracted without considering whether the maximum or minimum location was required for the correct registration. Once the location is obtained, the data is extracted from the input image by centering the centroid of the template at the max/min location and then extracting data from the scene based upon the stored Δx and Δy locations for each variable window as depicted in Figure 47. After the data is extracted from the image, the data within each window is processed and the results are stored as a feature vector. The feature vector creation process discussed in the next chapter processes the data extracted from both the scene and the input image.

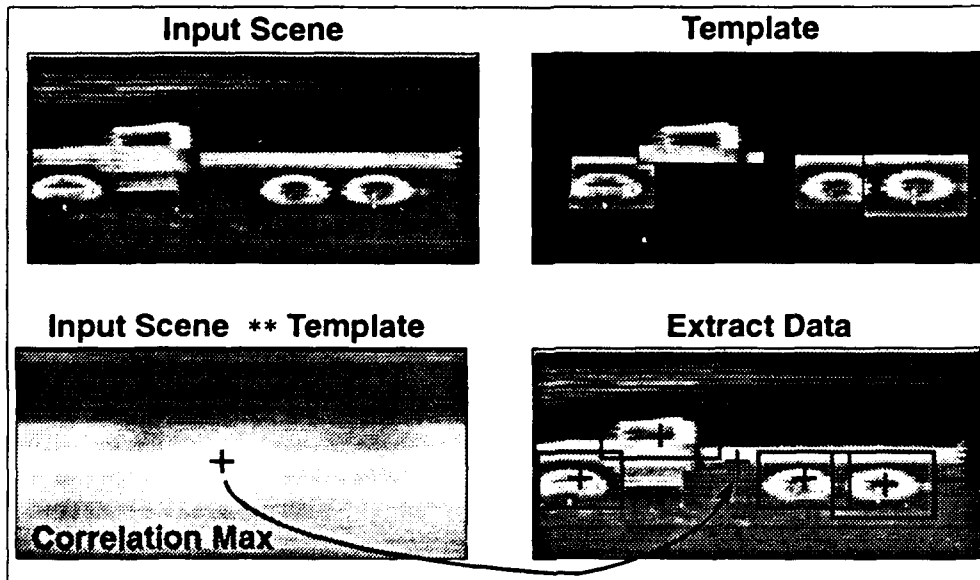


Figure 47. Using the Correlation of the Template and the Input Image for Registration for Subsequent Processing

5.5 Feature Vector Creation

The feature vectors created by the relative location architecture are based upon processing each template window and its corresponding window in the image together. Initially, the magnitude of the correlation between each window pair was used to create features for subsequent inclusion in a feature vector. After studying the problem in further detail, it became obvious that the correlation concept was limited because of diurnal shifts and crepuscular problems. The processing was subsequently changed to allow for the variation in scene contrast. The second method of calculating the features was to perform an inner product of the magnitude of each Fourier coefficient from a discrete Fourier transform (DFT) of each window pair with the dc component removed from each DFT calculation before performing the modified dot product.

Initially the feature vectors were created using the process described in Eq (31).

$$z_k = \max_{ij} |f^k \star g^k| \quad (31)$$

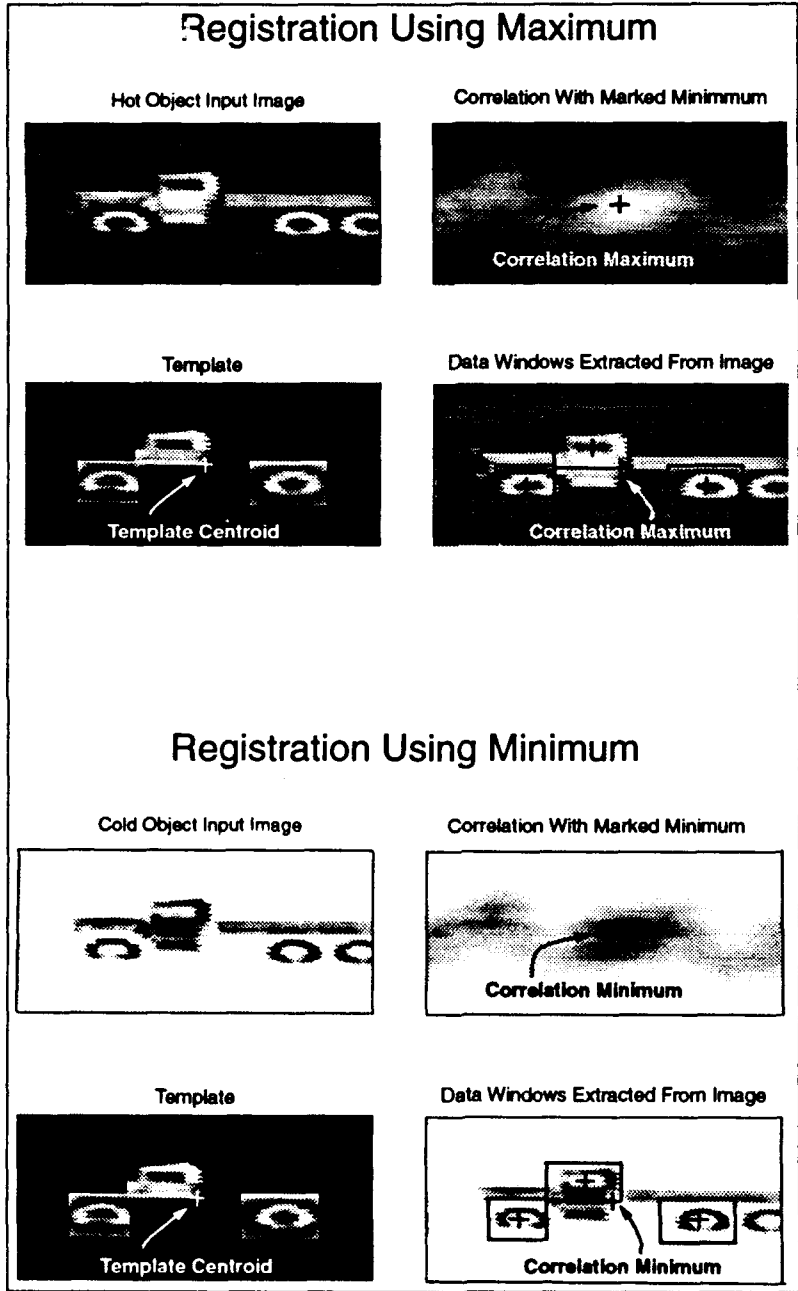


Figure 48. Two Separate Representations of The Same Target With Diurnal Differences With Associated Data Extraction.

where

$z_k \equiv$ feature value for window k

$f^k \equiv$ image window k

$g^k \equiv$ template window k

$i, j \equiv$ pixel location within the window

Because of the diurnal problem a better method of extracting features from the variable windows was developed. The key idea used in the approach to solve the (black-on-white) versus (white-on-black) recognition problem was that the reverse video image can be considered to be the normal image subtracted from a constant background of the maximum pixel value. By observing the Fourier transform of each image a simple solution was obvious. Define the normal image as $n_{x,y}$ and the reverse video image as $r_{x,y}$. As shown in Eq (32), $r_{x,y}$ is simply the result of subtracting $n_{x,y}$ from a constant background equal to the maximum pixel brightness. By taking the Fourier transform of both $n_{x,y}$ and $r_{x,y}$, as shown in Eqs (33) and (34), a simple relationship is found for feature extraction as described mathematically by Eq (35).

$$r_{x,y} = C - n_{x,y} \quad (32)$$

$$\mathcal{F}\{r_{x,y}\} = \mathcal{F}\{C - n_{x,y}\} \quad (33)$$

$$\mathcal{R}(u, v) = C \cdot \delta(0, 0) - \mathcal{N}(u, v) \quad (34)$$

which when the dc component is thrown away reduces to

$$\tilde{\mathcal{R}}(u, v) = -\tilde{\mathcal{N}}(u, v) \quad (35)$$

where

- $C \equiv$ maximum pixel value
- $x \equiv$ pixel position in x direction
- $y \equiv$ pixel position in y direction
- $u \equiv$ pixel position in u spatial frequency direction
- $v \equiv$ pixel position in v spatial frequency direction
- $n_{x,y} \equiv$ normal image
- $r_{x,y} \equiv$ reverse video image
- $\mathcal{R}(u, v) \equiv$ Fourier transform of reverse image
- $\mathcal{N}(u, v) \equiv$ Fourier transform of normal image
- $\tilde{\mathcal{R}}(u, v) \equiv$ Fourier transform of reverse image with dc component removed
- $\tilde{\mathcal{N}}(u, v) \equiv$ Fourier transform of normal image with dc component removed

The initial step was to take the discrete Fourier transform (DFT) of each window. The dc spatial frequency component was then set to zero for each DFT. Then the features were calculated using the process described in Eq (36).

$$\hat{z}_k = \sum_i \sum_j |\tilde{\mathcal{F}}_{ij}^k| \cdot |\tilde{\mathcal{G}}_{ij}^k| \quad (36)$$

where

- $\hat{z}_k \equiv$ feature value for window k using second method
- $\tilde{\mathcal{F}}_{ij}^k \equiv$ ij coefficient of modified DFT of image window k
- $\tilde{\mathcal{G}}_{ij}^k \equiv$ ij coefficient of modified DFT of template window k

Along with the crepuscular problem, many images may have varying degrees of contrast between the object and the background. The images contained in the data set used in the research contained a wide variation in contrast. The methodology used to create features to solve the crepuscular problem was also adequate for images with various contrast ratios. Once the features for the variable windows in a template are calculated, they along with the class of the input image are stored in a file for training the neural network classifier. The feature vector creation process is also used in the identification process, but the resultant vector is sent to a feedforward layer with fixed weights for the classification task. The training and classification tasks will be described in the next two sections.

5.6 Neural Network Training

Once the feature vector is created for an image it is stored in a data file. The next image is presented and the resultant feature vector is added to the data file. Once the entire database of images is processed, the feature vector file is used to train a backpropagation neural network (58) for training. Most of the feature vectors used in training each neural network used for the classification task had three features while the rest had four features. This limited number of features was sufficient to perform the classification task without memorizing the input training set. In addition, the limited number of features was necessary to insure that the dimensionality of the feature space was sufficiently small for the number of exemplars available for each class (4) (11). The backpropagation algorithm used for the classifier in this dissertation research was a modified version of the code developed by Ruck (43). The code was modified to store the statistical properties of the input training set as well as the weights produced during training. The statistics of the input training set are required before the weights can be used because of the transformation initially performed on the data before training the neural network. The data is initially transformed using the relation given by Eq (37).

$$x_i = \frac{x_i - \bar{x}_i}{\sigma_i} \quad (37)$$

where

x_i \equiv feature i of input feature vector

\bar{x}_i \equiv average value of feature i over entire training set

σ_i \equiv standard deviation of feature i over entire training set

This transformation must be applied to any feature vector presented to the back-propagation network to preserve the relative relationships between features in the feature vector in terms of the weight space represented by the stored weights.

Once the statistics of the training set are calculated, each vector is transformed as it is randomly called for training. The system architecture chosen consists of one neural network for each template. This results in each neural network being trained to recognize objects in its class (high output) while rejecting objects not in its class (low output). Figure 49 shows the results of a typical training run. As can be seen in Figure 49, there were 150 total feature vectors with 100 used for training and 50 used for testing the progress of the training. A neural network configuration of three inputs, ten hidden nodes and two output nodes was trained. After about ten epochs (1000 iterations), the neural network has been trained to within a few percentage points of its final classification accuracy (10,000 iterations). The first column in Figure 49 represents the iteration number, the second column is the training error, the third column is the training accuracy, the fourth column is the test set error and the last column is the test set accuracy. As can be seen in Figure 49 the training error is reduced as the network is trained longer and longer. Once the neural network is finished training, the weights along with the training set statistics are stored for use by the classifier in the system architecture. Figure 50 shows the entire training process.

The database contains 150 total vectors.

127 in class 0

23 in class 1

The number of vectors assigned to training by class are:

85 training vectors in class 0.

15 training vectors in class 1.

Means: 65562546.080000 215811838.480000 168215949.480000

Std dev: 31467133.595636 93980204.813248 80735503.810511

Wt Selection seed (initial_seed): 123456789

Db Partition seed (part_seed): 1918940490

Training Vector Selection seed(trn_seed): 1191645590

Weights file: demo0.wts0

Network Size: 3-10-0-2

Source database: demo0.ruck

Training Rate (eta): 0.3

Momentum (alpha): 0.7

Batch Size: 1

Features Used: All Features in demo0.ruck.

Fraction vectors assigned to training: 0.67

Normalization: 1

l: 0, ERR: 1.3191e-01, ACC: 8.5000e-01, TSERR: 1.2127e-01, TSACC: 8.4000e-01
l: 500, ERR: 1.3180e-02, ACC: 9.9000e-01, TSERR: 6.3784e-03, TSACC: 1.0000e+00
l: 1000, ERR: 1.1050e-02, ACC: 9.9000e-01, TSERR: 2.9547e-03, TSACC: 1.0000e+00
l: 1500, ERR: 1.4735e-02, ACC: 9.8000e-01, TSERR: 1.3206e-02, TSACC: 1.0000e+00
l: 2000, ERR: 1.0046e-02, ACC: 9.9000e-01, TSERR: 1.7777e-03, TSACC: 1.0000e+00
l: 2500, ERR: 6.1060e-03, ACC: 9.9000e-01, TSERR: 2.4001e-03, TSACC: 1.0000e+00
l: 3000, ERR: 8.0301e-03, ACC: 1.0000e+00, TSERR: 5.9739e-03, TSACC: 1.0000e+00
l: 3500, ERR: 4.3853e-03, ACC: 1.0000e+00, TSERR: 1.6871e-03, TSACC: 1.0000e+00
l: 4000, ERR: 4.7236e-03, ACC: 9.9000e-01, TSERR: 1.5341e-03, TSACC: 1.0000e+00
l: 4500, ERR: 3.9299e-03, ACC: 1.0000e+00, TSERR: 1.7096e-03, TSACC: 1.0000e+00
l: 5000, ERR: 4.3609e-03, ACC: 9.9000e-01, TSERR: 1.4958e-03, TSACC: 1.0000e+00
l: 5500, ERR: 3.3909e-03, ACC: 1.0000e+00, TSERR: 1.5677e-03, TSACC: 1.0000e+00
l: 6000, ERR: 3.3964e-03, ACC: 1.0000e+00, TSERR: 1.9260e-03, TSACC: 1.0000e+00
l: 6500, ERR: 2.8120e-03, ACC: 1.0000e+00, TSERR: 1.3994e-03, TSACC: 1.0000e+00
l: 7000, ERR: 3.0464e-03, ACC: 1.0000e+00, TSERR: 1.8416e-03, TSACC: 1.0000e+00
l: 7500, ERR: 2.9063e-03, ACC: 1.0000e+00, TSERR: 1.5992e-03, TSACC: 1.0000e+00
l: 8000, ERR: 2.4936e-03, ACC: 1.0000e+00, TSERR: 1.7030e-03, TSACC: 1.0000e+00
l: 8500, ERR: 2.5644e-03, ACC: 1.0000e+00, TSERR: 1.7902e-03, TSACC: 1.0000e+00
l: 9000, ERR: 2.5772e-03, ACC: 1.0000e+00, TSERR: 1.5121e-03, TSACC: 1.0000e+00
l: 9500, ERR: 2.6502e-03, ACC: 1.0000e+00, TSERR: 1.4633e-03, TSACC: 1.0000e+00
l: 10000, ERR: 2.4046e-03, ACC: 1.0000e+00, TSERR: 1.4952e-03, TSACC: 1.0000e+00

Figure 49. Typical Output From a Training Run for The Neural Network.

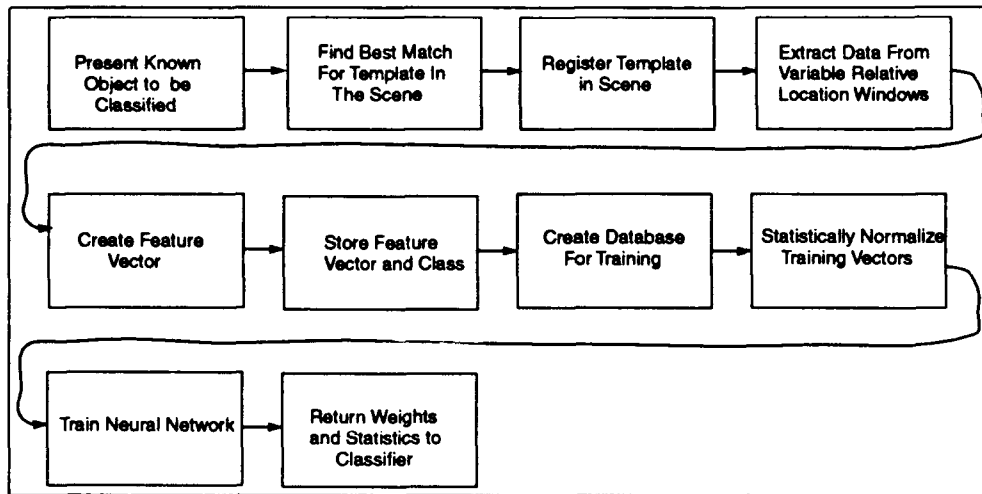


Figure 50. Block Diagram of Neural Network Training

5.7 Input Image Classification

When all of the neural networks associated with each of the desired classes have been trained, the system can perform classification on an unknown image. The classification process is depicted in Figure 51. As can be seen in the figure, the entire process is dependent upon the ability of each neural net classifier to accept or reject the object in question. Ideally one neural network would have a high in-class value while all others would have a low in-class value. Due to out-of-plane rotations, dissimilar object types within the same class, and contrast differences, more than one neural net output may be large or all may be fairly small. As can be seen in Figure /refclassifier, the input image is presented to the system and feature vectors are generated to feed each of the neural networks in the system. The template is registered at the best match location in the scene and the data is extracted from the input image at the variable window locations and a feature vector is created for processing. The feature vector is presented to the template's associated neural network for classification and the in-class output value is stored. This process is repeated until all of the templates and their associated neural network outputs have been stored. If the maximum output is larger than a predetermined threshold the

class of the object is declared to be the class associated with the largest in-class output. If the threshold value is not exceeded then no class is declared the winner. This could be due to an entirely new object being introduced or a known object in an orientation which would not allow for discrimination via the features extracted by the preprocessing steps. For instance, it is difficult to tell whether an El Camino is a car or a truck if you're looking at it from the front.

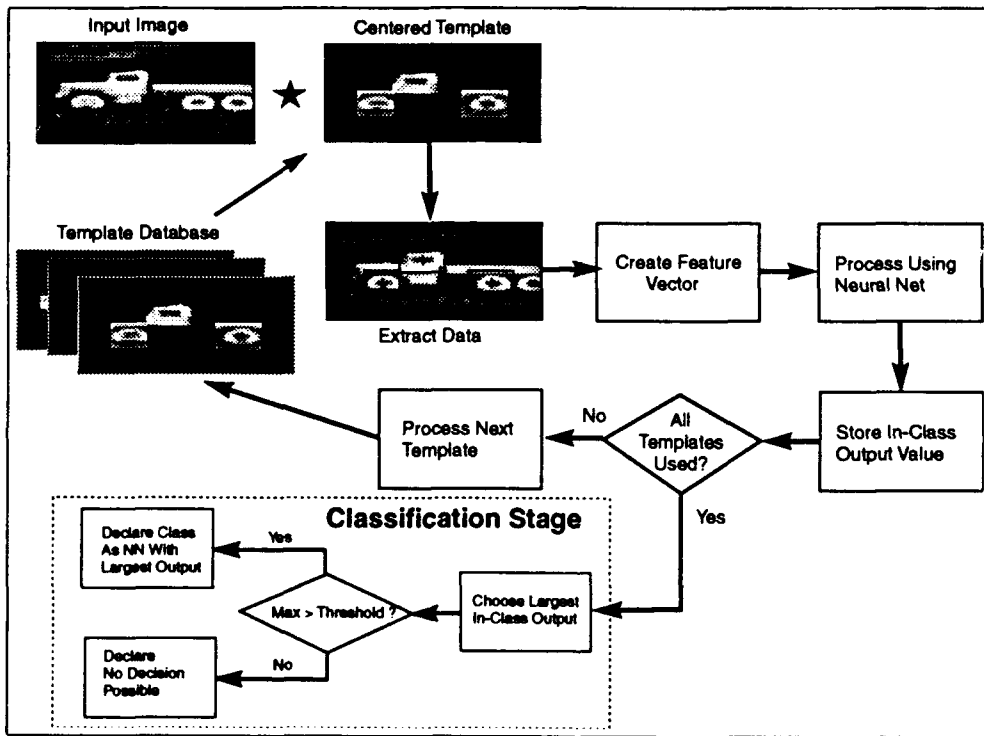


Figure 51. Block Diagram of Classifier

Once the image is classified, the next image in question is presented and the process is repeated until all of the unknown images are classified. The system architecture used for classification was fairly effective. The next section presents the classification performance for the system.

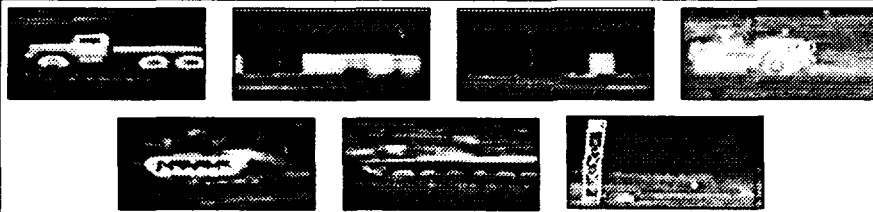
5.8 *System Performance*

The relative location architecture as described in the previous section was by far the best classification system for the data set used in this dissertation research. The results from the relative location system are compared to those using matched filter templates (whole scene and segmented templates). The general recognition capability of the relative location system was generally greater than 90 percent while the matched filter using the entire scene was 82 percent. When the cropped templates are used, the classification accuracy drops to 51 percent. In addition, when a direct matched filter approach is used with the templates developed for the relative location concept the classification accuracy drops to 41 percent. The results obtained using the relative location concept are presented in detail in the next section.

5.9 *Discussion of Results*


The results obtained using several different matched filters as well as the relative location architecture are presented in Tables 6-10. Table 6 contains the results obtained when the templates are full images. The actual templates used in the classification task are shown above the results in Table 6. The templates were correlated with the FLIR database and the template with the maximum correlation was chosen as the class of the object in question. The results are fairly good (82 percent), but they are deceptive. The majority of the correlation value is determined by the background. This becomes evident as we consider the results given in Table 7 which represent a more realistic matched filter set. The templates were generated by removing the background from the image while retaining the desired object to be recognized in the template. The overall result of 52 percent would never be acceptable in a classification system. The results presented in Table 8 are the matched filter classification accuracy when the templates used for the relative location architecture are used as matched filter templates. The result of 41 percent is very low and shows how the energy in the template plays a significant role in matched filter correlation. Table 9 gives the results when the relative location architecture is used for the classification

task. The result of 93 percent is much better than that obtained using traditional matched filter approaches. In addition, the reverse video classification problem is overcome using the methodology of section 5.5. Table 10 gives the result for a combination of normal and reverse video images and the associated classification accuracy of the relative location architecture. As can be seen, the overall accuracy of the system is unchanged. The next section explains why the relative location concept is better than matched filter methods in classifying FLIR imagery.




Results Using Correlation With Full Templates			
Target Type	Total Targets	Correct	Pct Correct
Truck	23	23	100 %
Large Tgt Board	10	9	90 %
Small Tgt Board	13	13	100 %
Jeep	24	24	100 %
Tank Set 1	43	17	40 %
Tank Set 2	21	21	100 %
Tower	16	16	100 %
Overall	150	123	82 %

Table 6. Classification Results Using Full Templates As Matched Filters




Results Using Correlation With Cropped Templates			
Target Type	Total Targets	Correct	Pct Correct
Truck	23	23	100 %
Large Tgt Board	10	6	60 %
Small Tgt Board	13	0	0 %
Jeep	24	24	100 %
Tank Set 1	43	2	5 %
Tank Set 2	21	21	100 %
Tower	16	0	0 %
Overall	150	76	51 %

Table 7. Classification Results Using Cropped Templates As Matched Filters




Results Using Relative Locations As Matched Filter			
Target Type	Total Targets	Correct	Pct Correct
Truck	23	14	61 %
Large Tgt Board	10	5	50 %
Small Tgt Board	13	0	0 %
Jeep	24	24	100 %
Tank Set 1	43	4	9 %
Tank Set 2	21	15	71 %
Tower	16	0	0 %
Overall	150	62	41 %

Table 8. Classification Results Using Relative Location Templates As Matched Filters



Results Using Relative Locations			
Target Type	Total Targets	Correct	Pct Correct
Truck	23	23	100 %
Large Tgt Board	10	10	100 %
Small Tgt Board	13	12	92 %
Jeep	24	23	96 %
Tank Set 1	43	40	93 %
Tank Set 2	21	17	81 %
Tower	16	15	94 %
Overall	150	140	93 %

Table 9. Classification Results Using Relative Locations



Results Using Relative Locations On Hot and Cold Object Input Images			
Target Type	Total Targets	Correct	Pct Correct
Truck	46	46	100 %
Large Tgt Board	20	20	100 %
Small Tgt Board	26	23	88 %
Jeep	48	46	96 %
Tank Set 1	86	80	93 %
Tank Set 2	42	35	83 %
Tower	32	30	94 %
Overall	300	280	93 %

Table 10. Classification Results Using Relative Locations For Both Reverse and Normal Input Images

5.10 Why Relative Locations Are Better Than Matched Filter Techniques

The relative location technique was able to improve upon the matched filter technique (93 percent versus 41 percent) using the same templates. This section will discuss why this is possible. The matched filter correlator can be considered as a global type operation. The individual correlations of the windows in the template are summed together to produce an output value. In addition, when the input image is energy normalized it uses a different normalization constant than the template. Other templates with greater area will tend to have higher correlation values than might otherwise be expected simply due to the larger number of pixels which can be summed during the correlation process. Ideally the matched filter pair would produce a value of unity (autocorrelation) each time an image containing an object of the same type as the matched filter was encountered. In real world imagery the autocorrelation value of the matched filter is never unity unless full image templates are used. In the case of full templates the autocorrelation value of unity is obtained when the input image is identical to the one selected as the template. In addition, the matched filters used in this dissertation research are not all to the same scale because the images themselves are not to the same scale. The scale differences does cause some difficulties because of the normalization problem presented earlier. The scale problem is probably the biggest factor in the failure of the towers and small target boards to match even the image from which the template was created. The relative location concept uses local as well as global information to produce a classification decision. The initial registration of the template on the input image is identical to matched filter correlation which is a global process. The global sum is used to find the best match in the scene with the template. The difference is that once the best match location is found, the local information extracted from the locations of the variably sized windows in the template is used to make the classification decision. This is possible because the backpropagation neural network is using the relative values between the windows rather than the sum of the values. To better understand the process consider Figure 52. The truck in the input image is correlated with both the truck template and the small target board template. The resultant feature vectors

as developed using Eq (36) and statistically normalized using Eq (37) are presented as well as the output of the truck and small target neural networks. As can be seen in the figure, the best match in the scene for the small target template occurs in the first feature. Note however, that for the small target feature vector all three values are high while the normal in-class response would have one feature high and the other two features low. The truck template has a good match and all three features have high values. Each neural network classifier has the ability to parse the input feature space into the appropriate decision regions and produces the desired result – a high value for the truck in-class output and a low value for the small target in-class output. The significance of the relative location concept lies in the fact that the relative window locations are in effect able to extract data from areas where it is the most useful and reject information in the regions which are not useful in the classification task. This is precisely what the eye scan patterns in Figure 43 are indicating. The really important locations are revisited a number of times while the brain performs “perceptual hypotheses.” The relative location process does have some drawbacks which are discussed in the next section.

5.11 Limitations of the Relative Location Concept

The previous section discussed why the relative location concept is better able to classify the FLIR image set used in the dissertation research. This section will discuss some of the limitations of the concept and possible solutions to overcome the limitations. The biggest limitation of the relative location concept is that the relative spacing of the windows themselves is not scale invariant. However, the DFT calculation performed within each window is scale invariant. The solution is simple as long as the range to the target is known. The scale factor required to transform the observed input image to the same scale as the template is given by Eq (38).

$$SF = \frac{\text{Template Range}}{\text{Image Range}} \quad (38)$$

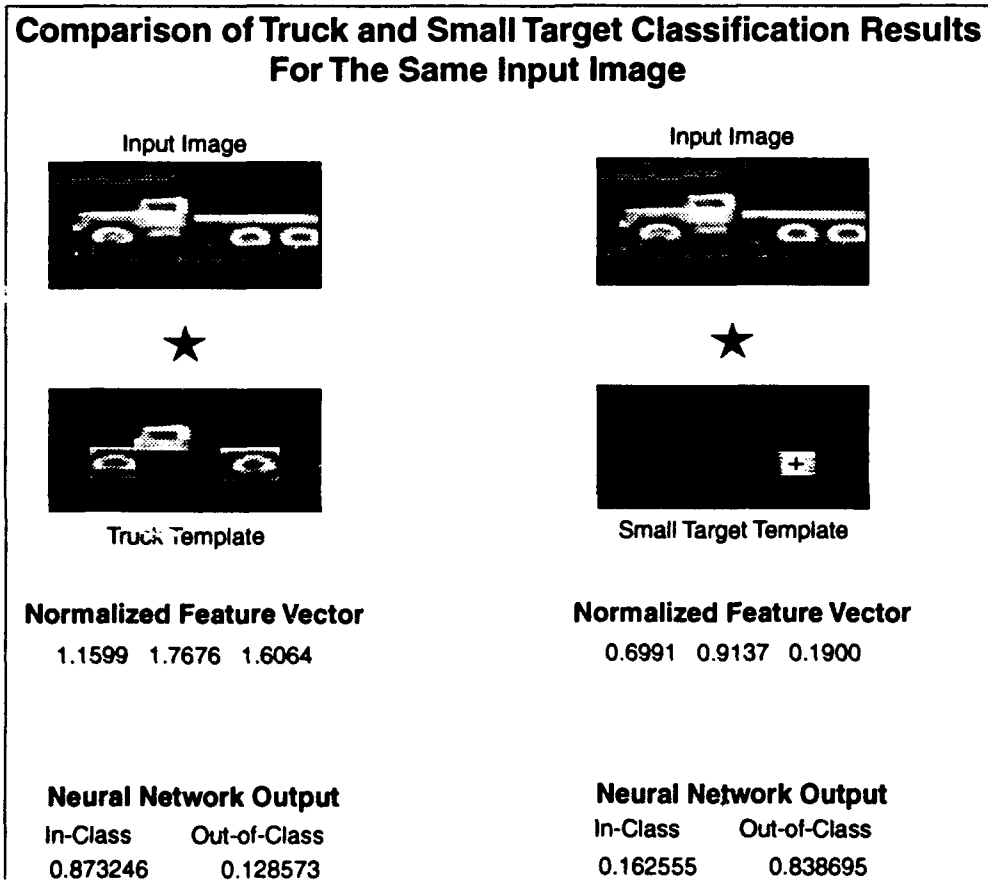


Figure 52. Comparison of Two Separate Templates And Their Separate Neural Network Outputs For The Same Input Image

Once the scale factor is determined then, through the use of similar triangles, the following transformation can be used to translate an image into the same space that the templates were generated in.

$$[\tilde{x}, \tilde{y}] = [x, y] \cdot \begin{vmatrix} S_x & 0 \\ 0 & S_y \end{vmatrix} \quad (39)$$

where

$$S_x = \text{SF} \cdot \text{length of image in } x \text{ direction}$$

$$S_y = \text{SF} \cdot \text{length of image in } y \text{ direction}$$

$$\tilde{x} \equiv \text{transformed pixel location in } x \text{ direction}$$

$$\tilde{y} \equiv \text{transformed pixel location in } y \text{ direction}$$

The scaling problem was determined to be of some concern but certainly one which was manageable. The other limitation of the relative location concept is that of registering the template properly on the scene of interest. This is due in part because of the crepuscular conditions inherent in the early morning and late evening. The problem is that even though the classifier is able to overcome the black-on-white versus white-on-black dilemma, the registration is dependent upon knowing what type of image is being viewed. Of course two templates could be stored with the maximum from each used as the centroid for data processing. This results in a considerable amount of processing overhead. Another solution would be to use windows based upon the centroid of the object being classified.

5.12 Conclusions

The relative location architecture developed in this chapter has been shown to be useful in classifying FLIR imagery over a wide range of conditions. The relative location architecture was able to correctly classify 93 percent of the images as compared to 41

percent when using the same templates as matched filters. The relative location concept allows for the localized application of windows to extract features while retaining the global perspective because the relative locations of the windows are fixed once defined by the user in the creation of the template. The next chapter reviews the contributions made using the relative location concept as well as the other principles presented in the dissertation.

VI. Conclusions and Recommendations

6.1 Conclusions

The research conducted in this dissertation was directed at finding ways to improve upon the classification techniques currently employed in identifying objects in forward looking infrared imagery. The results presented in the dissertation are excellent considering the various aspect angles and lighting conditions encountered in the FLIR imagery. Three different approaches to solving the pattern recognition problem are presented along with a method for determining which features are the most salient for performing the classification task. The first approach used the combination of a roving window, Gabor filters and a bank of backpropagation neural networks acting as a Bayesian optimal estimator. The second approach used the roving window, locations of resonant Gabor filters, various distance metrics, and a backpropagation neural network performing as a Bayesian classifier. The third method departed from the Gabor approach and used instead a user defined template which contained a number of relative location windows for data extraction and processing combined with a bank of backpropagation neural networks performing as Bayesian classifiers. As was shown in the course of the research, the relative location concept is a robust method for extracting useful features from FLIR imagery. The relative location architecture allows for user defined template creation which when combined with the extraction of data from the best match location in the scene provides an improved method of classifying FLIR imagery. The results presented in Chapter 5 show conclusively that the relative location architecture outperformed more traditional matched filter techniques.

As a direct result of the initial research a method was developed (see Chapter 4) to identify the most useful features in performing the pattern recognition task. This method was developed from the Bayesian properties of feedforward neural networks. The application of the method to the pattern classification problem resulted in a reduction of features from 1080 to three with no loss in accuracy. The next section presents the

contributions made to the body of scientific knowledge as a direct result of the research conducted in this dissertation.

6.2 Contributions

There were four major contributions to the scientific body as a result of the dissertation research: 1) A new architecture based upon the gestalt representation of objects using Gabor coefficients as inputs. 2) A new classification architecture which uses relative location information of selected textures as a method of classifying tactical targets. 3) A neural network architecture which allows for the addition of new classes without extensive retraining. 4) A feature saliency metric based upon the Bayesian nature of feedforward neural networks.

The neural network architectures used as classifiers for the target recognition problem were based upon two main themes. The first was that of learning the gestalt representation (see Chapter 2) of each of the classes with the neural network performing as a function estimator with the classification task performed using a MSE measurement. The overall classification accuracy obtained using the gestalt representation was 62.3 percent which was an improvement over the 43.2 percent reported by Lazofson (27:70), but inadequate for a classifier. The second approach was a more traditional approach of using various distance metrics from the centroid of a cluster of the most resonant Gabor filter correlations. The distances between the centroid and each of the Gabor filter correlation peaks were used as feature vector descriptors for the target classes with the classifier performing as a Bayesian statistical classifier (see Chapter 3). The best result using the various distance metrics of Chapter 3 was a classification accuracy of 73 percent which was also inadequate. After the first two system architectures showed limited classification accuracy, a third approach using the relative locations of specific spatial features was developed as described in Chapter 5. The results obtained using the relative location concept were 100 percent over the three class problem used in the two previous approaches. This technique was then applied to a seven class problem which included crepuscular (twilight) conditions in the

FLIR imagery. In the seven class problem the relative location technique was able to correctly classify 93 percent of the FLIR images. An outgrowth of the second approach was that it readily became apparent that a limitation on the number of features would be required to preclude memorization (4, 11) problems in the statistical classification scheme. This led to a major contribution in the area of feature saliency from a Bayesian point of view. Ruck (45) and Gish (15) proved in separate but similar analyses that the multilayer feedforward neural network is indeed a Bayesian classifier. Because the outputs of a feedforward neural network, when trained under fairly minimal requirements, become the a posteriori class conditional probabilities when the network converges, a logical extension was to explore using the probability result as a means of performing a sensitivity analysis, as described in Chapter 4, between the output and a given input feature. An interesting result of the analysis was that the intuitive metric proposed by Ruck (44) was actually the same (within a multiplicative scaling constant) as that obtained from the Bayesian analysis. Hence the metric Ruck proposed was an optimal one from an feature saliency viewpoint. This further removes any doubt as to the Bayesian behavior of feedforward networks. The results highlighted here are discussed in detail in the dissertation.

6.3 Recommendations

The dissertation research was successful in classifying FLIR imagery but had some problems unrelated to the classification task. These problems were due to certain assumptions being made concerning the best location to extract data from the scene of interest. The registration problem is due to not knowing whether there is a black-on-white scene or white-on-black scene being presented to the system architecture. Once the type of scene is known, the registration dilemma is solved. In the absence of this knowledge a pair of templates would need to be maintained for each class to be identified. This results in a doubling in processing time. There is plenty of research which could be done to improve upon the registration problem.

Many of the contributions made in this dissertation followed work done by others which had a few “what if ” questions remaining. When the “what if” questions were solved new insights were made for solving the pattern recognition problem. Take the results presented here and build upon them to make even better pattern classifiers in the future.

Bibliography

1. Kevin W. Ayer. Gabor transforms for forward looking infrared image segmentation. Master's thesis, Air Force Institute of Technology, 1989.
2. Alan Conrad Bovik, Marianna Clark, and Wilson S. Geisler. Multichannel texture analysis using localized spatial filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):55–73, January 1990.
3. David Casasent and Raghuram Krishnapuram. Curved object recognition by hough transformations and inversions. *Pattern Recognition*, 20(2):181–188, 1987.
4. Thomas M. Cover. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Transactions on Electronic Computers*, EC-(14):326–334, June 1965.
5. Y. Le Cun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, and L.D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1:541–551, 1989.
6. John G. Daugman. Two-dimensional spectral analysis of cortical receptive field profiles. *Vision Research*, 20:847–856, 1980.
7. John G. Daugman. Complete discrete 2-d gabor transforms by neural networks for image analysis and compression. *IEEE Transactions on Acoustics Speech and Signal Processing*, 36(7):1169–1179, July 1988.
8. Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, New York, 1973.
9. S.A. Dudani et al. Aircraft identification by moment invariants. *IEEE Transactions on Computers*, C-26:39–45, 1977.
10. David L. Flannery, John S. Loomis, and Mary Milkovich. Transform-ratio ternary phase-amplitude filter formulation for improved correlation discrimination. *Applied Optics*, 27(19):4079–4083, October 1988.
11. Donald H. Foley. Considerations of sample and feature size. *IEEE Transactions on Information Theory*, IT-18:618–626, September 1972.
12. Kunihiro Fukushima. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural Networks*, 1:119–130, 1988.
13. D. Gabor. Theory of communication. *The Journal of the Institution of Electrical Engineers*, 93:429–457, 1946.
14. Arthur Ginsburg. Psychological correlates of a model of the human visual system. Master's thesis, Air Force Institute of Technology, June 1971.

15. Herbert Gish. A probabilistic approach to the understanding and training of neural network classifiers. In *Proceedings of International Conference on Acoustics Speech and Signal Processing 90*, pages 1361–1364, 1990.
16. Martin S. Gizzy, Ephraim Katz, Robert A. Schumer, and J. Anthony Movshon. Selectivity for orientation and direction of motion of single neurons in cat striate and extrastriate visual cortex. *Journal of Neurophysiology*, 63(6):1529–1543, June 1990.
17. Robert Hecht-Nielsen. Nearest matched filter classification of spatiotemporal patterns. *Applied Optics*, 26(10):1892–1899, May 1987.
18. Marvin L. Hill. Feature extraction using the hough transform. Master's thesis, Air Force Institute of Technology, 1987.
19. Joseph L. Horner, editor. *Optical Signal Processing*. Academic Press, Inc., 1987.
20. Joseph L. Horner and James R. Leger. Pattern recognition with binary phase only filters. *Applied Optics*, 24(5):609–611, March 1985.
21. D. Hubel and T. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *Journal of Physiology*, 160:106–154, 1962.
22. Jones, P. Judson, and Larry A. Palmer. The two-dimensional spatial structure of simple receptive fields in the striate cortex of cats. *Journal of Neurophysiology*, 58:1187–1211, December 1987.
23. Matthew Kabrisky. *A Proposed Model for Visual Information Processing in the Human Brain*. University of Illinois Press, 1966.
24. Fumio Kanaya and Shigeki Miyake. Bayes statistical behavior and valid generalization of pattern classifying neural networks. *IEEE Transactions on Neural Networks*, 2(4):471–475, July 1991.
25. Laurence C. Lambert. Evaluation and enhancement of the afit autonomous face recognition machine. Master's thesis, Air Force Institute of Technology, 1987.
26. Steven R. Lay. *Analysis with an Introduction to Proof*. Prentice Hall, New Jersey, 2nd edition, 1990.
27. Laurence E. Lazofson. A biologically inspired neural network architecture for image processing. Master's thesis, Air Force Institute of Technology, December 1990.
28. Curt A. Levey, Richard D. Crawshaw, and J. Michael Oyster. Gabor wavelet analysis for target detection and classification. In *2nd Annual Tri-Service Neural Network Applications Workshop*, pages 1–4, 1991.
29. Richard P. Lippmann. Pattern classification using neural networks. *IEEE Communications Magazine*, pages 47–63, November 1989.
30. Aleksandr Romanovich Luria. *Higher Cortical Functions In Man* Basic Books Inc., 1966.

31. Kishan G. Mehrotra, Chilukuri K. Mohan, and Sanjay Ranka. Bounds on the number of samples needed for neural learning. *IEEE Transactions on Neural Networks*, 2(6):548–558, November 1991.
32. Paul L. Meyer. *Introductory Probability and Statistical Applications*. Addison-Wesley Publishing Company, 2nd edition, 1970.
33. Marvin L. Minsky and Seymour A. Papert. *Perceptrons – Expanded Edition*. MIT Press, 1988.
34. Frederick C. Mish, editor. *Webster's Ninth New Collegiate Dictionary*. Merriam Webster Inc., 1990.
35. Richard A. Oberndorf. Analysis of visual illusions using gabor filters. Master's thesis, Air Force Institute of Technology, 1990.
36. Kevin L. Priddy, Steven K. Rogers, Dennis W. Ruck, Gregory L. Tarr, and Matthew Kabrisky. Bayesian selection of important features for feedforward neural networks. *Neurocomputing*, 1992. Accepted for publication Paper Number 92-92.
37. Michael C. Roggemann. *Multiple Sensor Fusion for Detecting Targets in FLIR and Range Images*. PhD thesis, Air Force Institute of Technology, June 1989.
38. Michael C. Roggemann, J.P. Mills, M. Kabrisky, S.K. Rogers, and J.A. Tatman. Multiple sensor tactical target detection in FLIR and range images. In *Proceedings of Tri-Service Data Fusion Symposium*, Johns Hopkins University, May 1989.
39. F. Rosenblatt. *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Spartan Books, Washington, 1959.
40. Michael W. Roth. Survey of neural network technology for automatic target recognition. *IEEE Transactions on Neural Networks*, 1(1):28–43, March 1990.
41. Dennis W. Ruck. Multisensor target detection and classification. Master's thesis, Air Force Institute of Technology, 1987.
42. Dennis W. Ruck. Multisensor fusion target classification. In *SPIE Symposium on Optics, Electro-Optics, and Sensors*, 1988.
43. Dennis W. Ruck. *Characterization of Multilayer Perceptrons and their Application to Multisensor Automatic Target Detection*. PhD thesis, Air Force Institute of Technology, December 1990.
44. Dennis W. Ruck, Steven K. Rogers, and Matthew Kabrisky. Feature selection using a multilayer perceptron. *The Journal of Neural Network Computing*, 2(2):40–48, Fall 1990.
45. Dennis W. Ruck, Steven K. Rogers, Matthew Kabrisky, Mark E. Oxley, and Bruce W. Suter. The multilayer perceptron as an approximation to a Bayes optimal discriminant. *IEEE Transactions on Neural Networks*, 1(4), 1990.
46. David E. Rumelhart, James L. McClelland, and the PDP Research Group. *Parallel Distributed Processing*, volume 1: Foundations. MIT Press, 1986.

47. David E. Rumelhart and James L. McClelland. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1. The MIT Press, 1986.
48. Robert Schalkoff. *Pattern Recognition Statistical Structural and Neural Approaches*. John Wiley and Sons, 1992.
49. Gregory L. Tarr. *Multi-Layered Feedforward Neural Networks for Image Segmentation*. PhD thesis, Air Force Institute of Technology, December 1991.
50. Michael R. Teague. Image analysis via the general theory of moments. *Journal of the Optical Society of America*, 70:920–930, 1980.
51. Carl W. Tong et al. Multisensor data fusion of laser radar and forward looking infrared (flir) for target segmentation and enhancement. In *SPIE Vol. 782 Infrared Sensors and Sensor Fusion*, pages 10–19, 1987.
52. Roger B. H. Tootell, Martin S. Silverman, Eugene Switkes, and Russell L. De Valois. Deoxyglucose analysis of retinotopic organization in primate striate cortex. *Science*, 218:902–904, November 1982.
53. Julius T. Tou and Rafael C. Gonzalez. *Pattern Recognition Principles*. Addison-Wesley Publishing Company, 1st edition, 1974.
54. M. R. Turner. Texture discrimination by gabor functions. *Biological Cybernetics*, 55:71–82, 1986.
55. Russell L. De Valois, Lisa G. Thorell, and Duane G. Albrecht. Periodicity of striate-cortex-cell receptive fields. *Journal of the Optical Society of America A*, 2(7):1115–1122, 1986.
56. Christopher P. Veronin, Steven K. Rogers, Matthew Kabrisky, Byron Welsh, Kevin L. Priddy, and Kevin W. Ayer. An optical image segmentor using wavelet filtering techniques as the front end of a neural network classifier. In *Proceedings of the SPIE 1991 International Symposium and Exhibition on Optical Engineering and Photonics in Aerospace Sensing*, 1991.
57. Philip D. Wasserman. *Neural Computing: Theory and Practice*. Van Nostrand Reinhold, New York, 1989.
58. Paul J. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard University, 1974.
59. Francis T. S. Yu, Suganda Jutamulia, Tsongneng W. Lin, , and Don A. Gregory. Adaptive real-time pattern recognition using a liquid crystal tv based joint transform correlator. *Applied Optics*, 26(8):1370–1372, April 1987.

Vita

Captain Kevin L. Priddy was born on December 23, 1955. Capt Priddy graduated from North Ft. Myers High School, N. Ft. Myers Florida, in June 1974. He entered the Army where he served for five years as a Pershing Missile electro-mechanical repairman. After meeting his wife in Oklahoma, they were married in April 1976 and moved to Germany for three years. He left the Army in May 1979 to pursue a Bachelor of Science degree from Brigham Young University. Capt. Priddy entered the Air Force in May of 1982 as a newly commissioned second lieutenant after receiving his BSEE degree in April 1982. Capt. Priddy served as a staff officer in the Tactical Air Forces Interoperability Group at Langley AFB VA from May 1982 to May 1984 before entering the School of Engineering, Air Force Institute of Technology at Wright-Patterson AFB OH in June 1984. In December 1985 he was awarded the Master of Science degree in Electrical Engineering with an emphasis in electro-optics and semiconductor devices. Following graduation he was assigned to the Avionics Laboratory at Wright-Patterson AFB where he was the program manager for the HAVE GLANCE infrared countermeasure program. He reentered AFIT to pursue the PhD degree in July 1989 with an emphasis in pattern recognition and neural networks. He is married to Wendy (Battenfield) Priddy of Lawton, Oklahoma and they have four children: Vanessa, Jessica, Michael and Samuel.

Permanent address: Capt Kevin L. Priddy
c/o Phyllis A. Priddy
3336 N. Key Dr. Apt J-6
North Fort Myers, FL 33903

April 1992

PhD Dissertation

**Feature Extraction and Classification of FLIR Imagery Using Relative Locations
of Non-Homogeneous Regions with Feedforward Neural Networks**

Kevin L. Priddy, Capt, USAF

Air Force Institute of Technology, WPAFB OH 45433-6583

AFIT/DS/ENG/92-01

Edmund G. Zelnio
Chief, Target Recognition Technology Branch
WL/AARA
Wright-Patterson AFB OH 45433

Approved for Public Release; Distribution Unlimited.

The classification of forward looking infrared (FLIR) imagery is explored using Gabor transform decomposition and relative locations of non-homogeneous regions combined with feedforward neural networks. A feature saliency metric is developed from a Bayesian sensitivity analysis of feedforward neural networks. This metric is then used to reduce the dimensionality of the feature vectors used to identify FLIR imagery without any degradation of classification accuracy. Several system architectures are developed using a roving window combined with a series of Gabor filters to produce feature vectors for presentation to a neural network classifier. One architecture uses the Gabor filter coefficients to learn the gestalt of known images. Several of the gestalt networks are then combined to determine the class of an unknown image. Another architecture uses centroid metrics for the cluster of Gabor resonances to feed a backpropagation network acting as a traditional Bayesian classifier. A system architecture is developed which uses the relative locations of texture regions within a user defined template to classify imagery. The relative location architecture is shown to outperform traditional matched filter classifiers. The relative location architecture is shown to be robust in solving the problems presented by crepuscular lighting conditions.

Image Processing, Gabor Transforms, FLIR Image Classification, Feature Saliency, Neural
Networks

119

Unclassified

Unclassified

Unclassified

UL