BINARY TREE CLASSIFIER AND CONTEXT CLASSIFIER

by

Hyonam Joo

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of
Master of Science
in
Electrical Engineering

APPROVED:

R. M. Haralick, Chairman

K. B. Yu

P. M. Lapsa

March, 1985

Blacksburg, Virginia

BINARY TREE CLASSIFIER AND CONTEXT CLASSIFIER

by

Hyonam Joo

R. M. Haralick, Chairman

Electrical Engineering

(ABSTRACT)

Two methods of designing a point classifier are discussed in this paper, one is a binary decision tree classifier based on the Fisher's linear discriminant function as a decision rule at each nonterminal node, and the other is a contextual classifier which gives each pixel the highest probability label given some substantially sized context including the pixel.

Experiments were performed both on a simulated image and real images to illustrate the improvement of the classification accuracy over the conventional single-stage Bayes classifier under Gaussian distribution assumption.

ACKNOWLEDGEMENTS

The author sincerely wishes to thank Dr. R.M. Haralick for his encouragement, for his suggestions, and for editing this thesis.

The author also gives many thanks to his family for their continuous encouragement and moral support.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

CHAPTER 1. INTRODUCTION

A decision tree classifier and a contextual classifier have been reported to be more accurate in pixel labeling than a single-stage classifier by many authors in recent years [7-13].

In the decision tree classifier, an unknown pixel is classified into a class through several decisions successively following a path in the tree. Starting at the root node and at each encountered nonterminal node, an internal partial decision is made on the path that the classification process should follow until a terminal node which has an associated class (label) is reached. The main advantages of the decision tree classifier reported so far are that the classification accuracy and the computation efficiency can be improved. In this paper, one possible way of designing a decision tree classifier emphasizing the classification accuracy advantage is presented which can be used when there are not so many features. In general, the design of a decision tree classifier consists of the following three components.

1) a tree structure or hierarchical ordering of the pattern classes
2) the choice of features to be used at each nonterminal

node

3) the decision rule to be used at each nonterminal node

For the decision tree classifier described in this paper, a binary tree structure is chosen for simplicity and all the features are used in the decision making process. At each nonterminal node, all of the pattern classes contained in that node are divided into two groups and the Fisher's linear discriminant function is computed together with the threshold value that maximizes the class purity of the child nodes. Every possible grouping of the set of classes contained in a nonterminal node are considered and that grouping which results in a decision which gives maximum purity to the children nodes is selected. The corresponding Fisher's linear discriminant function and the threshold value are used as a decision rule at that node.

The contextual classifier is characterized by the fact that it classifies an unknown pixel using the entire context of the image or a substantially sized context neighboring the pixel. Basically, the effect of context is that some entity Z can have certain properties, when Z is viewed in isolation, which change when Z is viewed in some context [4]. One might expect that classification accuracy is higher if an unknown pixel is classified using context rather than when it is classified using only the measurement made on that pixel

without context, and it turns out to be true in most cases. For example, a single pixel is not likely to be classified as water if it is surrounded by the pixels classified as ground in a remotely sensed data. The classification result of the conventional noncontextual classifier leaves many isolated pixels and many small groups of pixels not connected with the blob they belong to. Thus, in the last few years there has been a trend to increase the use of context in the labeling operation.

The use of context in pixel labeling can be found in many papers. The dominant contextual technique has been one of cooperative processing of neighboring pixels by a relaxation technique. Toussaint [4] presented a tutorial survey of techniques for using contextual information in pattern recognition emphasising the problem of text recognition, and Haralick [1] gave a survey of decision making in context. Tilton and Swain [2,3] use a p-context array which contains spatial information of (p-1) pixels surrounding and neighboring the current pixel in the context pixel classification process. They derived the optimal decision rule using the context array and focus their attention on finding an unbiased estimate of the context function which is a statistical characterization of the context to be used in the decision rule. Yu and Fu [6] also noted that the spectral information of the surrounding pixels is correlated with the center pixel

being considered. They investigated the spatial correlation between pixels and developed a spatial stochastic recursive contextual classification method. Wharton [5] presented a contextual anaysis procedure based on the local frequency distribution of scene components and showed a two-stage contextual classifier.

In this paper, we present a theory and an algorithm of a new contextual classifier that assigns each pixel the highest probability label given some substantially sized context involving the pixel. The algorithm takes the form of a recursive neighborhood operator and turns out to be a two-pass algorithm first applied in a top down scan of the image and then in a bottom up scan of the image.

Chapter 2 describes the binary decision tree classifier and the context classifier is presented in chapter 3. Experiments performed on the basis of those two classification techniques and discussion on the results are given in chapter 4 followed by concluions.

CHAPTER 2. BINARY DECISION TREE CLASSIFIER

Compared to the single-stage classifier which tests an unknown pixel against all classes and classifies the unknown pixel to one of the classes considered in the classification process, the decision tree classifier classifies the unknown pixel through a hierarchical decision procedure. The classification process can be described by means of a tree, in which each terminal node represents one pattern class, i.e. the final classification, and the interior nodes represent collection of classes. In particular, the root node represents the entire collection of classes into which a pattern may be classified [8]. Figure 1 shows a typical binary decision tree classifier. When an unknown pixel enters the decision tree at the root node, a decision rule associated with the root node is applied to the pixel to determine the decendent path that it will follow. This process is repeated until a terminal node is reached. Every terminal node has an associated class that the pixel is assigned in.

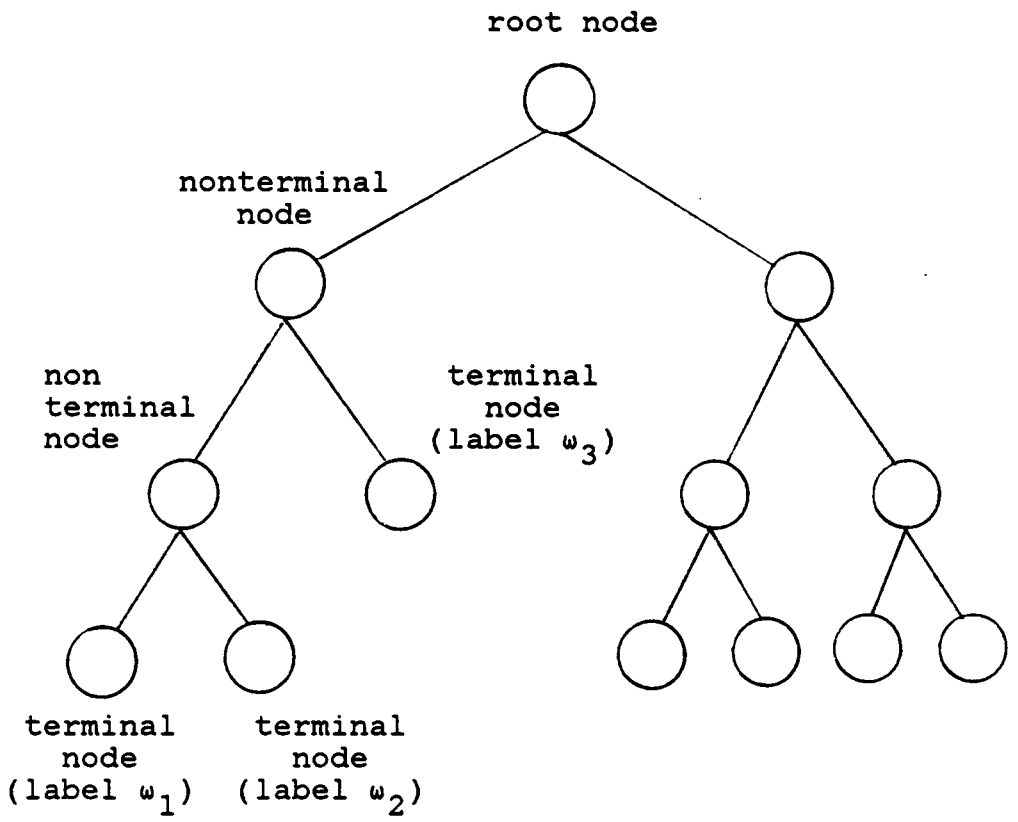

Figure 1. An example of a binary decision tree: ω_i represents label associated with the terminal node.

The binary decision tree classifier is studied by many researchers [9,10,11,12] who discuss methods of selecting optimal features at each nonterminal node and give decision rules associated with the selected feature sets. But most of them assume that complete probabilistic information is given in their derivation of the optimal strategy. Even though it has been reported that the decision tree classifier gives a more accurate classification result and more efficient computation, there are still problems to be solved in the design of optimal decision tree classifier. The three major problems consist of designing a tree structure, choice of features to be used at each nonterminal node, and the choice of the decision rule to be used at each nonterminal node. Kurzynski [8] focuses attention on the third problem and derives an optimal decision rules with respect to the probability of misclassification for a given tree structure and features used at each nonterminal node.

In this chapter we present a nonparametric method of designing a binary decision tree classifier that uses a simple linear discriminant function as a decision rule at each nonterminal node.

For a tree with a simple structure, such as a binary tree with a linear discriminant function as considered in this paper, the number of descendent nodes are fixed to two and only thing to be specified is the discriminant function. But

for cases where the tree structure is complex, the number of descendent nodes of a nonterminal node is not fixed, and also the decision functions can be more complicated. As the number of features increases, we meet another problem of choosing a best feature set to be used in separating the sample space of each nonterminal node. We assume in our derivation that the number of features are small and focus our attention on the method of obtaining a simple decision rule.

Since all the features obtained are to be used in the decision rule, the main concern in the design of our binary decision tree classifier is the separation of two groups of classes among training samples at each nonterminal node.

Let X be the d-dimensional feature vector measured at a pixel. Then the linear discriminant function is a linear function of the form,

$$g(X) = V^t X + v_{d+1} \qquad (2.1)$$

where V is a weighting vector and v_{d+1} is called the threshold. This form of linear discriminant function separates the feature space into two regions; the decision rule assigns X to one region if $g(X) > 0$ and to the other region if $g(X) < 0$, or if $g(X) = 0$, the indeterminacy may be resolved as one pleases. It is easy to see that using a linear discriminant function is equivalent to projecting a pattern X onto a line

in the direction of vector V and comparing the position of the projection with that of the threshold v_{d+1}.

In this paper, Fisher's linear discriminat function is used as a decision rule at each nonterminal node. Fisher's linear discriminant function is obtained by maximizing the Fisher discriminant ratio which, discribed below, is the ratio of between class scatter to within class scatter.

Let $z = V^t X$ be the projected point, μ_i denote the class conditional mean vector of X of class i, and $\mu = \Sigma_i P_i \mu_i$ denote the mean vector of the mixture distribution, where P_i represents the probability of the class i in the training sample. Then the between class covariance (or scatter) matrix S_b for a two classes case can be expressed as,

$$S_b = \sum_{i=1}^{2} P_i(\mu_i-\mu)(\mu_i-\mu)^t \qquad (2.2)$$

If we let S_i be the class conditional covariance matrix of class i, then

$$S_i = E_i \{ (X-\mu_i)(X-\mu_i)^t \} , \ i=1,2 \qquad (2.3)$$

and S_w be the average class conditional covariance matrix,

$$S_w = \sum_{i=1}^{2} P_i S_i$$

Finally, let S_t designate the covariance matrix of the mixture distribution,

$$S_t = E \{ (X-\mu)(X-\mu)^t \}$$

S_b and S_w are frequently called the between class scatter matrix and the within class scatter matrix. In the one dimensional projected space of $z = V^tX$, one can easily show that the between class scatter s_b and the within class scatter s_w are expressed as,

$$s_b = V^tS_bV \tag{2.4}$$

$$s_w = V^tS_wV \tag{2.5}$$

Then, the Fisher discriminant ratio is defined as,

$$F(V) = s_b / s_w = V^tS_bV \; / \; V^tS_wV \tag{2.6}$$

When using the Fisher discriminant ratio, we seek to compute an optimum direction V, such that orthogonally projected samples are maximally discriminated. The optimum direction V can be found by taking the derivative of equation (2.6) and set to zero as,

$$\nabla F(V) = (V^t S_w V)^{-2}\{ 2S_b V V^t S_w V - 2V^t S_b V S_w V \} = 0$$

From the above equation, it follows that

$$V = K\, S_w^{-1}(\mu_1 - \mu_2) \qquad (2.7)$$

where K is a constant. The threshold v_{d+1} is that value which maximally discriminates between two classes.

The design procedure of the binary tree classifier using the Fisher linear discriminant function is as follows;

At each nonterminal node n, let $\Omega^n = \{ \omega_c^n ; c=1,\ldots,NC \}$ be the set of classes to be classified at node n and NC is the number of possible classes. Let N_c^n be the number of pixels (training samples) for class c in node n and $N^n = \sum_{c=1}^{NC} N_c^n$ be the total number of training samples in node n. The set Ω^n is successively partitioned into all possible partitions having two groups of classes, class LEFT and class RIGHT. For each partition, the weighting vector V of Fisher's linear discriminant function is computed using the set of training samples contained in that node, $X^n = \{ X_k^n ; k = 1,\ldots,N^n \}$, and the weighting vector V is normalized such that $|V| = 1$.

Using the weighting vector V, $f(X_k^n) = V^t X_k^n$ is computed for all $k = 1,\ldots,N^n$. The decision rule to be applied in node

n is specified as follows: If $f(X_k^n)$ is less than or equal to v_{d+1}, then X_k^n is in class LEFT, otherwise X_k^n is in class RIGHT. Now, we vary the threshold from the minimum to the maximum value of $f(X_k^n)$ in a step of 1. For each value, X_k^n is classified using the decision rule specified above and the number of samples n_{Lc} and n_{Rc} are counted, where n_{Lc} represents the number of samples X_k^n classified as LEFT and whose true class is ω_c, and n_{Rc} represents the number of samples X_k^n classified as RIGHT and whose true class is ω_c.

Let n_L be the total number of samples classified as LEFT and n_R be the total number of samples classified as RIGHT, i.e.

$$n_L = \sum_{c=1}^{NC} n_{Lc}$$

$$n_R = \sum_{c=1}^{NC} n_{Rc}$$

Then, the purity PR of the child nodes of n is defined as

$$PR = \sum_{c=1}^{NC} p_{Lc} \log p_{Lc} + \sum_{c=1}^{NC} p_{Rc} \log p_{Rc} \qquad (2.8)$$

where

$$p_{Lc} = n_{Lc} / n_L$$

$$p_{Rc} = n_{Rc} / n_R$$

The threshold is selected such that it maximizes the purity value PR as defined in (2.8).

The purity in (2.8) is designed such that it gives maximum value when the training samples are completely separable. For example, consider a nonterminal node with three classes in the training samples with equal number of samples for each class. Then, the purity of this nonterminal node is -3(log3)/3 = -log3. If the selected decision rule separates the training samples such that the LEFT child contains one class and the RIGHT child contains the other two classes, the purity of the child nodes is 0 - 2(log2)/2 = -log2. But in the worst case where both the LEFT and the RIGHT child contains the same number of samples for each class, the purity of the child node is -3(log3)/3 - 3(log3)/3 = -2log3. Thus we can easily see that the purity value of the former case where the training samples are completely separable is greater than the purity value of the latter case where the training samples are not separable.

For every possible grouping of the set Ω^n, the weighting vector V is computed together with the threshold v_{d+1} that maximizes the purity of the child nodes. Among those threshold values obtained for all groupings of the set Ω^n, the one with the maximum purity is selected, and the weighting vector

associated with that threshold is chosen as a decision function at that node.

In the process of selecting a threshold and constructing the decision tree, there are still two more questions to be answered.

First, does the decision rule obtained at a nonterminal node n separate the sample space X^n into the one it was designed to be ? To answer this question, we first define type I error and type II error as follows. Let type I error be the probability that a pixel whose true class is LEFT is classified as class RIGHT, and type II error be the probability that a pixel whose true class is RIGHT is classified as class LEFT. Then, if the sample space is completely separable, we would get zero for both type I and type II error. Since this is not always the case, we control these errors by considering only those thresholds in the process of selecting threshold that give type I error less than ε_I and type II error less than ε_{II} where ε_I and ε_{II} are pre-determined values before we start constructing the decision tree.

The second question is "When do we stop generating a decision tree ?". First of all, it is not reasonable to generate a decision tree which has more terminal nodes than the total number of training samples. Using this consideration

as a starting point, we set the maximum level of the decision tree to be $\log_2(n^1) - 1$, which makes the number of terminal nodes less than $n^1/2$, where n^1 is the number of training samples in node 1, root node. Next, in the process of expanding a nonterninal node, if we cannot find a decision rule that gives type I error less than ε_I and type II error less than ε_{II}, which means that the sample space is not separable in ε_I and ε_{II} error level, we stop expanding this nonterminal node. From the training samples of this node, the class which is most probable is assigned to be the label of this node.

This process of decision tree contruction is repeated until there is no nonterminal node left or the level of the decision tree reaches the maximum level. When we reach the maximum level, all the remaining nonterminal nodes are assigned a class which is the most probable class for its associated training samples.

CHAPTER 3. CONTEXTUAL CLASSIFIER

The most desirable kind of labeling process would give each pixel the highest probability label given the entire context of the image. The next most desirable kind of labeling process would give each pixel the highest probability label given some substantially sized context neighboring the pixel. In this chapter the theory for such a contextual classifier is presented.

The two pass algorithm given in this chapter takes the form of a recursive neghborhood operator first applied in a top down scan of the image and then in a bottom up scan of the image. The algorithm itself is related to a forward dynamic programming algorithm put in a two dimensional mesh setting. To explain the meaning of what the algorithm produces, select any pixel in the image. Now consider all the row monotonically increasing paths which begin at any border pixel of the image above the selected pixel, go through the selected pixel, and end at some bottom pixel of the image below the selected pixel. Each such path represents a context for the pixel. Corresponding to each path and the observed pixel data on the path, there is an associated highest probability label for the given pixel. Among all the paths there is some best path whose associated highest probability

label is higher than the highest probability label of every other path. In two scans of the image, the context algorithm is able to assign to each pixel of the image the highest probability label coming from its best path.

The theory for the algorithm requires two distinct ideas. The first idea produces a decomposition for the problem. Finding the highest probability label given the best path passing through the pixel can be accomplished by finding two probabilities, the probability for each possible label given the best path beginning above the pixel and terminating at the pixel and the probability for each possible label given the best path beginning at the pixel and terminating below the pixel. Finding these probabilities is what the algorithm accomplishes in top down scan and the bottom up scan. The decomposition tells how to combine these probabilities to determine the highest probability label given the context of the best path through the pixel.

The second idea produces a recursive decomposition which tells how to determine the conditional probability for each label given the data on the pixel's best upper (or lower) path from this some kind of conditional probability of the pixel's neighbors which have already been processed. The decomposition bears a definite similarity to the one used in forward dynamic programming and as well bears some similarity

to the iteration technique employed in some relaxation methods.

We now present basic concepts and notation conventions.

A path means any connected sequence of pixels, each pixel neighboring its successor, in which the path does not intersect itself. A row monotonically increasing path is a path in which each successor pixel is on the same row or one row below its predecessor.

The set U_{rc} designates the set of all row monotonically increasing paths which begin at some border pixel of the image above or to the left of pixel (r,c) and terminate at pixel (r,c) and not containing any pixels on row r beyond column c where (r,c) means the location of the pixel in the image at row r and column c.

The set L_{rc} designates the set of all row monotonically increasing paths which begin at (r,c) and terminate at some border pixel below or to the right of pixel (r,c). These are illustrated in Figure 2.

The set Z_{rc} designates the set of all row monotonically increasing paths beginning from a border of the image passing through pixel (r,c) and continuing to another border pixel of the image. The relationship between Z_{rc} and U_{rc} and L_{rc} should be obvious. Z_{rc} is just the join of all paths in U_{rc} with the paths in L_{rc}.

The set U^*_{rc} designates the set of all row monotonically increasing paths which begin at some border pixel of the image at the same row or above pixel (r,c) and terminate at pixel (r,c). The main difference between the set U^*_{rc} and the set U_{rc} is that the set U_{rc} does not include paths which contain any pixels on the same row r beyond column c while the set U^*_{rc} does.

The set L^*_{rc} designates the set of all row monotonically increasing paths which begin at pixel (r,c) and terminate at some border pixel at the same row or below pixel (r,c). While the set L_{rc} does not include paths which contain any pixels on the same row r before column c, the set L^*_{rc} does include those paths. This is illustrated in Figure 3.

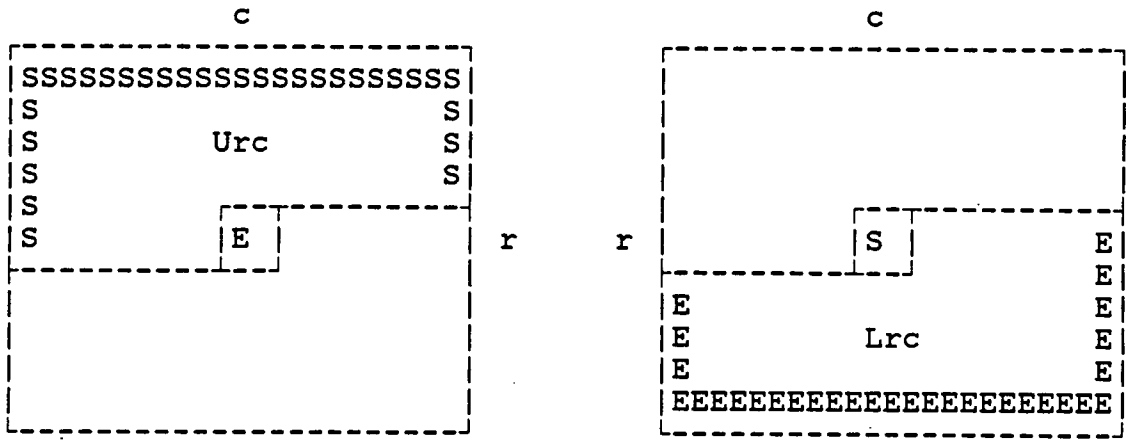

Figure 2. The set U_{rc} and L_{rc}: U_{rc} is the set of all row monotonically increasing paths beginning on a border of the image above or to the left of the pixel (r,c) and terminating at the pixel (r,c) and not containing any pixel on row r beyond column c. L_{rc} is the set of all row monotonically increasing paths begining at the pixel (r,c) and terminating on the border of the image below or to the right of the pixel (r,c). S and E designate the possible starting and ending pixels of a path.

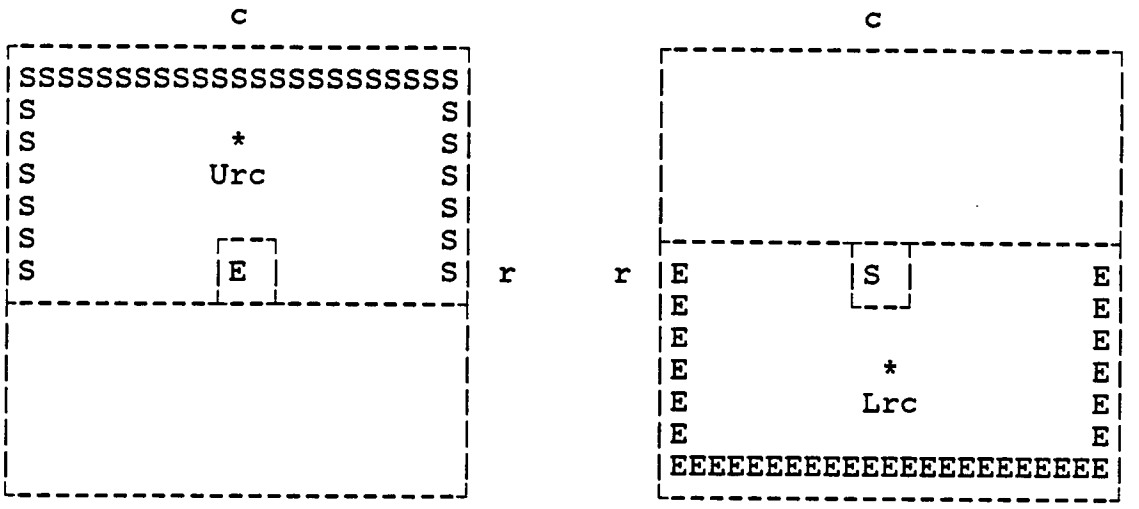

Figure 3. The set U^*_{rc} and L^*_{rc}: U^*_{rc} is the set of all row monotonically increasing paths beginning on a border of the image at the same row or above pixel (r,c) and terminating at pixel (r,c). L^*_{rc} is the set of all row monotonically increasing paths beginning at the pixel (r,c) and terminating at some border pixel at the same row or below pixel (r,c). S and E designate the possible starting and ending pixels of a path.

For pixel (i,j), let X_{ij} designate the measurements of the pixel (i,j). Let Q be any path. $P(X_{ij}\ ;\ (i,j) \in Q)$ designates the joint probability of all the measurements taken from the pixels on the path Q. For pixel (r,c), let e_{rc} designate the correct but unknown label at pixel (r,c). Then $P(e_{rc},\ X_{ij}\ ;\ (i,j) \in Q)$ designates the joint probability that the pixel (r,c) takes the label e_{rc} and the joint measurements made for the pixels on the path Q are

$\{\ X_{ij}\ ;\ (i,j) \in Q\ \}$.

$f_{Z_{rc}}(e_{rc})$ is defined to be the probability that pixel (r,c) takes label e_{rc} and that the joint measurements on the best row monotonically increasing path Q through (r,c) is

$\{\ X_{ij}\ ;\ (i,j) \in Q\ \}$.

Thus

$$f_{Z_{rc}}(e_{rc}) = \max_{Q \in Z_{rc}} P(e_{rc}, X_{ij}\ ;\ (i,j) \in Q) \qquad (3.9)$$

Just as $f_{Z_{rc}}(e_{rc})$ designates the probability of label e_{rc} and joint measurements $\{\ X_{ij}\ ;\ (i,j) \in Q\ \}$ arising from the best row monotonically increasing path Q in Z_{rc}, let $g_{U_{rc}}(e_{rc})$ designate the probability of label e_{rc} and joint measurements $\{\ X_{ij}\ ;\ (i,j) \in Q\ \}$ arising from the best row monotonically increasing path in U_{rc} and let $g_{L_{rc}}(e_{rc})$ designate the probability of label e_{rc} and joint measurements $\{\ X_{ij}\ ;\ (i,j) \in Q\ \}$ arising from the best row monotonically increasing path in L_{rc}.

It is easy to demonstrate the conditions under which $f_{Z_{rc}}(e_{rc})$ has the decomposition.

$$f_{Z_{rc}}(e_{rc}) = g_{U_{rc}}(e_{rc})\, g_{L_{rc}}(e_{rc}) \,/\, P(X_{rc}|e_{rc}) \qquad (3.10)$$

This decomposition requires some assumptions on the joint probability of the labels and the measurements and the joint probability of the labels.

By Bayes formula,

$$P(e_{ij},X_{ij};(i,j) \in Q) = P(X_{ij};(i,j) \in Q|e_{ij};(i,j) \in Q) \times P(e_{ij};(i,j) \in Q)$$

Assuming that the measurements at each pixel depend only on the true label at that pixel and measurement noise for one pixel does not influence the measurement noise for another pixel, we have

$$P(X_{ij};\ (i,j) \in Q|e_{ij};\ (i,j) \in Q) = \prod_{(i,j)\in Q} P(X_{ij}|e_{ij}) \qquad (3.11)$$

The probability $P(e_{ij}\ ;\ (i,j) \in Q)$ is the joint prior probability of having the true labels for each pixel (i,j) on the path Q be e_{ij}.

This probability encodes all the information we have about context. If, for example, we had independence,

$$P(e_{ij};\ (i,j) \in Q) = \prod_{(i,j)\in Q} P(e_{ij})$$

Thus we would discover that the highest probability assignment we could make using the context is precisely the highet probability assignment we could make using only the local information.

The simplest assumption of higher order independence is a Markov like assumption in which the joint prior probability becomes a function expressible as the product of functions whose arguments are the label pairs for succesive pixels in the path Q. This is a second order generalized conditional independence assumption. Letting R(Q) designate the set of all pairs of succesive pixels in the path Q, we have,

$$P(e_{ij};(i,j) \in Q) = \prod_{((i,j),(k,l))\in R(Q)} A(e_{ij},e_{kl}) \qquad (3.12)$$

Using assumptions (3.11) and (3.12) it is easy to demontrate the decomposition (3.10).

$$f_{Z_{rc}}(e_{rc}) = \max_{Q \in Z_{rc}} P(e_{rc}, X_{ij};\ (i,j) \in Q)$$

$$= \max_{Q \in Z_{rc}} \sum_{\substack{e_{ij} \\ (i,j) \in Q^-}} P(e_{ij}, X_{ij};\ (i,j) \in Q) \qquad (3.13)$$

where Q^- designates the set of all pixels in Q except the pixel (r,c) and the summation over all e_{ij} where $(i,j) \in Q^-$ designates an iterated summation, one sum for each pixel $(i,j) \in Q^-$, the sum taken over all possible values for the label the pixel (i,j) can take.

Substituting (3.11) and (3.12) into (3.13) we have,

$$f_{Z_{rc}}(e_{rc}) = \max_{Q \in Z_{rc}} \sum_{\substack{e_{ij} \\ (i,j) \in Q^-}} \prod_{(i,j) \in Q} P(X_{ij} | e_{ij}) \prod_{((i,j),(k,l)) \in R(Q)} A(e_{ij}, e_{kl})$$

Since Z_{rc} can be decomposed as the join of U_{rc} and L_{rc}, there results,

$$f_{Z_{rc}}(e_{rc}) = \max_{Q_1 \in U_{rc}} \max_{Q_2 \in L_{rc}} \sum_{\substack{e_{ij} \\ (i,j) \in Q_1^-}} \sum_{\substack{e_{ij} \\ (i,j) \in Q_2^-}}$$

$$\{ \prod_{(i,j)\in Q_1} P(X_{ij}|e_{ij}) \prod_{(i,j)\in Q_2} P(X_{ij}|e_{ij})/P(X_{rc}|e_{rc})$$

$$\prod_{((i,j),(k,l))\in R(Q_1)} A(e_{ij},e_{kl}) \prod_{((i,j),(k,l))\in R(Q_2)} A(e_{ij},e_{kl})\}$$

(3.14)

Rearranging (3.14) we can group all expressions involving Q_1 together and all expressions involving Q_2 together and we obtain,

$$f_{Z_{rc}}(e_{rc}) = \frac{1}{P(X_{rc}|e_{rc})}$$

$$\max_{Q_1\in U_{rc}} \sum_{\substack{e_{ij} \\ (i,j)\in Q_1^-}} \prod_{(i,j)\in Q_1} P(X_{ij}|e_{ij}) \prod_{((i,j),(k,l))\in R(Q_1)} A(e_{ij},e_{kl})$$

$$\max_{Q_2\in U_{rc}} \sum_{\substack{e_{ij} \\ (i,j)\in Q_2^-}} \prod_{(i,j)\in Q_2} P(X_{ij}|e_{ij}) \prod_{((i,j),(k,l))\in R(Q_2)} A(e_{ij},e_{kl})$$

$$= \frac{1}{P(X_{rc}|e_{rc})} \; g_{U_{rc}}(e_{rc})\, g_{L_{rc}}(e_{rc})$$

The decomposition of $g_{U_{rc}}$ will be in terms of the neighboring $g_{U_{rc-1}}$ and $h^*_{U_{r-1c-1}}$, $h^*_{U_{r-1c}}$, and $h^*_{U_{r-1c+1}}$ where the h_U^* will be defined in the next paragraph.

By definition

$$g_{U_{rc}}(e_{rc}) = \max_{Q \in U_{rc}} P(e_{rc}, X_{ij};\ (i,j) \in Q)$$

$$= \max_{Q \in U_{rc}} \sum_{\substack{e_{ij} \\ (i,j) \in Q^-}} P(e_{ij}, X_{ij};\ (i,j) \in Q)$$

$$= \max_{Q \in U_{rc}} \sum_{e_{ij}} \prod_{(i,j) \in Q^-} P(X_{ij} | e_{ij}) \prod_{((i,j),(k,l)) \in R(Q)} A(e_{ij}, e_{kl}) \qquad (3.15)$$

Just as $g_{U_{rc}}$ designates the probability of label e_{rc} and joint measurements $\{ X_{ij} ;\ (i,j) \in Q \}$ arising from the best rwo monotonically increasing path in U_{rc} let $h^*_{U_{rc}}$ designate the probability of label e_{rc} and joint measurements $\{ X_{ij} ;\ (i,j) \in Q \}$ arisinig from the best row monotonically increasing path in U^*_{rc}.

Then, by definition

$$h_{U^*_{rc}}(e_{rc}) = \max_{Q \in U^*_{rc}} P(e_{rc}, X_{ij};\ (i,j) \in Q)$$

$$= \max_{Q \in U^*_{rc}} \sum_{\substack{e_{ij} \\ (i,j) \in Q^-}} P(e_{ij}, X_{ij};\ (i,j) \in Q)$$

$$= \max_{Q \in U^*_{rc}} \sum_{\substack{e_{ij} \\ (i,j) \in Q^-}} \prod_{(i,j) \in Q} P(X_{ij} | e_{ij}) \prod_{((i,j),(k,l)) \in R(Q)} A(e_{ij}, e_{kl}) \qquad (3.16)$$

To do the decomposition for $g_{U_{rc}}$ we need to recognize that whoever the best path is, the best path to (r,c) must have come from (r,c-1), (r-1,c-1), (r-1,c), or (r-1,c+1). Because the best path cannot cross itself, if the best path came from (r,c-1) then the path must be in U_{rc-1}. However, there is no damage of the path crossing itself if the best path comes from (r-1,c-1). Hence, such a path must be in U^*_{r-1c-1}. Likewise, a best path comming from (r-1,c) must be in U^*_{r-1c} and a best path comming from (r-1,c+1) must be in U^*_{r-1c+1}. This is illustrated in Figure 4.

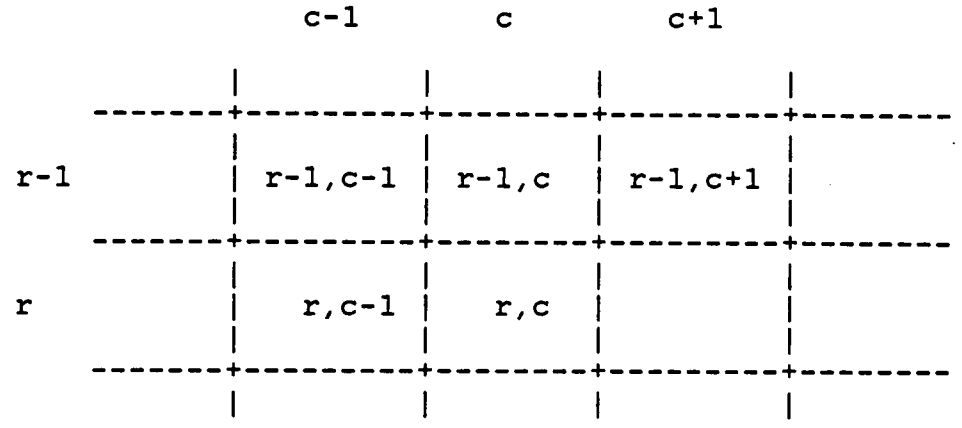

Figure 4. The decomposition for $g_{U_{rc}}$: The best path to (r,c) must have come from (r,c-1), (r-1,c-1), (r-1,c), or (r-1,c+1) where (r,c) means the location of the pixel in the image at row r and column c.

Using this idea, we can rewrite (3.15) as

$$g_{U_{rc}}(e_{rc}) = P(X_{rc}|e_{rc}) \max \{ \max_{Q \in U_{rc-1}} W\, A(e_{rc-1}, e_{rc}),$$

$$\max^{*}_{Q \in U_{r-1c-1}} W\, A(e_{r-1c-1}, e_{rc}),$$

$$\max^{*}_{Q \in U_{r-1c}} W\, A(e_{r-1c}, e_{rc}),$$

$$\max^{*}_{Q \in U_{r-1c+1}} W\, A(e_{r-1c+1}, e_{rc}) \} \qquad (3.17)$$

where

$$W = \sum_{\substack{e_{ij} \\ (i,j) \in Q}} \prod_{(i,j) \in Q} P(X_{ij}|e_{ij}) \prod_{((i,j),(k,l)) \in R(Q)} A(e_{ij}, e_{kl})$$

Examining the first term in the maximization and comparing it to $g_{U_{rc-1}}(e_{rc-1})$ as defined by (3.15) we discover that the expressions are almost identical. The only difference is that the expression for $g_{U_{rc-1}}(e_{rc-1})$ involves a summation iterated over all $(i,j) \in Q^{-}$ while the first expression in the maximization involves a summation iterated over all

$(i,j) \in Q$. Since the maximization done in the first term is over all paths terminating in (r,c-1), the summation over e_{rc-1} can be interchanged with the maximization over all $Q \in U_{rc-1}$.

A similar reorganization can be done with the second, third, and fourth terms of the summation after comparing them with $h^*_{U_{r-1c-1}}$, $h^*_{U_{r-1c}}$, $h^*_{U_{r-1c+1}}$. This results in

$$g_{U_{rc}}(e_{rc}) = P(X_{rc}|e_{rc}) \max\{ \sum_{e_{rc-1}} \max_{Q \in U_{rc-1}} W\, A(e_{rc-1}, e_{rc}),$$

$$\sum_{e_{r-1c-1}} \max^*_{Q \in U_{r-1c-1}} W\, A(e_{r-1c-1}, e_{rc}),$$

$$\sum_{e_{r-1c}} \max^*_{Q \in U_{r-1c}} W\, A(e_{r-1}, e_{rc}),$$

$$\sum_{e_{r-1c+1}} \max^*_{Q \in U_{r-1c+1}} W\, A(e_{r-1c-1}, e_{rc})\} \qquad (3.18)$$

where

$$W = \sum_{e_{ij}} \prod_{(i,j) \in Q} P(X_{ij}|e_{ij}) \prod_{((i,j),(k,l)) \in R(Q)} A(e_{ij}, e_{kl})$$

$$(i,j) \in Q^-$$

Upon direct substitution of (3.16) and (3.17) into (3.18) we obtain

$$g_{U_{rc}}(e_{rc}) = P(X_{rc}|e_{rc}) \times$$

$$\max \{ \sum_{e_{rc-1}} g_{U_{rc-1}}(e_{rc-1})\, A(e_{rc-1}, e_{rc}),$$

$$\sum_{e_{r-1c-1}} h^*_{U_{r-1c-1}}(e_{r-1c-1})\, A(e_{r-1c-1}, e_{rc}),$$

$$\sum_{e_{r-1c}} h^*_{U_{r-1c}}(e_{r-1c})\, A(e_{r-1c}, e_{rc}),$$

$$\sum_{e_{r-1c+1}} h^*_{U_{r-1c+1}}(e_{r-1c+1}) A(e_{r-1c+1}, e_{rc})\} \qquad (3.19)$$

Equation (3.19) says that for each label value e_{rc}, the joint probability of e_{rc} and $\{ X_{ij} ; (i,j) \in Q \}$ where Q is the path giving the highest joint probability can be obtained on the basis of the just previously computed $g_{U_{rc-1}}$ and on the previously computed $h^*_{U_{r-1c-1}}$, $h^*_{U_{r-1c}}$, $h^*_{U_{r-1c+1}}$ coming from the row above the current row (see Figure 5).

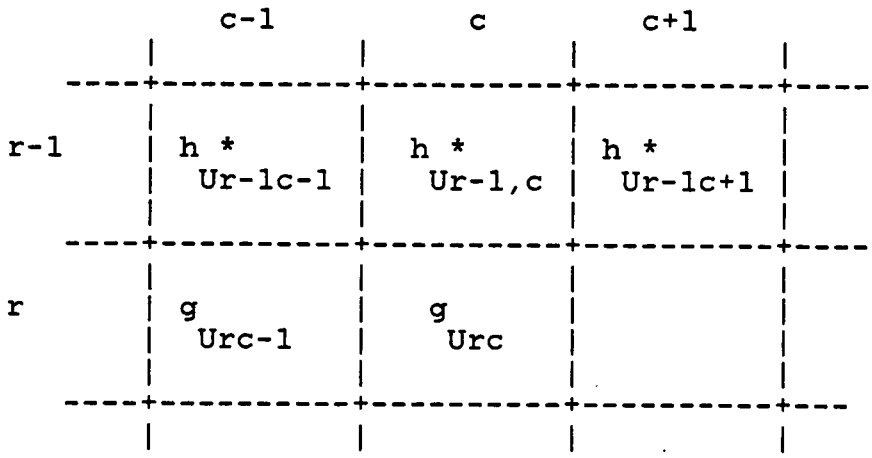

Figure 5. The computation of $g_{U_{rc}}$: Using h_U^* values computed in the row above the current row, $g_{U_{rc}}$ can be computed recursively from $h_{U_{r-1c-1}}^*$, $h_{U_{r-1c}}^*$, $h_{U_{r-1c+1}}^*$, and the just previously computed $g_{U_{rc-1}}$ in the left to right scan of the image.

So providing we can demonstrate a way to compute $h_{U_{rc}}^*$, equation (3.19) specifies a recurive neighborhood operator which scans the image in a top down left right scan to produce for each pixel (r,c) and for each label e_{rc}, $g_{U_{rc}}(e_{rc})$.

Fortunately, we are able to provide an algorithm for computing $h_{U_{rc}}^*$. Its development proceeds along similar lines to that of $g_{U_{rc}}$. For a path from U_{rc}^* to such (r,c), it must first have gone through (r,c-1), (r,c+1), (r-1,c-1), (r-1,c), or (r-1,c+1). Futhermore, if it went through (r,c-1), since the path cannot cross itself, it must be a path in U_{rc-1}. Hence,

$$
\begin{aligned}
h_{U_{rc}}^*(e_{rc}) = P(X_{rc}|e_{rc}) \max \{ & \max_{Q \in U_{rc-1}} W\, A(e_{rc-1}, e_{rc}), \\
& \max_{Q \in U_{rc+1}}^{*} W\, A(e_{rc+1}, e_{rc}), \\
& \max_{Q \in U_{r-1c-1}}^{*} W\, A(e_{r-1c-1}, e_{rc}), \\
& \max_{Q \in U_{r-1c}}^{*} W\, A(e_{r-1c}, e_{rc}), \\
& \max_{Q \in U_{r-1c+1}}^{*} W\, A(e_{r-1c+1}, e_{rc}) \}
\end{aligned}
\tag{3.20}
$$

where

$$W = \sum_{\substack{e_{ij} \\ (i,j)\in Q}} \prod_{(i,j)\in Q} P(X_{ij}|e_{ij}) \prod_{((i,j),(k,l))\in R(Q)} A(e_{ij},e_{kl})$$

From our development of equation (3.20), we know that the maximization over the first, third, fourth, and fifth terms of (3.20) yields $g_{U_{rc}}(e_{rc})$. Thus we have

$$h_{U_{rc}}^*(e_{rc}) = \max \{ g_{U_{rc}}(e_{rc}),\ P(X_{rc}|e_{rc}) \max_{Q\in U_{rc+1}^*} \sum_{\substack{e_{ij} \\ (i,j)\in Q}} \prod_{(i,j)\in Q} P(X_{ij}|e_{ij}) \prod_{((i,j),(k,l))\in R(Q)} A(e_{ij},e_{kl}) A(e_{rc+1},e_{rc}) \} \tag{3.21}$$

In a similar manner as in the development of (3.19), we interchange the order of the summation over e_{rc+1} with the maximization over all paths Q in U_{rc+1}^*. Now (3.21) becomes

$$h_{U_{rc}}^*(e_{rc}) = \max \{ g_{U_{rc}}(e_{rc}),\ P(X_{rc}|e_{rc}) \sum_{e_{rc+1}} A(e_{rc+1},e_{rc}) [\max_{Q\in U_{rc+1}^*} \sum_{\substack{e_{ij} \\ (i,j)\in Q}} \prod_{(i,j)\in Q} P(X_{ij}|e_{ij}) \prod_{((i,j),(k,l))\in R(Q)} A(e_{ij},e_{kl})] \} \tag{3.22}$$

By definition of (3.17), the bracketed [ ] expression of (3.22) is precisely $h_{U_{rc+1}}^*(e_{rc})$. Now this results in,

$$h_{U_{rc}}^*(e_{rc}) = \max \{ g_{U_{rc}}(e_{rc}), P(X_{rc}|e_{rc}) \sum_{e_{rc+1}} h_{U_{rc+1}}^*(e_{rc+1}) A(e_{rc+1}, e_{rc}) \} \quad (3.23)$$

Equation (3.23) states that $h_{U_{rc}}^*$ can be recursively computed from $g_{U_{rc}}$ and the previous $h_{U_{rc+1}}^*$ in a right left scan of a row done after $g_{U_{rc}}$ has been computed. To start the recursive calculation (3.23) we take $h_{U_{rc}}^*(e_{rc}) = g_{U_{rc}}(e_{rc})$ for that column position c which is the rightmost position.

In summary, equation (3.19) and (3.23) give the following algorithm for the computation of $g_{U_{rc}}(e_{rc})$. From (3.19) we perform a top down left right scan of the image recursively computing $g_{U_{rc}}$ from the previous $g_{U_{rc-1}}$ and the $h_{U_{r-1c-1}}^*$, $h_{U_{r-1c}}^*$, $h_{U_{r-1c+1}}^*$ which had been computed on the previous row. Following the computation of $g_{U_{rc}}$ for all pixels on row r, we perform a right left scan of row r using equation (3.23) to compute $h_{U_{rc}}^*$. An obvious mirror image derivation applies to $g_{L_{rc}}$. To compute it, we perform a bottom up right left scan of the image recursively computing $g_{L_{rc}}$ from the previous

$g_{L_{rc+1}}$ and the $h^*_{L_{r+1c-1}}$, $h^*_{L_{r+1c}}$, and $h^*_{L_{r+1c+1}}$ which had been computed on the previous row from the bottom up scan. Following the computation of $g_{L_{rc}}$ for all pixels on row r, we perform a left right scan of row r and compute $h^*_{L_{rc}}$.

As soon as $g_{L_{rc}}(e_{rc})$ has been computed, it can be combined with $g_{U_{rc}}(e_{rc})$ as given in equation (3.10) to compute $f_{Z_{rc}}(e_{rc})$. Then the label e_{rc} which maximizes $f_{Z_{rc}}(e_{rc})$ can be determined and pixel (r,c) labeled with this label.

CHAPTER 4. EXPERIMENTAL RESULTS AND DISCUSSION

To show the increase in classsification accuracy of the binary decision tree clsssifier and the contextual classifier, several experiments were performed on the images using the newly designed classifiers.

First, a simulated image is used to examine the accuracy improvement of the new classifiers as compared to the single stage Bayes classifier, given that the class conditional mean vectors and covariance matrices were known. Then, they are applied on real images to investigate their performance in more realistic case. We present one experiment on thermal image to compare the ability to detect an object in the image, two experiments on LANDSAT data to compare the classification accuracy, and two more experiments on thermal images.

The simulated image is generated from a real LANDSAT image. The real image consists of a subimage of digital remote sensing data collected by the LANDSAT MSS (multispectral sensor) which has known ground truth. This is a 151 × 151 subframe of MSS scenes of Roanoke VA taken on 13 April 1976 (Figure 20) which contains five ground cover classes,

Class 1 : Urban or built-up land

Class 2 : Agricultural land

Class 3 : Rangeland

Class 4 : Forest land

Class 5 : Water

It has four spectral bands with the ground truth data shown in Figure 6. Due to the small sample size, ground cover class 3 and 5 are not used in the simulated image.

From the sample image and the ground truth data, we first estimate the mean vectors μ_i and the covariance matrices Σ_i for each class i. Then a simulated image with the following characteristics is created [2]. (1) Each pixel in the simulated image represents the same class as in the ground truth data, (2) all classes have multivariate Gaussian distribution having the means and covariance matrices estimated from the sample image, (3) all pixels are class-conditionally independent of adjacent pixels.

Let μ and Σ be one of the estimated mean vector and covariance matrix of some class, then the simulated image can be generated as follows.

From the multivariate Gaussian assumption,

$$P(X|e) = (2\pi\Sigma)^{-n/2} \exp \{ -\frac{1}{2} (X-\mu)^t \Sigma^{-1} (X-\mu) \}$$

where n is the dimension of the feature space. Using orthogonal transformation, Σ^{-1} can be rewritten as U^tDU where U is orthogonal and D is a diagonal matrix.

$$P(X|e) = (2\pi\Sigma)^{-n/2}\exp\ \{\ -\ \frac{1}{2}\ (X-\mu)^t U^t DU\ (X-\mu)\ \}$$

Letting $Z = U(X-\mu)$, we get

$$P(Z|e) = (2\pi UD^{-1}U^t)^{-n/2}\ \exp\ \{\ -\ \frac{1}{2}\ Z^tDZ\ \}$$

Therefore, we can transform the random vector X into a random vector $Z = [Z_i]$, having independent components Z_i where $Z_i \sim N(0,1/d_i)$.

Thus, the components of the Z vector is produced by a Gaussian random number generator. The vector Z is then transformed into X by

$$X = U^tZ + \mu$$

The simulated image thus obtained is first classified by the Bayes classifier with known mean vectors and covariance matrices. Figure 7 shows the classified image with equal priors for each class and Figure 8 shows the classified image with correct and known priors.

The simulated image is again classified by the binary decision tree classifier and the result is shown in Figure 9. All pixels in the simulated image are used as training samples in generating the decision tree.

In the derivation of our contextual classifier algorithm, we used a Markov like assumption and expressed $P(e_{ij} ; (i,j) \in Q)$ as the product of functions whose arguments are the label pairs for succesive pixels in the path Q (3.12). Even though the contextual information is not used to its full extent due to the ignorance of some correlations between pixels in the path Q, the improvement in classification accuracy was superior as will be shown later. To compute the function $A(e_{ij},e_{kl})$ for the testing purpose in this paper, we assumed a Gaussian stationary two-dimensional process. Here stationary means that the correlation between pixels are position independent in the image. The function A is estimated for three directional pairs of pixels, horizontal, vertical, and diagonal. The function A is approximated by the frequency distribution of each pairs of labels in all three directions from the ground truth data. Assuming the same multi-dimensional normal distribution, the class conditional covariance matrices and mean vectors used in the Bayes classifier is used to compute $P(X_{rc}|e_{rc})$ in the contextual classifier together with the function A obtained by the

method described before. Figure 10 shows the result of the contextual classifier.

To examine the performance of each classifier when the image is corrupted by random noise, we added Gaussian normal noise $N(0,\sigma^2)$ to the simulated image. Then, the noisy image is classified by four classifiers, Bayes with equal priors, Bayes with correct priors, binary decision tree, and contextual classifier. The overall classification accuracy is measured as the ratio of the number of correctly classified pixels to the number of total classified pixels and is plotted as a function of the noise standard deviation in Figure 11.

It can be seen in Figure 11 that the contextual classifier is superior to any other classifier and that the binary decision tree classifier and the Bayes classifier with correct priors have similar performance. But in a situation when we do not know the correct priors, the Bayes classifier shows poor classification accuracy.

To see what happens before a node is split and after, we projected three class samples onto the first two features (bands). Figure 12 shows such a scattergram before the root node is split and Figure 13 and Figure 14 show the scattergrams of the left and right node after the root node is split. It can be seen from those figures that class 1

(symbol o) which was in class LEFT and class 2 (square) and 4 (triangle) which were in class RIGHT in the design process are separated in child nodes by the decision rule. In fact, number of pixels contained in the child nodes are,

	class 1	class 2	class 3
left child	8838	1832	215
right child	5528	3946	2442
root node	14366	5778	2657

Now we extend our test on the real data sets. The first image is an 8-12 micron thermal image of size 200 × 200 taken at Grafenwoehr, Germany which contains one object in the center (Figure 15). For the classification of this image, two texture features, entropy and the inverse difference moment, are measured first from the image [14]. The original measurement in the thermal image and the two texture features compose the feature space of the classification. Training samples are selected from the original image as shown in Figure 16 and assuming that the underlying class conditional probability $p(X|e)$ is multi-dimensional normal, class-conditional covariance matrices and mean vectors are estimated. Using a multi-dimensional normal distribution, we first classify the image by the Bayes classifier and the result is shown in Figure 17. Next, the image is classified by the binary decision tree classifier and the contextual

classifier which are shown in Figure 18 and 19. We can see the effect of the contextual classifier by noting the difference in the boundary lines of the classified object between Figure 17 and Figure 19. Compared to the smooth boundary lines in the result of the contextual classifier, Bayes classifier leaves uneven boundary lines.

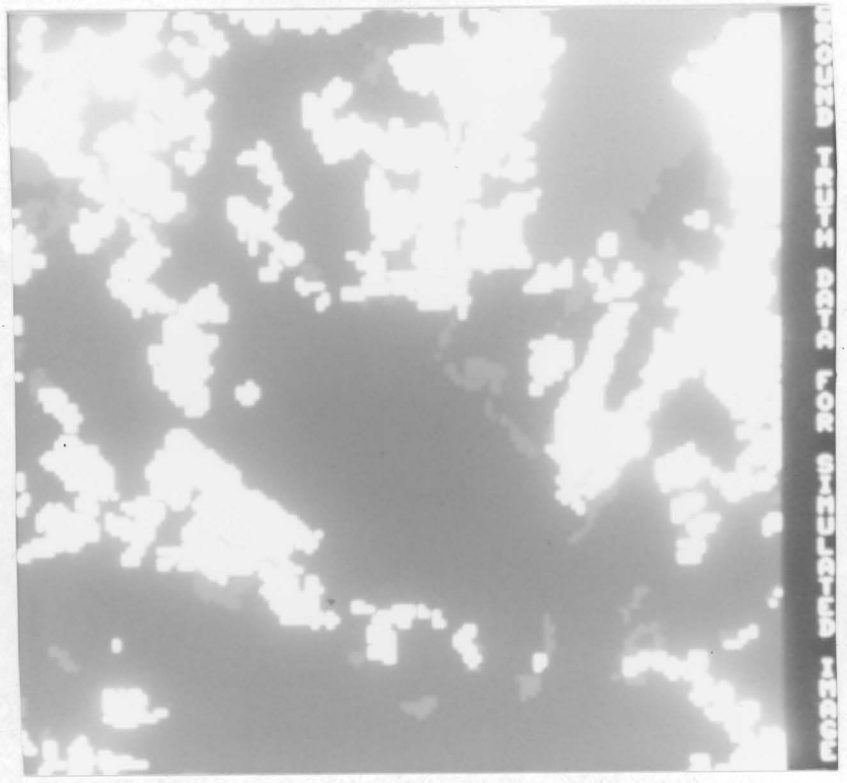

Figure 6. Ground truth data of the simulated image: Blue (class 1) - urban or built-up land, White (class 2) - agricultural land, Green (class 4) - forest land, Orange (class 7) - barren land.

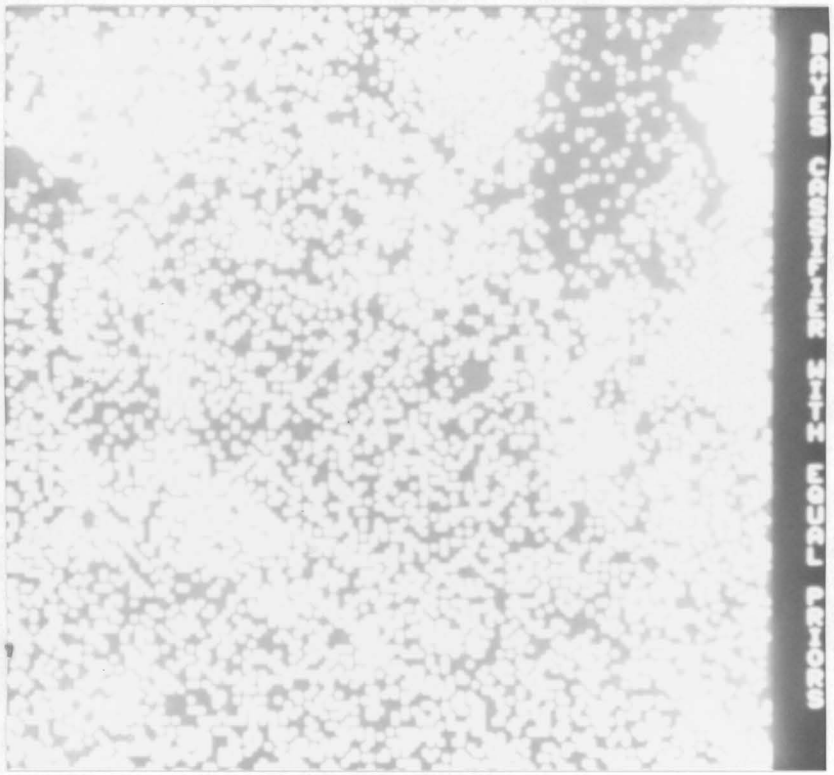

Figure 7. Classified result of simulated image - Bayes classifier: Classified by the Bayes classifier with equal priors. Blue (class 1) - urban or built-up land, White (class 2) - agricultural land, Green (class 4) - forest land.

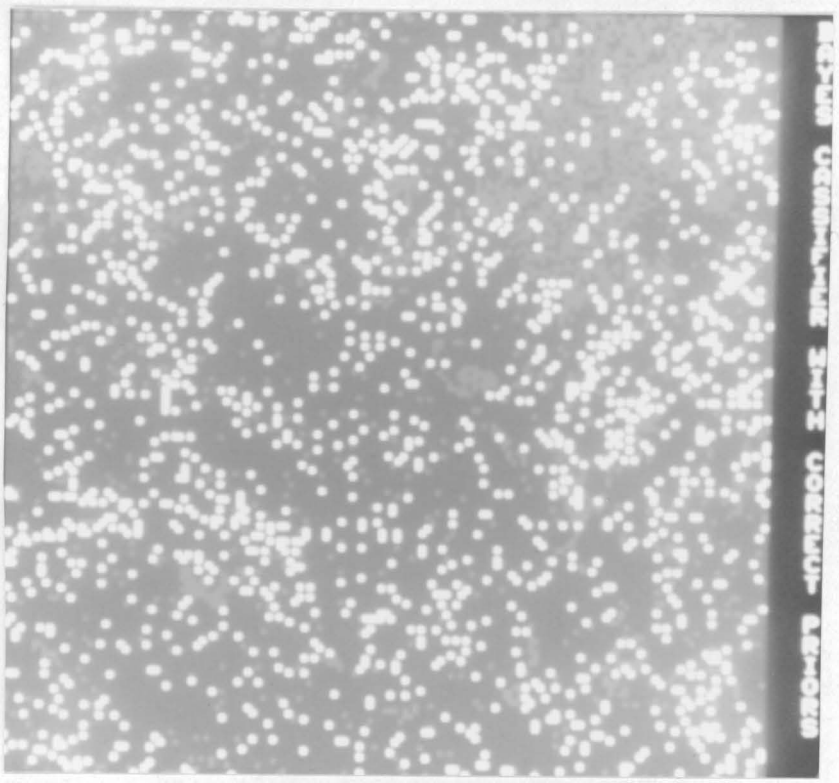

Figure 8. Classified result of simulated image - Bayes classifier: Classified by the Bayes classifier with correct priors. Blue (class 1) - urban or built-up land, White (class 2) - agricultural land, Green (class 4) - forest land.

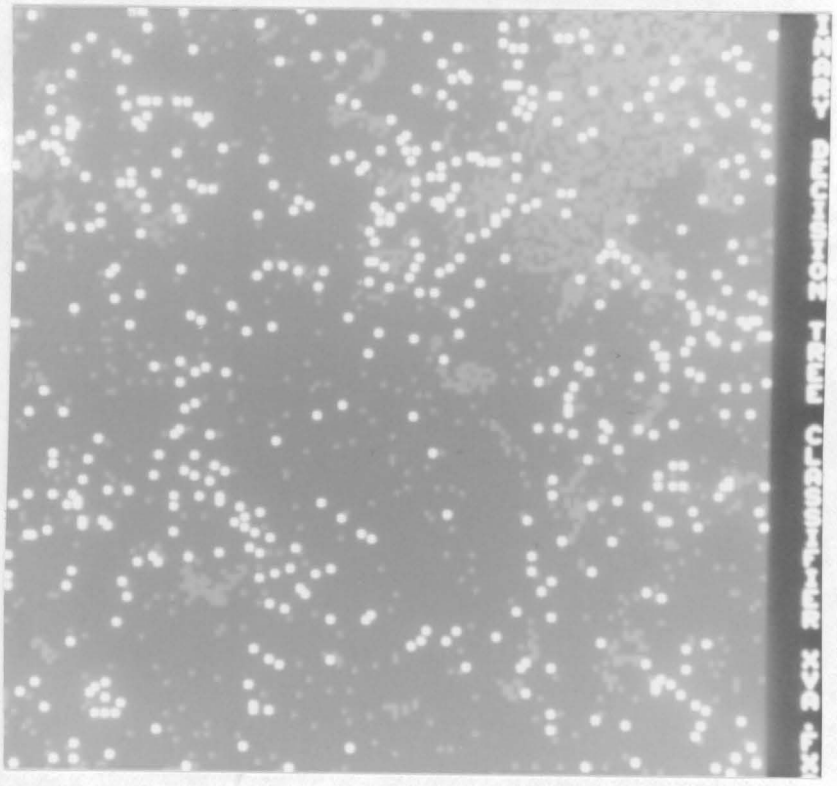

Figure 9. Classified result of simulated image - Tree classifier: Classified by the binary decision tree classifier. Blue (class 1) - urban or built-up land, White (class 2) - agricultural land, Green (class 4) - forest land.

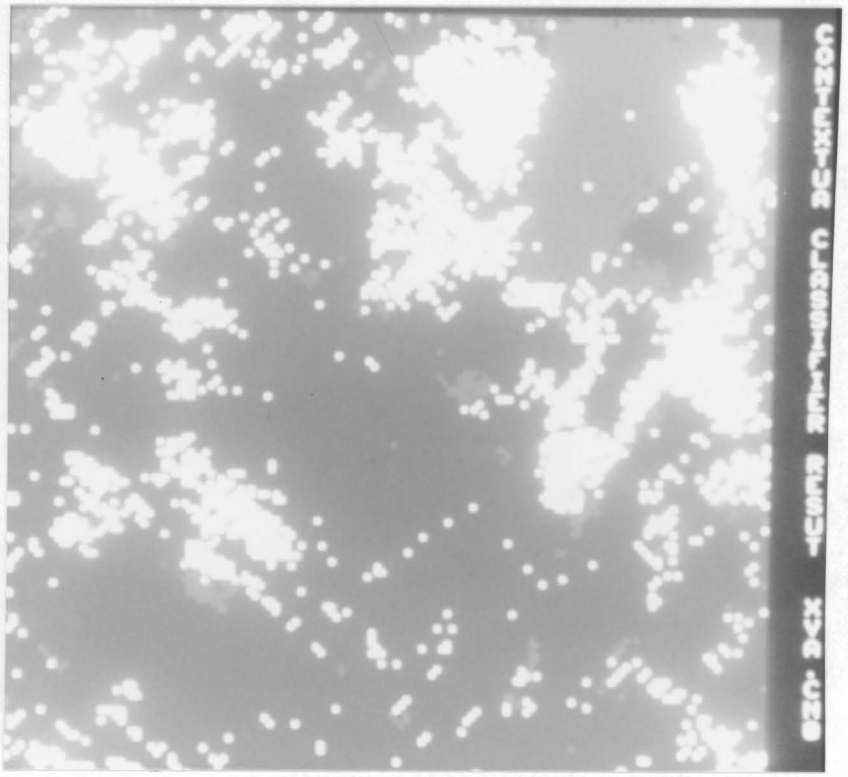

Figure 10. Classified result of simulated image - Contextual classifier: Classified by the contextual classifier. Blue (class 1) - urban or built-up land, White (class 2) - agricultural land, Green (class 4) - forest land.

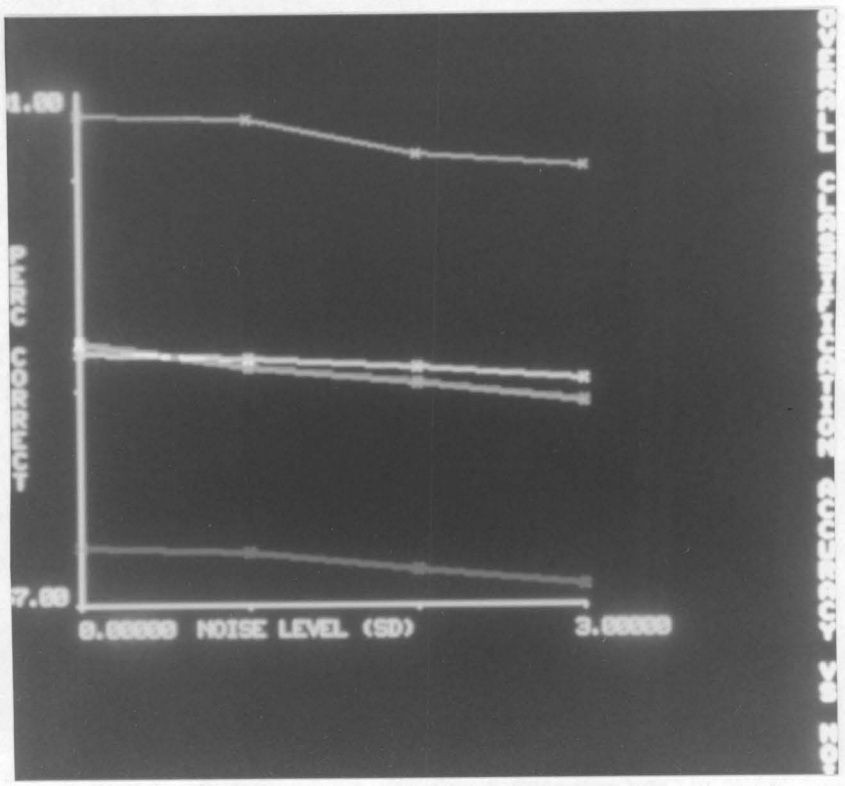

Figure 11. Overall classification accuracy curves vs noise level: Green line - contextual classifier, Yellow line - Bayes classifier with correct priors, Blue line - Binary decision tree classifier, Red line - Bayes classifier with equal priors.

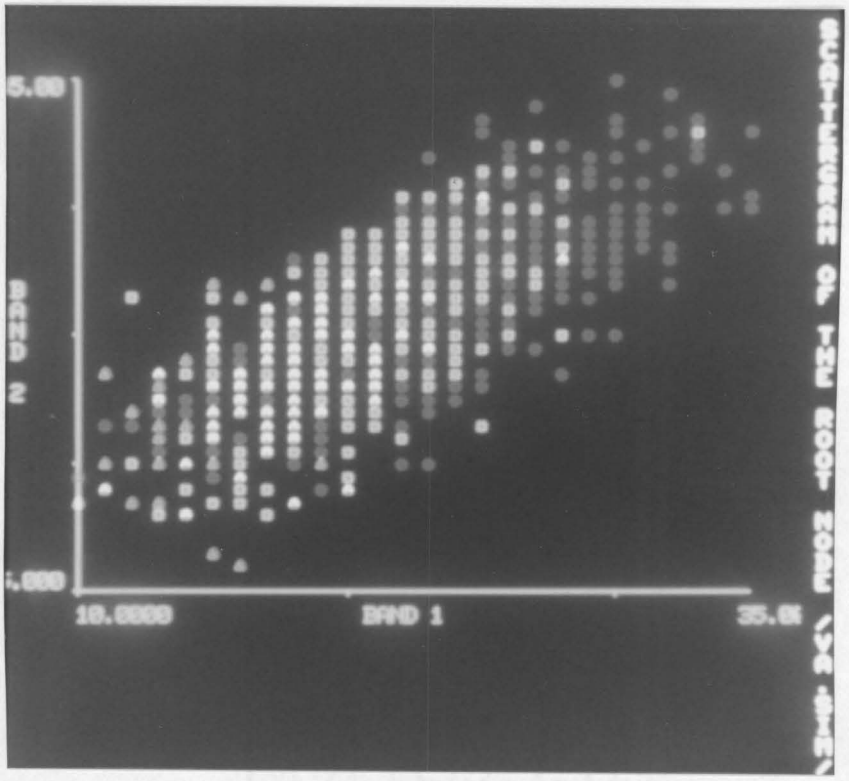

Figure 12. Scattergram of the root node of the tree classifier: Projection of 3 class samples onto the first two features of the simulated image. Only 4% of the samples are drawn in this scattergram. Circle (class 1) - urban or built-up land, Square (class 2) - agricultural land, Triangle (class 4) - forest land.

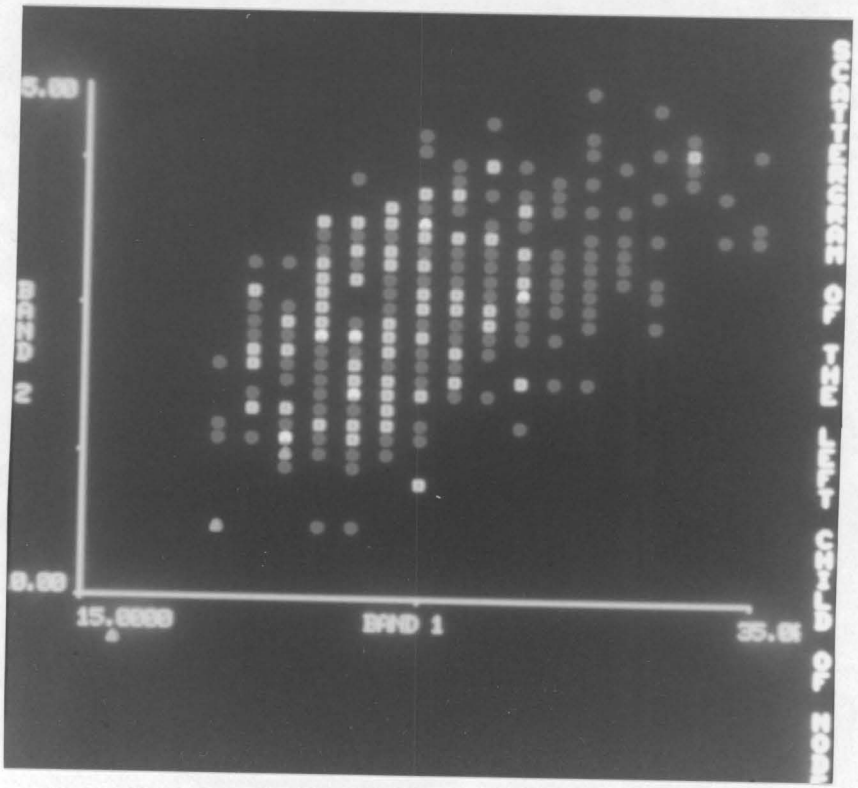

Figure 13. Scattergram of the left child of the root node: Projection of three class samples onto the first two features of the simulated image. Only 4% of the samples are drawn in this scattergram. Circle (class 1) - urban or built-up land, Square (class 2) - agricultural land, Triangle (class 4) - forest land.

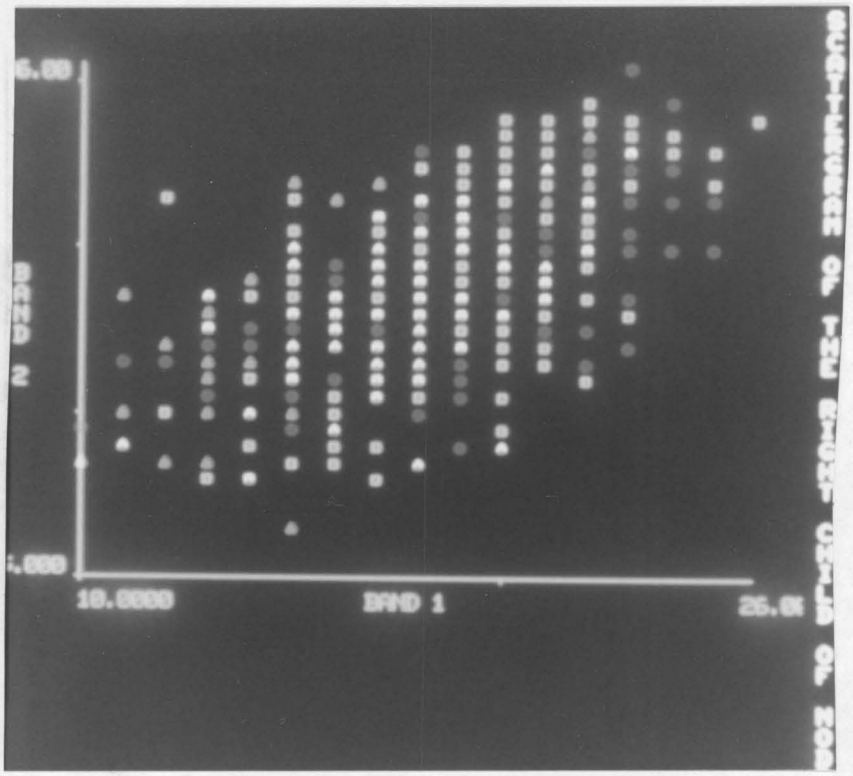

Figure 14. Scattergram of the right child of the root node: Projection of three class samples onto the first two features of the simulated image. Only 4% of the samples are drawn in this scattergram. Circle (class 1) - urban or built-up land, Square (class 2) - agricultural land, Triangle (class 4) - forest land

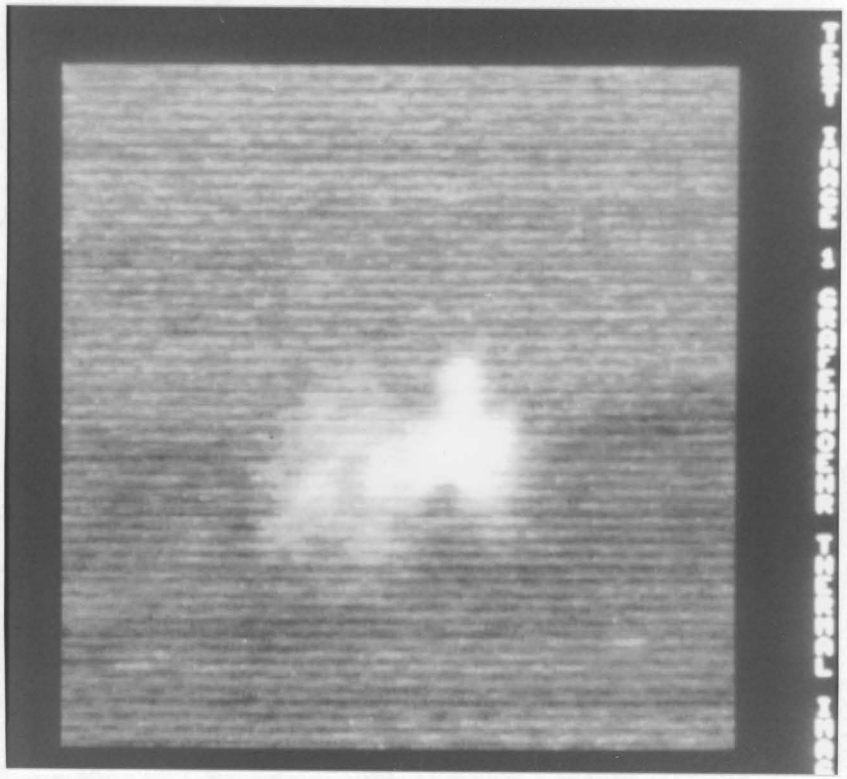

Figure 15. Test image 1: 8-12 micron thermal image of size 200 × 200 taken at Grafenwoehr, Germany. This image contains one object (white) at the center.

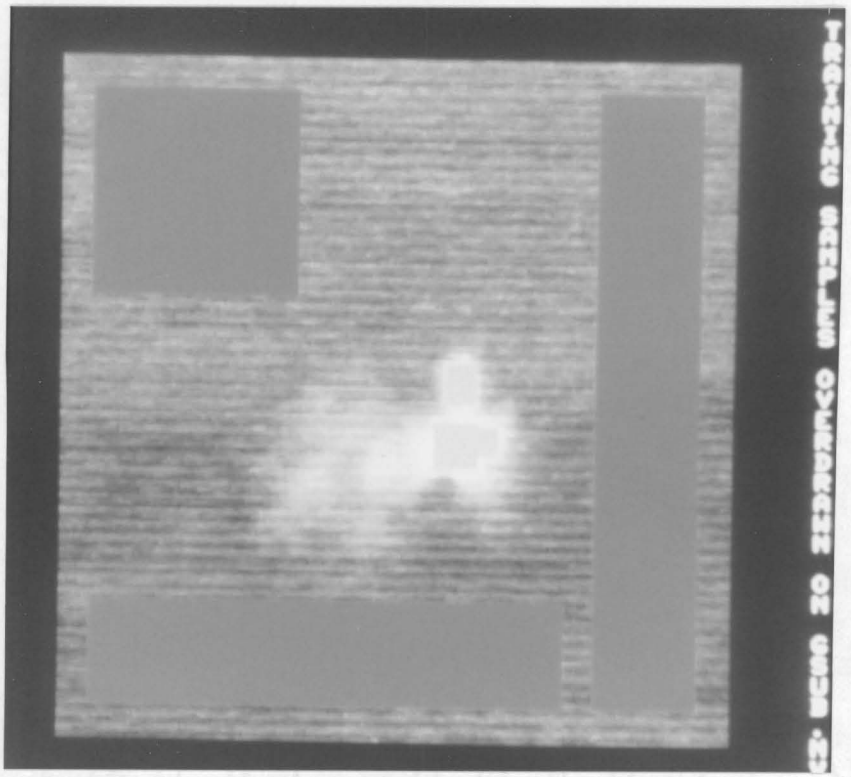

Figure 16. Training samples taken from test image 1: Ractangular boxes represent the selected training samples. Yellow - object, Brown - background.

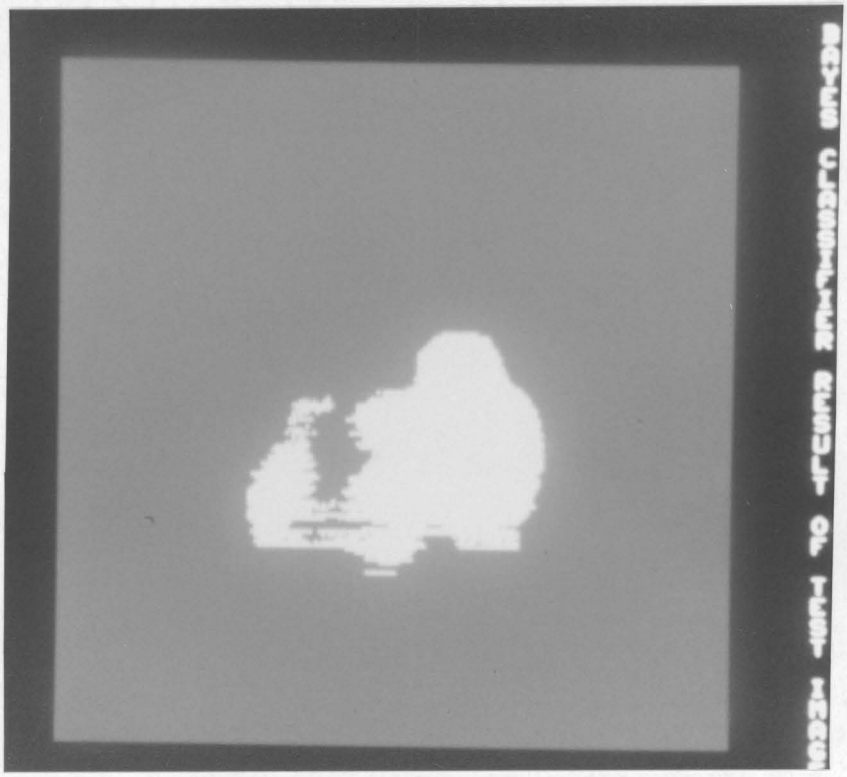

Figure 17. Classified result of test image 1 - Bayes classifier: Classified by the Bayes classifier with equal priors. Yellow - object, Brown - background.

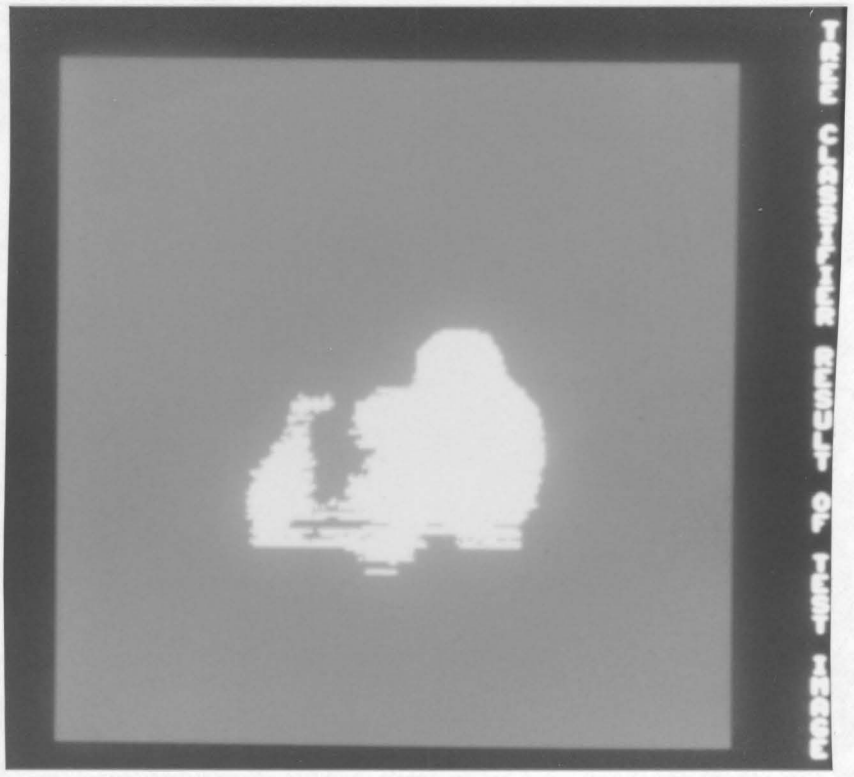

Figure 18. Classified result of test image 1 - Tree classifier: Classified by the binary decision tree classifier. Yellow - object, Brown - background.

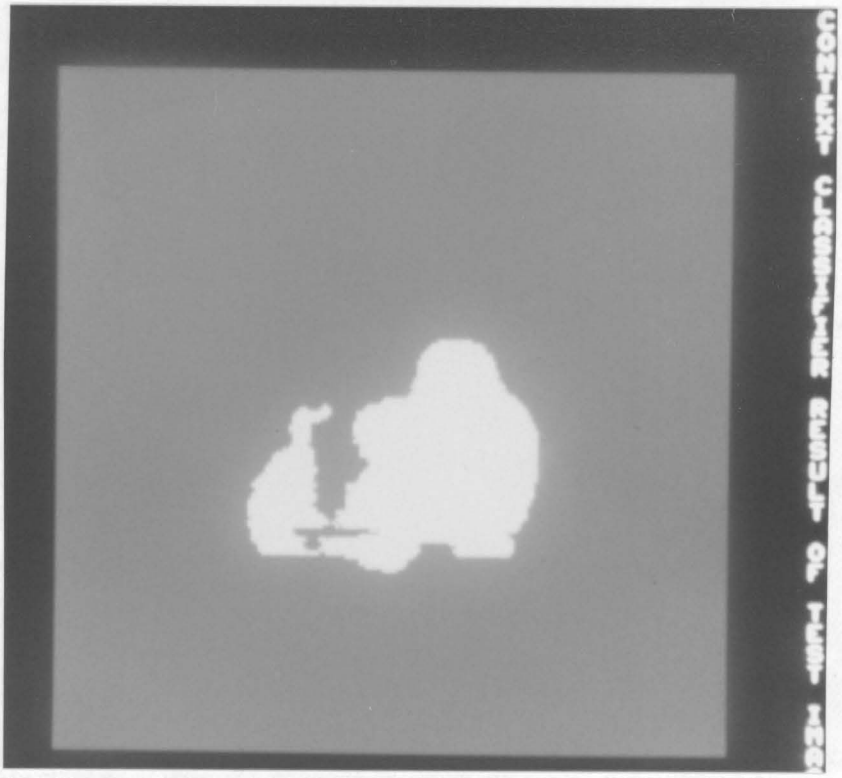

Figure 19. Classified result of test image 1 - Contextual classifier: Classified by the contextual classifier. Yellow - object, Brown - background.

To give the numerical comparison of classification accuracy, the three classifiers, Bayes, binary tree decision, and contextual classifiers, are applied to two digital remote sensing data which have known ground truth data.

The first image is the one used to generate the simulated image before. It has four spectral bands and the first band of the image is shown in Figure 20. The objective of the analysis was to discriminate three ground cover classes 1,2, and 4. Figure 21 shows the training samples overdrawn on the ground truth data. Results of the three classifiers are shown in Figure 22,23, and 24 and the contingency tables are given in Table 1. The overall classification accuracy in Table 1 is again measured as the ratio of the number of correctly classified pixels to the number of total classified pixels.

As shown in Table 1, the binary decision tree classifier gave a 4.5 % increase in overall classification accuracy, while the contextual classifier gained 5.3 % increase over the Bayes classifier. Even though the binary decision tree classifier uses the linear discriminant function it has been shown that it gave more accurate results than the Bayes classifier which uses multi-dimensional normal joint density function. Examining the result of the contextual classifier, it can be seen that as expected within each class it assigns the ambiguous noisy pixels to their most likely class, leaving almost homogeneous regions for each class. Also, it smoothed noisy boundaries between each class.

The second image is a four band image of size 200 × 175 which was taken in June 1979. Figure 25 shows the first band of the image and the ground truth data is given in Figure 26. This image contains eight ground cover classes.

Class 1 : Wheat

Class 2 : Alfalfa

Class 3 : Potatoes

Class 4 : Corn

Class 5 : Beans

Class 6 : Apples

Class 7 : Pasture (irrigated)

Class 8 : Rangeland

One fourth of the image is sampled as a training set to compute the class-conditional covariance matrices and mean vectors. The binary decision tree also uses the same training set to generate the decision tree. Figures 27,28,and 29 show the result of the three classifiers and the contingency tables are given in Table 2. We again found that the overall classification accuracy has been increased 7.7 % in the binary decision tree classifier and 5.7 % in the contextual classifier over the Bayes classifier.

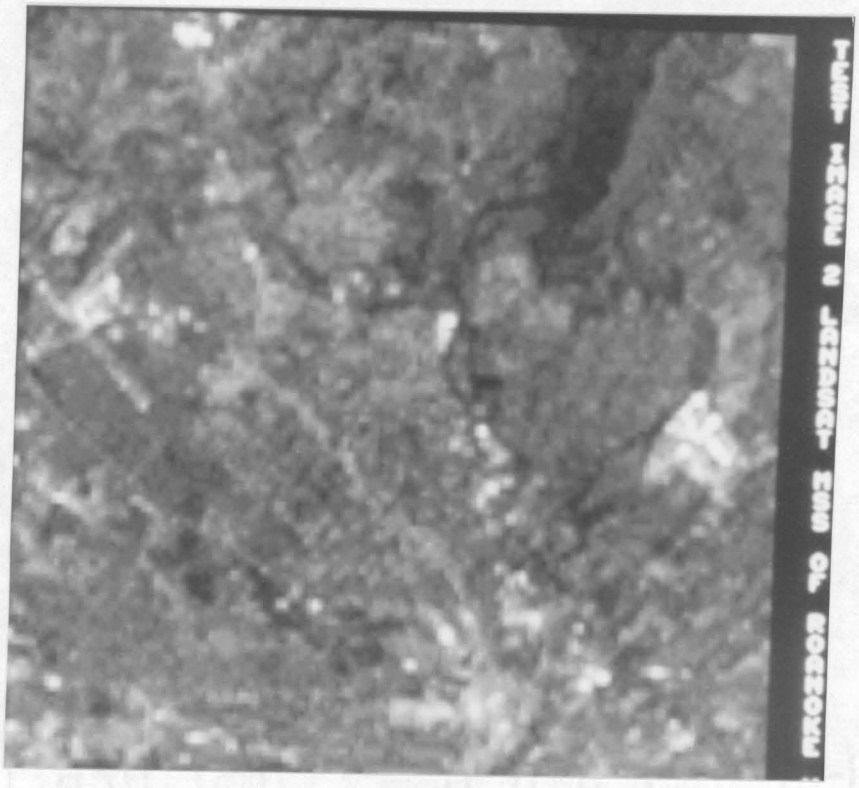

Figure 20. Test image 2: 151 × 151 subframe of MSS scene of Roanoke VA taken on 13 April 1976 (band 1). longitude from 79°52' to 80°00'W; latitude from 37°15' to 37°23'N.

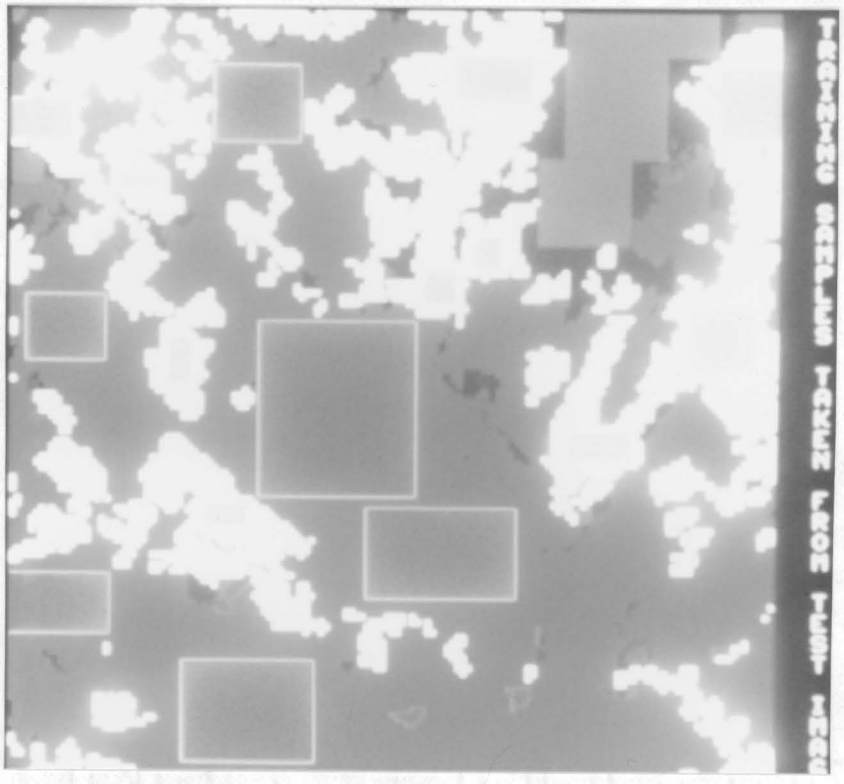

Figure 21. Training samples taken from test image 2: Small rectangular boxes represent portions of the gound truth data selected as training samples.

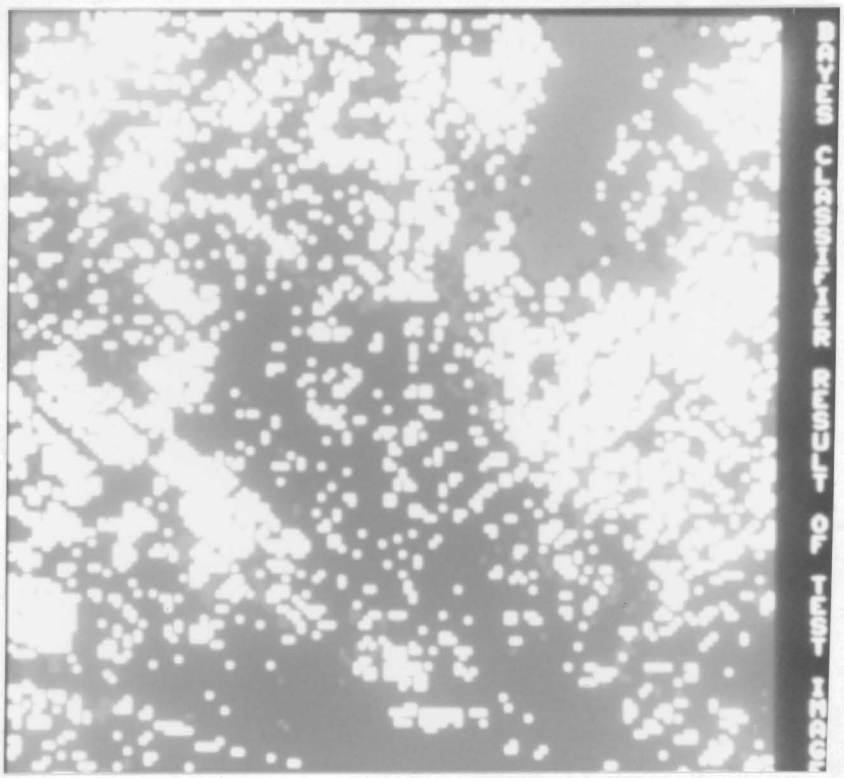

Figure 22. Classified result of test image 2 - Bayes classifier: Classified by the Bayes classifier with equal priors. Blue (class 1) - urban or built-up land, White (class 2) - agriultural land, Green (class 4) - forest land.

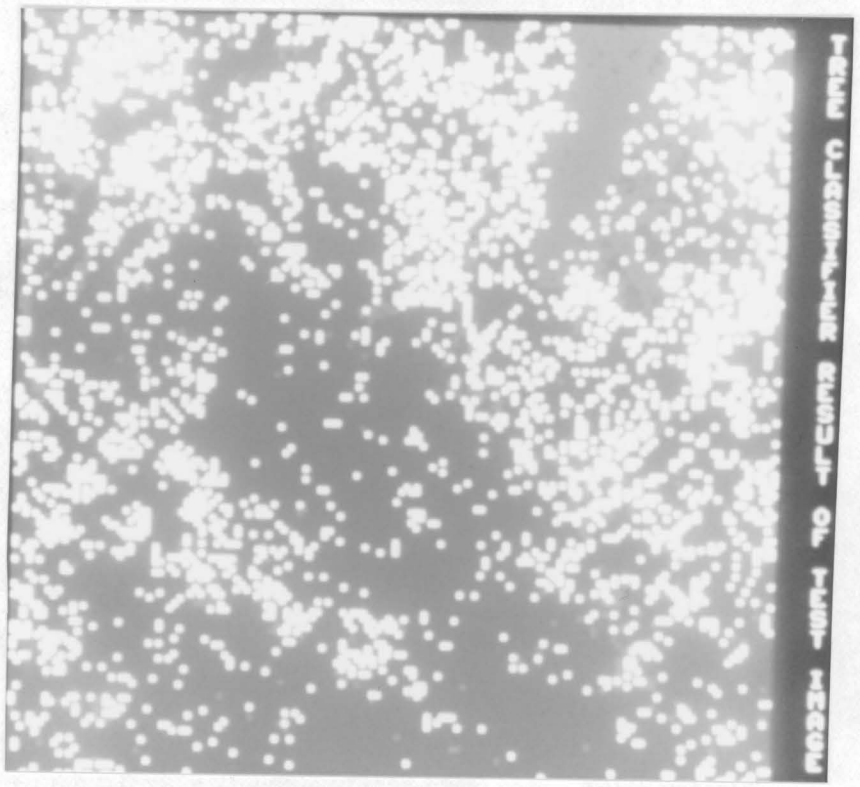

Figure 23. Classified result of test image 2 - Tree classifier: Classified by the binary decision tree classifier. Blue (class 1) - urban or built-up land, White (class 2) - agricultural land, Green (class 4) - forest land.

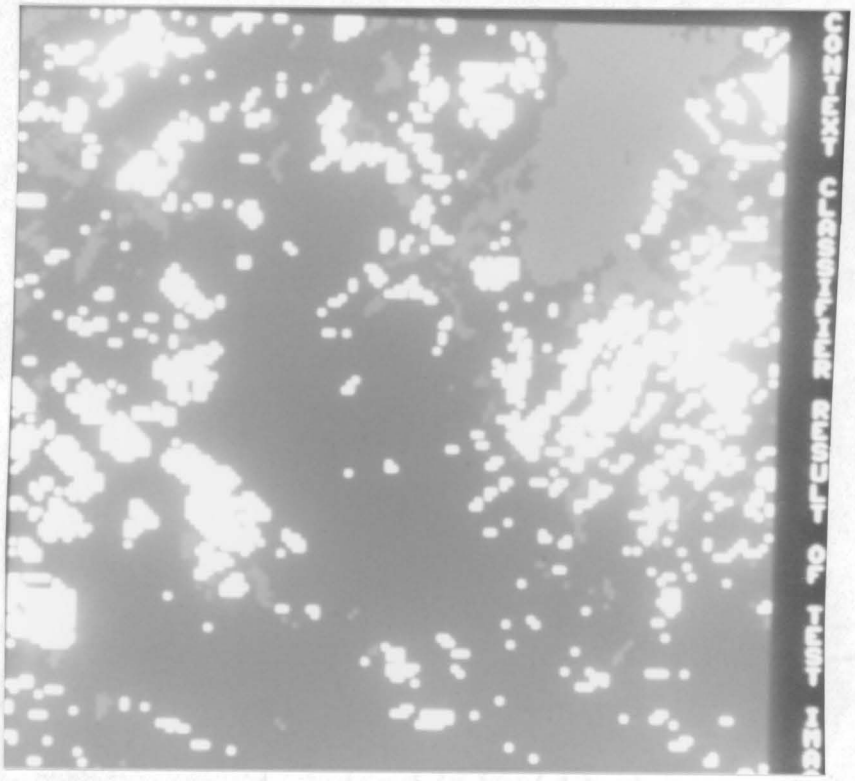

Figure 24. Classified result of test image 2 - Contextual classifier: Classified by the contextual classifier. Blue (class 1) - urban or built-up land, White (class 2) - agricultural land, Green (class 4) - forest land.

Table 1. Contingency tables for classification results of test image 2 (column = assigned class, row = true class , URB = urban or built-up land, AGR = agricultural land, FST = forest land). Scale factor for the number of pixels = 10.

Table 1 (a) Bayes classifier result

class	URB	AGR	FST	total	Acc(%)*
URB	971	411	54	1436	67.6
AGR	176	337	65	578	58.3
FST	45	18	189	252	75.0
total	1192	766	308	2266	66.1**

Table 1 (b) Binary decision tree classifier result

class	URB	AGR	FST	total	Acc(%)*
URB	1251	152	33	1436	87.1
AGR	355	186	37	578	32.2
FST	53	38	161	252	63.9
total	1659	376	231	2266	70.5**

* percent of correct classification.
** overall classification accuracy : ratio of the number correctly classified pixels to the number of total classified pixels.

Table 1 Contingency tables for classification results of test image 2 (continued)

Table 1 (c) Contextual classifier result

class	URB	AGR	FST	total	Acc(%)*
URB	1287	133	17	1437	89.6
AGR	388	161	29	578	27.9
FST	78	4	170	252	67.5
total	1753	298	216	2267	71.4**

* percent of correct classification
** overall classification accuracy

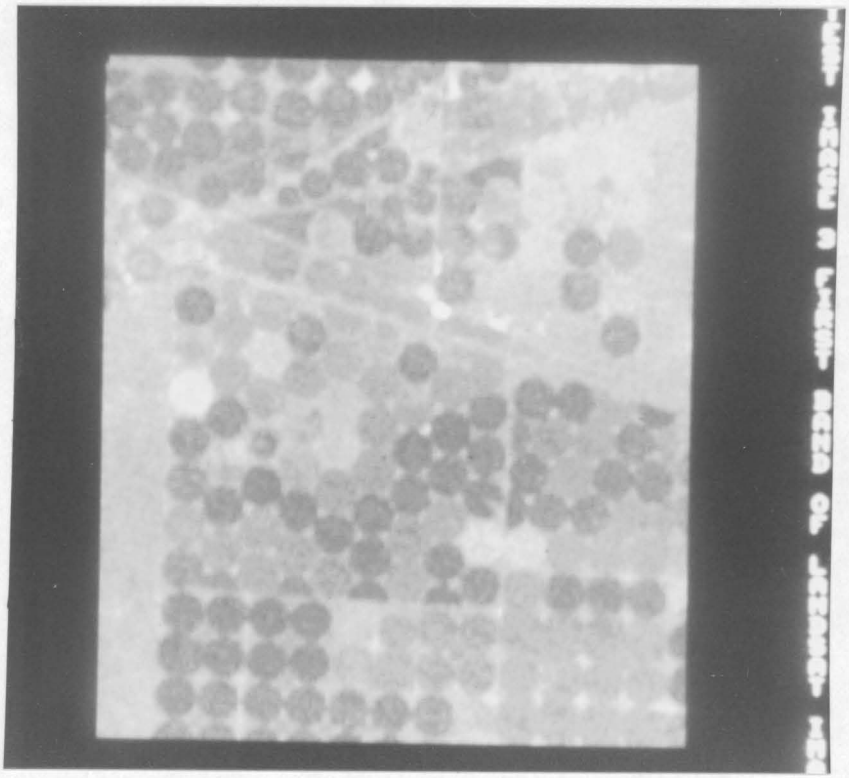

Figure 25. Test image 3: First band of the LANDSAT test image 2. A 200 × 175 size image taken in June 1979.

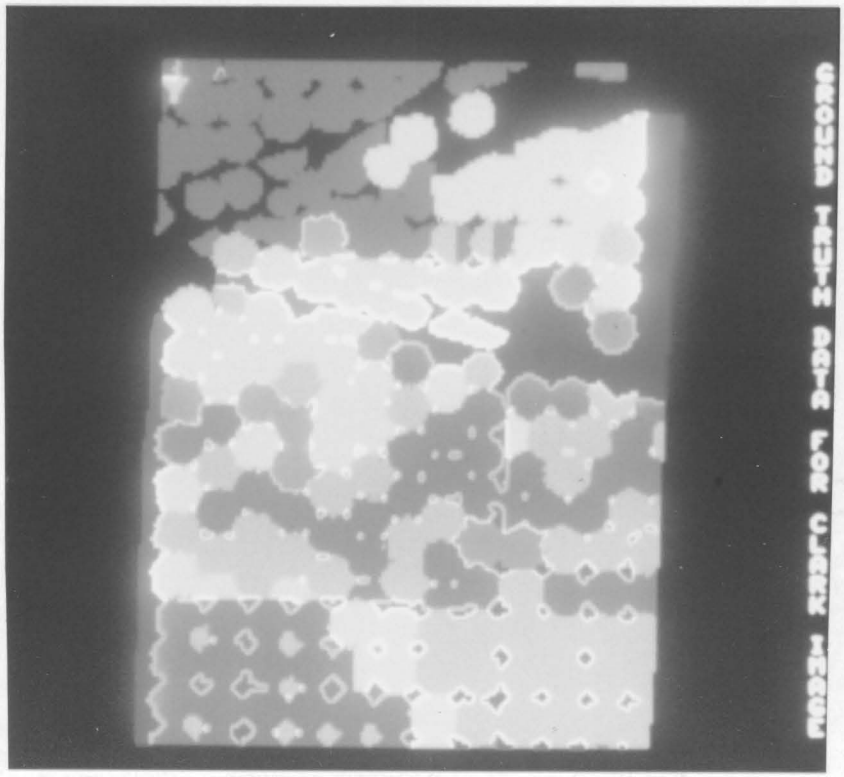

Figure 26. Ground truth data for test image 3.: There are eight classes in this data.

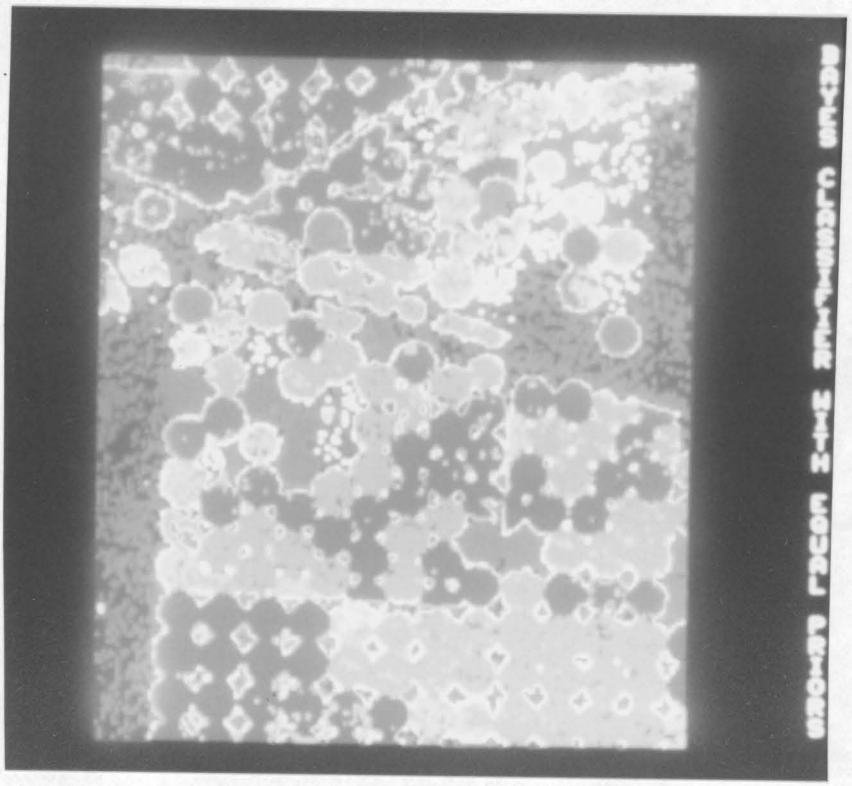

Figure 27. Classified result of test image 3 - Bayes classifier: Classified by the Bayes classifier with equal priors

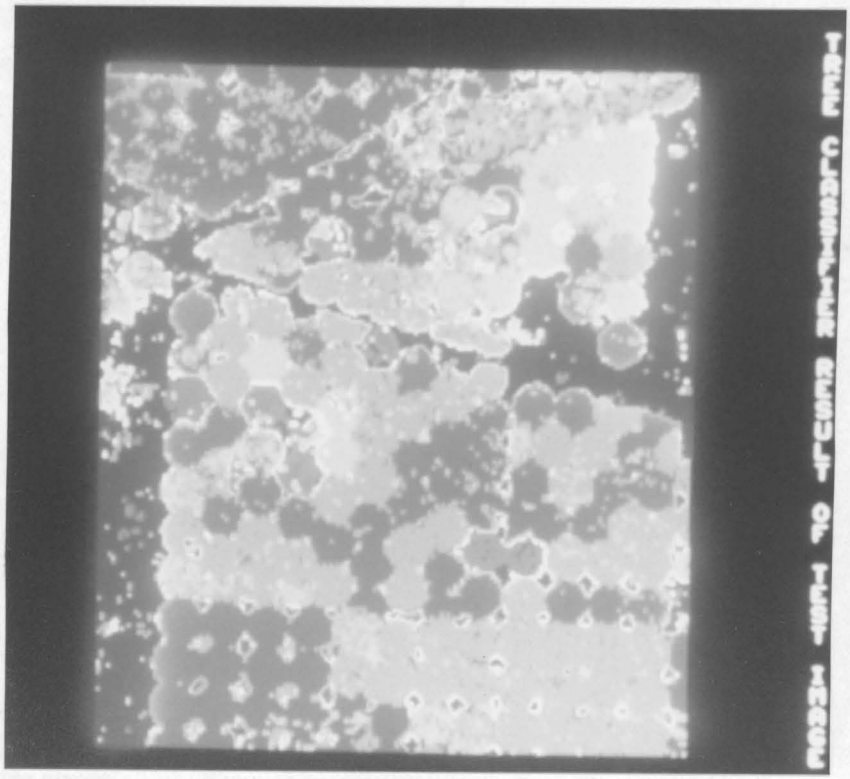

Figure 28. Classified result of test image 3 - Tree classifier: Classified by the binary decision tree classifier

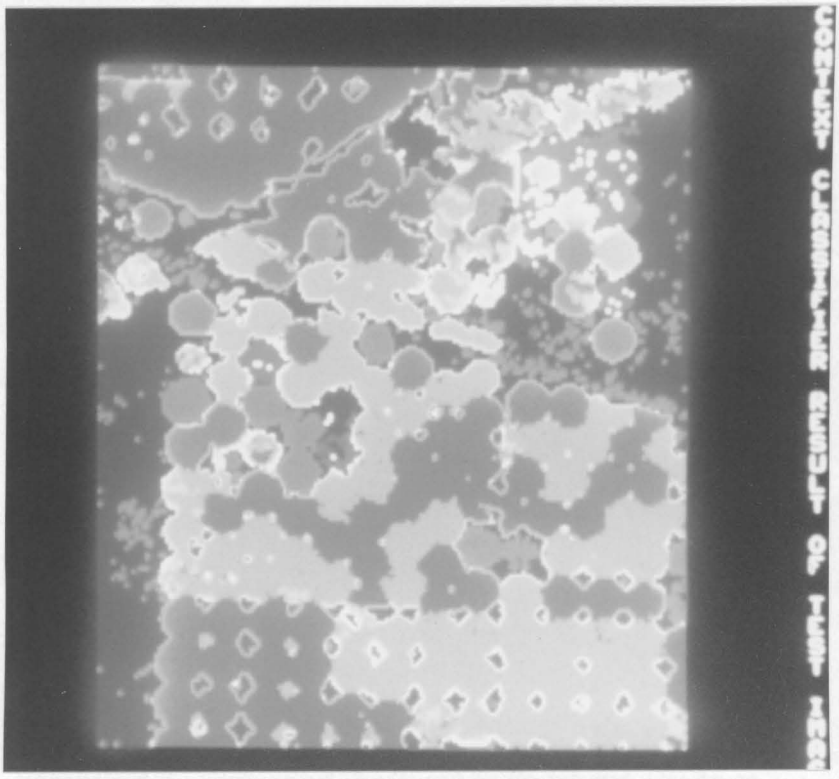

Figure 29. Classified result of test image 3 - Contextual classifier: Classified by the contextual classifier.

Table 2 Contingency tables for classification results of test image 3 (column = assigned class row = true class). Scale factor of the number of pixels = 10.

Table 2 (a) Bayes classifier result

class	WHT	ALF	POT	CRN	BNS	APL	PAS	RNG	total	Acc(%)*
WHT	829	42	62	7	8	9	2	48	1007	82.3
ALF	94	138	207	15	22	22	99	59	656	21.0
POT	41	40	493	8	6	17	7	30	642	76.8
CRN	1	1	1	50	6	2	1	6	68	73.5
BNS	1	0	1	14	20	1	1	4	42	47.6
APL	0	0	0	0	0	1	0	0	1	100.0
PAS	0	0	0	0	0	0	4	0	4	100.0
RNG	10	11	18	87	44	19	10	177	376	47.1
total	976	232	782	181	106	71	124	324	2796	61.2**

Table 2 (b) Binary decision tree classifier result

class	WHT	ALF	POT	CRN	BNS	APL	PAS	RNG	total	Acc(%)*
WHT	840	27	114	1	4	0	0	22	1008	83.3
ALF	134	242	239	5	12	0	0	22	654	37.0
POT	52	37	527	4	4	0	0	18	642	82.1
CRN	3	5	1	33	9	0	0	16	67	49.3
BNS	4	3	1	6	18	0	0	12	44	40.9
APL	1	0	1	0	0	0	0	1	3	0.0
PAS	0	3	0	0	1	0	0	0	4	0.0
RNG	46	20	28	2	13	0	0	268	377	71.1
total	1080	337	911	51	61	0	0	359	2799	68.9**

* percent of correct classification
** overall classification accuracy

Table 2 Contingency tables for classification results of test image 3 (continued)

Table 2 (c) Contextual classifier result

class	WHT	ALF	POT	CRN	BNS	APL	PAS	RNG	total	Acc(%)*
WHT	904	7	44	3	0	0	0	50	1008	89.7
ALF	102	58	268	11	1	2	104	109	655	8.9
POT	30	7	558	5	1	2	4	35	642	86.9
CRN	2	1	3	51	0	0	0	11	68	75.0
BNS	2	0	1	21	5	0	0	13	42	11.9
APL	0	0	0	0	0	1	0	1	2	50.0
PAS	0	0	0	0	0	0	4	1	5	80.0
RNG	18	2	30	30	1	1	4	290	376	77.1
total	1058	75	904	121	8	6.	116	510	2798	66.9**

WHT (class 1) : Wheat
ALF (class 2) : Alfalfa
POT (class 3) : Potatoes
CRN (class 4) : Corn
BNS (class 5) : Beans
APL (class 6) : Apples
PAS (class 7) : Pasture (irrigated)
RNG (class 8) : Rangeland

* percent of correct classification
** overall classification accuracy

The increase in classification accuracy and efficiencies can be compared to the other contextual classifiers proposed by several authors. Yu and Fu [6] used a stationary stochastic process on a two-dimensional plane as a model to extract the spatial correlation parameters, which is the context information used in their recursive contextual classifier. When tested on a real image, their results gained about 7% increase in classification accuracy over the conventional Bayes classifier in second stage of the recursive contextual classifier. Tilton and Swain [2,3] derived an unbiased estimate of the so-called context function $G(\theta^p)$ they used in the contextual classifier. The function $G(\theta^p)$, the context distribution, is the relative frequency with which θ^p occurs in the array θ, where θ^p is the true label of the p-context array θ and the p-context array is a local context, a set of pixels neighboring the pixel in consideration. Optimal estimate of $G(\theta^p)$, $T_\theta p(X)$, was obtained such a function that it minimizes the mean squared error given by,

$$MSE_\theta p = E[\ T_\theta p(X) - G(\theta^p)\ ]^2$$

They reported 2-6 % improvement in classification accuracy over the noncontextual maximum likelihood classifier.

From the descriptions above, it is obvious to see that the computation cost of those two contextual classifiers in-

creases drastically as the size of the context increases. Compared to those classifiers, the result of our classifier, which only needs the computation of the frequency distribution of pairwise pixel labels, shows compatible improvement in classification accuracy. Improvements can also be made in the contextual classifier proposed in this paper by obtaining a proper model for the computation of the function $A(e_{ij},e_{kl})$. Also, it should be mentioned that the computational cost is low as one can see in the equations (3.19) and (3.23) where the max function is used in computing $g_{U_{rc}}(e_{rc})$ and $h_{U^*_{rc}}(e_{rc})$.

More tests on classifying objects have been conducted on two thermal images. Two images were processed as follows: Two texture features, entropy and the inverse difference moment, are computed. These values and the original measurement value compose the feature space. Then, training samples are taken manually from the original image and used to estimate the class-conditional covariance matrices and mean vectors. These estimated underlying distributions are used both in the Bayes classifier and the contextual classifier. Prior probability for the object was set to 0.01 in the Bayes classifier. One fourth of the test image and the result of the Bayes classifier are used to generate the decision tree. The function $A(e_{ij},e_{kl})$ is estimated from the result of the Bayes classifier and used in the contextual classifier.

The first image is an 8-14 micron thermal image containing nine special objects to be detected of size 270 × 480 which was taken from the Fort Polk, Louisiana data base provided by ETL. The original image and the training samples are shown in Figure 30 and 31. Classified images of the three classifiers are shown in Figure 32, 33, and 34.

The second image is a two band image with spectral band 4.5-5.0 micrometers and 8-14 micrometers. This image contains eleven special objects to be detected of size 341 × 201 which was taken from the ERIM 2 multispectral data base, Fort A. P. Hill, VA. Two bands of the original image and the training samples are shown in Figure 35 and 36. Classified images of the three classifiers are also shown in Figure 37, 38, and 39.

We performed more tests on the LANDSAT images and found that the binary decision tree classifier gained 2-7% increase and the contextual classifier gained 4-8% increase in overall classification accuracy compared to the Bayes classifier under Gaussian distribution assumption.

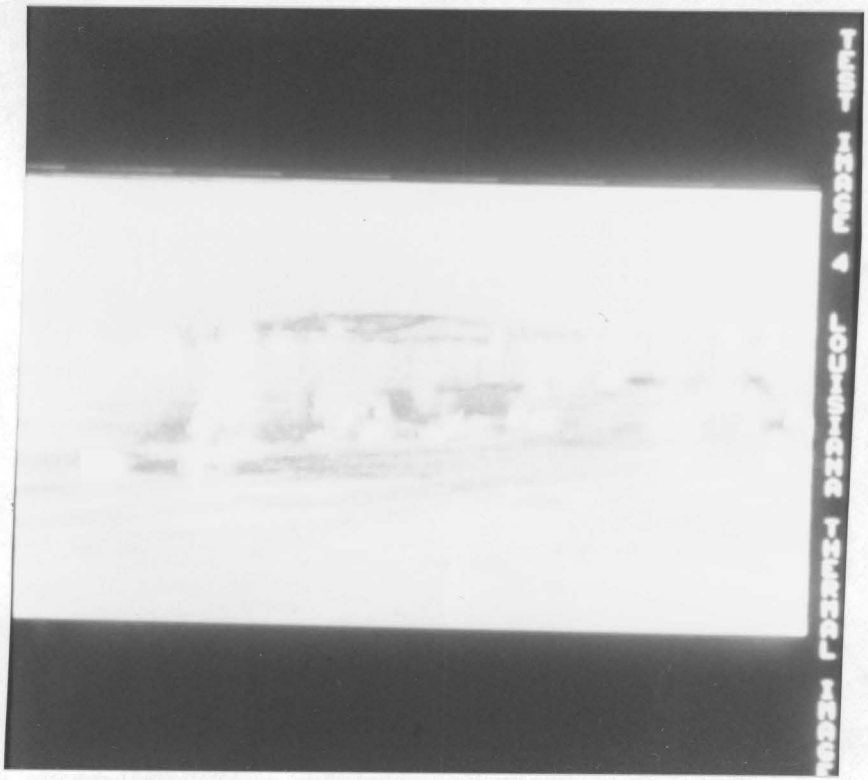

Figure 30. Test image 4: 8-14 micron thermal image containing nine special objects to be detected (bright blobs). A 270 × 480 size image which was taken from Fort Polk, Louisiana data base.

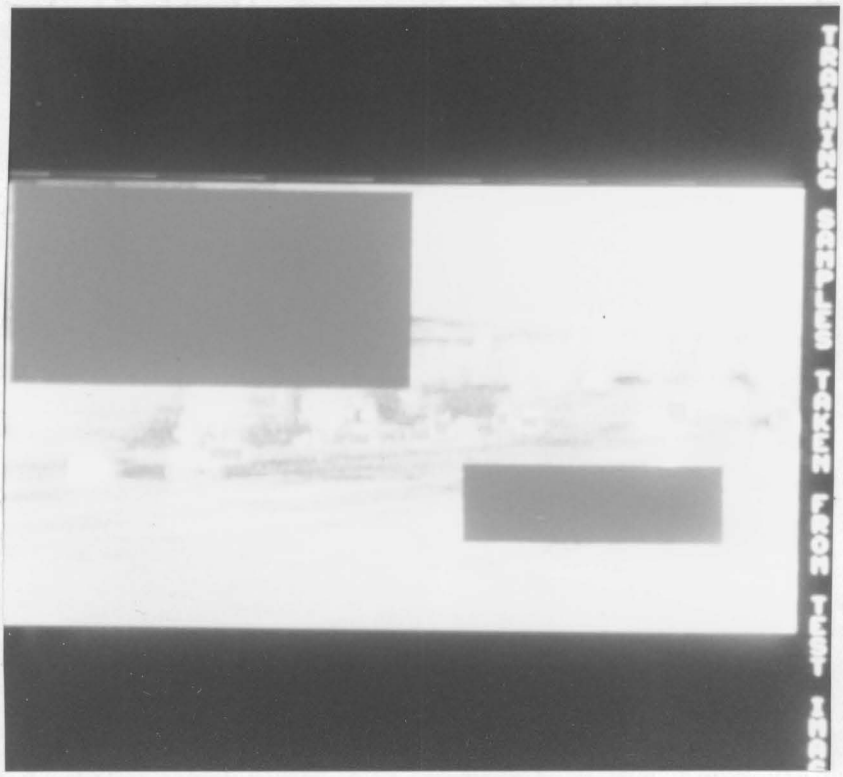

Figure 31. Training samples of test image 4: Ractangular boxes represent the selected training samples. Yellow - objects, Brown - background.

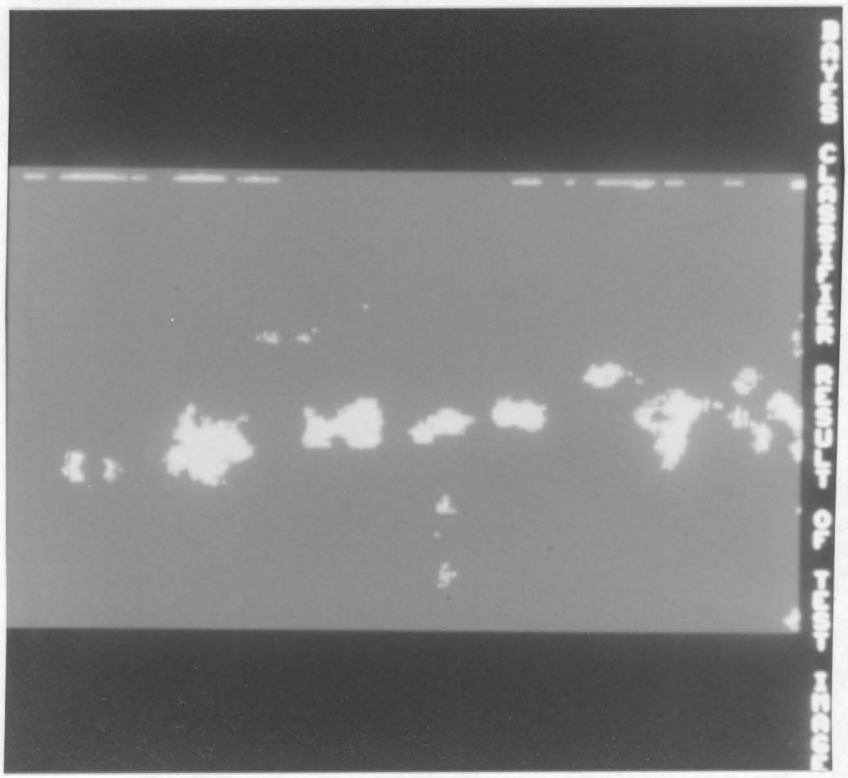

Figure 32. Classified result of test image 4 - Bayes classifier: Classified by the Bayes classifier. Prior for the object = 0.01. (Yellow - object, Brown - background)

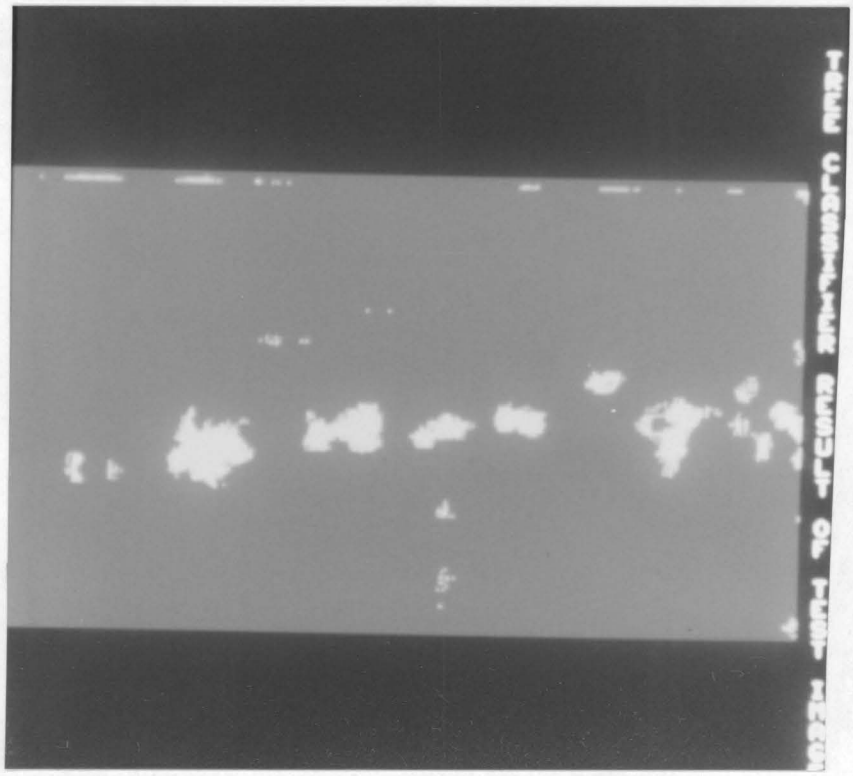

Figure 33. Classified result of test image 4 - Tree classifier: Classified by the binary decision tree classifier. (Yellow - object, Brown - background)

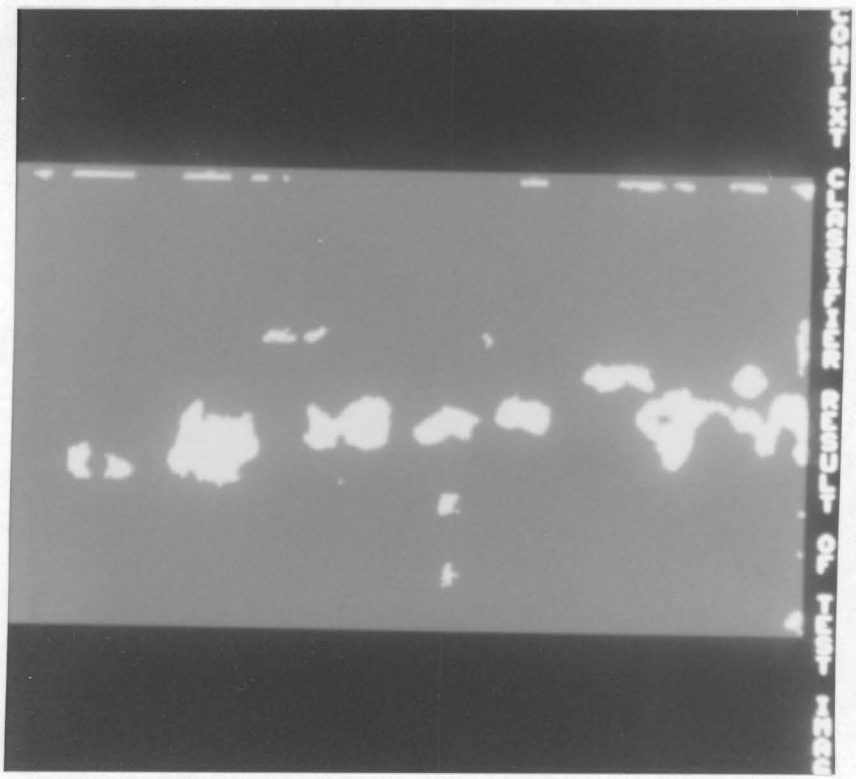

Figure 34. Classified result of test image 4 - Contextual classifier: Classified by the contextual classifier. (Yellow - object, Brown - background)

Figure 35. Test image 5: First two bands of the original thermal image in 4.5-5.0 micrometers spectral band which contains eleven special objects (white blobs) to be detected. A 341 × 202 size image taken from the ERIM 2 multispectral data base, Fort A. P. Hill, VA.

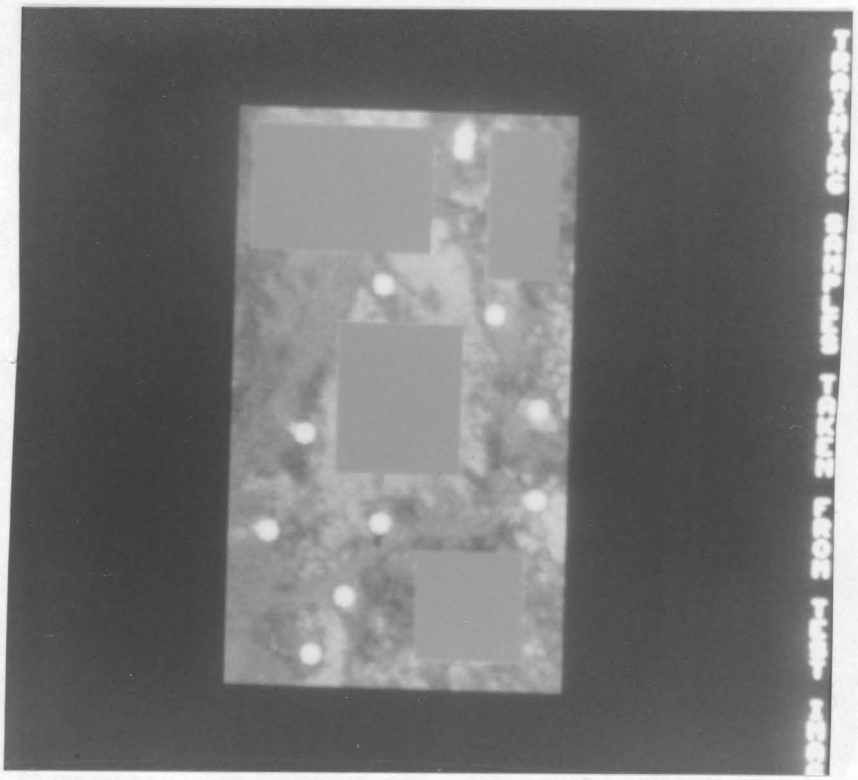

Figure 36. Training samples of test image 5: Yellow blobs represent the selected training samples of the objects. Brown boxes represent the selected training samples of the background.

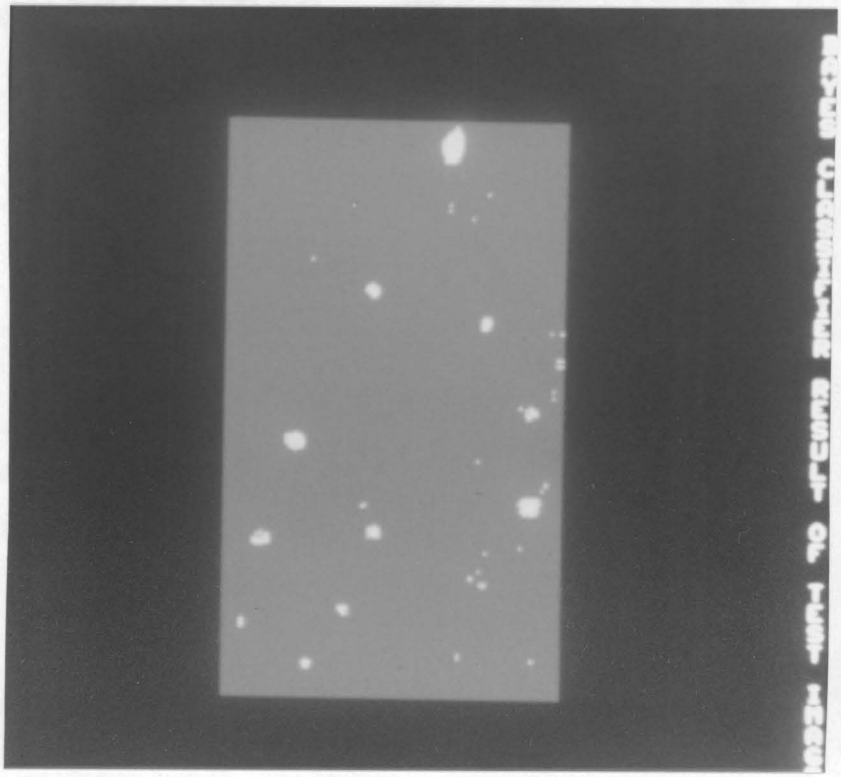

Figure 37. Classified result of test image 5 - Bayes classifier: Classified by the Bayes classifier with prior of object = 0.01. (Yellow - object, Brown - background)

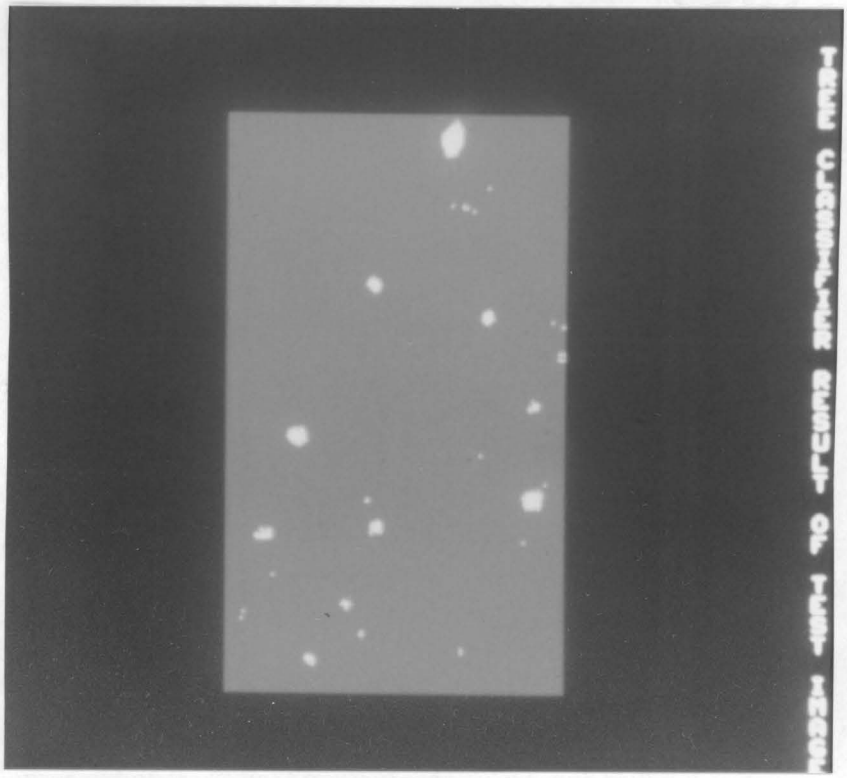

Figure 38. Classified result of test image 5 - Tree classifier: Classified by the binary decision tree classifier. (Yellow - object, Brown - background)

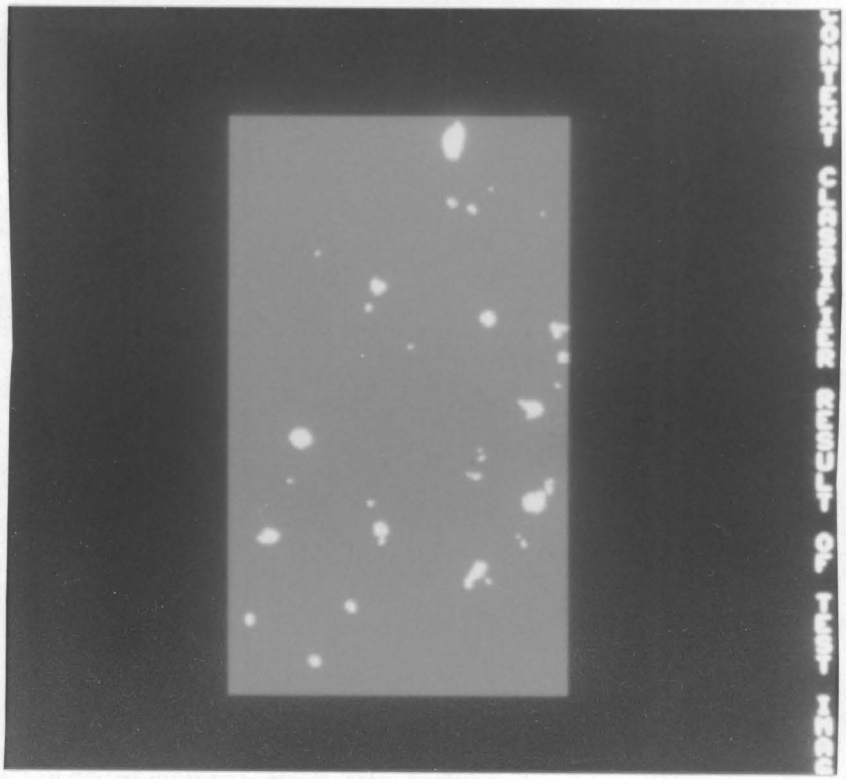

Figure 39. Classified result of test image 5 - Contextual classifier: Classified by the contextual classifier. (Yellow - object, Brown - background)

CONCLUSIONS

We presented algorithms for designing a binary decision tree classifier and a contextual classifier in this paper.

The binary tree was selected for its simplicity, and given this structure, we derived a classifier that uses a linear discriminant function as a decision rule at each nonterminal node of the tree. When the number of features and classes are small, it shows an excellent classification result as compared to the conventional single-stage Bayes classifier with equal priors. But as the number of features and classes increase, computation cost for constructing the classifier increases. This problem can be easily seen by looking at the designing process where it considers every possible groupings of the classes at each nonterminal node to find the best decision rule in the sense to achieve a maximum purity in child nodes. The algorithm presented here is just one of the many possible ways of designing a binary decision tree classifier and no claim can be made about the optimality of the algorithm.

The contextual classifier was designed so that it gives each pixel the highest probability label given some substantially sized context including the pixel. Applied to the

simulated image, its performance was superior to any other classifiers considered in this paper. It was also shown that the overall classification accuracy was increased in the real images.

Using assumptions (3.11) and (3.12) we were able to derive a two pass algorithm which can be expressed as in equations (3.19) and (3.23). Even though we used a simple frequency measuring method to estimate the joint probability of the label pairs for succesive pixels in the path, we obtained higher classification accuracy for the context experiments. We also compared the improvement in overall classification accuracy to the other contextual classifiers developed so far and found comparable results.

APPENDIX A. REFERENCES

1. R. M. Haralick, Decision making in context, IEEE trans. pattern analysis and machine intelligence, vol.PAMI-5, no.4, pp.417-428, July 1983.

2. P. H. Swain, S. B. Vardeman, and J. C. Tilton, Contextual classification of multispectral image data, Pattern recognition, vol.13, no.6, pp.429-441, 1981.

3. J. C. Tilton, S. B. Vardeman, and P. H. Swain, Estimation of context for statistical classification of multispectral image data, IEEE Trans. Geoscience and remote sensing, vol.GE-20, no.4, pp.445-452, Oct. 1982.

4. G. T. Toussaint, The use of context in pattern recognition, Pattern recognition, vol.10, pp.189-204, 1978.

5. S. W. Wharton, A contextual classification method for recognizing land use patterns in high resolution remotely senced data, Pattern recognition, vol.15, no.15, pp.317-324, 1982.

6. T. S. Yu and K. S. Fu, Recursive contextual classification using a spatial stochastic model, Pattern recognition, vol.16, no.1, pp.89-108, 1983.

7. P. Argentiero, R. Chin, and P. Beaudet, An automated approach to the design of decision tree classifiers, IEEE trans. pattern analysis and machine intelligence, vol.PAMI-4, no.1, pp.51-57, Jan. 1982.

8. M. W. Kurzynski, The optimal strategy of a tree classifier, Pattern recognition, vol.16, no.1, pp.81-7, 1983.

9. G. H. Landeweerd, T. Timmers, E. S. Gelsema, Binary tree versus single level tree classification of white blood cells, pattern recognition, vol.16, no.6, pp.571-577, 1983.

10. Y. K. Lin and K. S. Fu, Automatic classification of cervical cells using a binary tree classifier, Pattern recognition, vol.16, no.1, pp.69-80, 1983.

11. J. K. Mui and K. S. Fu, Automated classification of nucleated blood cells using a binary tree classifier, IEEE trans. pattern analysis and machine intelligence, vol.PAMI-2, no.2, pp.429-443, Sept. 1980.

12. Q. Y. Shi and K. S. Fu, A method for the design of binary tree classifiers, pattern recognition, vol.16, no.6, pp.593-603, 1983.

13. C. Wu, D. Landgree, and P. Swain, The decision tree approach to classification, School of Electrical Engineering, Perdue univ., TR-EE 75-17, May 1975.

14. R. M. Haralick, K. Shanmugam, and I. Dinstein, Texture features for image classification, IEEE trans. syst., man, cybern., vol.SMC-3, no.6, pp.610-621, 1973.

The vita has been removed from the scanned document