To the Graduate Council:

I am submitting herewith a thesis written by Yupeng Zhang entitled "Bayesian Weighted K-Means Clustering Algorithm as Applied to Cotton Trash Measurement." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Philip W. Smith, Major Professor

We have read this thesis
and recommend its acceptance:

Marshall Pace

Accepted for the Council:

Interim Vice Provost and
Dean of The Graduate School

Bayesian Weighted K-Means Clustering Algorithm as Applied to Cotton Trash Measurement

A Thesis

Presented for the

Mater of Science

Degree

The University of Tennessee, Knoxville

Yupeng Zhang

August 2001

DEDICATION

This thesis is dedicated

to my wife

Wen Yang

for her love and continuous support.

ACKNOWLEDGEMENTS

I want to express my gratitude to those who assisted me during the course of my research and preparation of this document. I gratefully acknowledge the guidance and suggestion of Dr. Philip W. Smith. His great help makes this thesis a reality. I also wish to thank my Thesis Committee, Dr. Philip Smith, Dr. Marshall Pace and Dr. Hairong Qi for their instruction and support.

This research was sponsored by Schaffner Technologies, Inc of Knoxville, Tennessee, whose financial support is gratefully acknowledged.

ABSTRACT

Image segmentation is one of most difficult tasks in computer vision. It plays a critical role in object recognition of natural images. Unsupervised classification, or clustering, represents one promising approach for solving the image segmentation problem in typical application environments. The K-Means and Bayesian Learning algorithms are two well-known unsupervised classification methods. The K-Means approach is computationally efficient, but assumes imaging conditions which are unrealistic in many practical applications. While the Bayesian learning technique always produces a theoretically optimal segmentation result, the large computational burden it requires is often unacceptable for many industrial tasks. A novel clustering algorithm, called Bayesian Weighted K-Means, is proposed in this thesis. Through simplification of the Bayesian learning approach's decision-making process using cluster weights, the new technique is able to provide approximately optimal segmentation results while maintaining the computational efficiency generally associated with the K-means algorithm. The capabilities of this new algorithm are demonstrated using both synthetic images with controlled amounts of noise, and real color images of cotton lint contaminated with non-lint material.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

Image segmentation is the process of dividing images into sets of coherent regions or objects. Although this ability is a fundamental component of human perception, automatic image segmentation is one of most difficult tasks in computer vision because it is an ill-posed problem. In other words, it is not possible to define a general metric or measure of 'correct' pixel grouping, making solutions to the problem highly task dependent. In spite of this difficulty, the demand for image segmentation algorithms is high, because grouping plays a critical role in the automated understanding of natural images. Due to this need, myriad segmentation algorithms have been proposed over the past thirty years [2] [4] [5] [6] [7] [8]. The constraints of a given application must still be defined before an appropriate segmentation approach can be chosen or developed, however.

One such application of image segmentation involves the measurement of non-lint material, or trash, in images of cotton samples. The amount of trash in cotton bales is a crucial factor in determining its market value. Existing image-based automatic trash measurement systems analyze the gray-scale images of a cotton sample surfaces using a fixed intensity threshold to identify pixels in cotton image as "lint" or "non-lint". The amount of trash is then quantified as the percentage of the surface area occupied by non-lint pixels. While this method performs well if the cotton is bright white in color, the

trash particles are always much darker than the lint, and the image light source does not degrade with use, these assumptions are often not satisfied in practice. Cotton colors range from bright white to dark yellow, non-lint material varies in darkness, and light sources degrade with use as shown in Figure 1.1. Thus, it is not possible to assign a single threshold that is able to accurately discriminate between lint and non-lint material under all possible conditions.

The primary goal of the work presented in this thesis is to develop an automatic, fast, robust, reliable, color image-based trash measurement system that is less sensitive to changes in cotton color and light degradation. Since the core of such a system is an algorithm for dividing the image into lint and non-lint regions, the appropriate choice of segmentation algorithms is crucial for success. The trash measurement task provides the following constraints to guide the choice of segmentation techniques:

- Computation time should not exceed 10 seconds per 3 MB full-color image.
- Color boundaries between lint and non-lint material are of variable contrast.
- Training data should not be required, because little is available.
- Segmentation should be fully automatic, requiring no human intervention.
- *A priori* probability of non-lint material can be estimated.
- *A priori* probability of non-lint material is always much smaller than for lint.

The majority of existing methods for image segmentation can be broadly classified into three categories: a) edge-based, b) region-oriented, and c) clustering approaches [9].

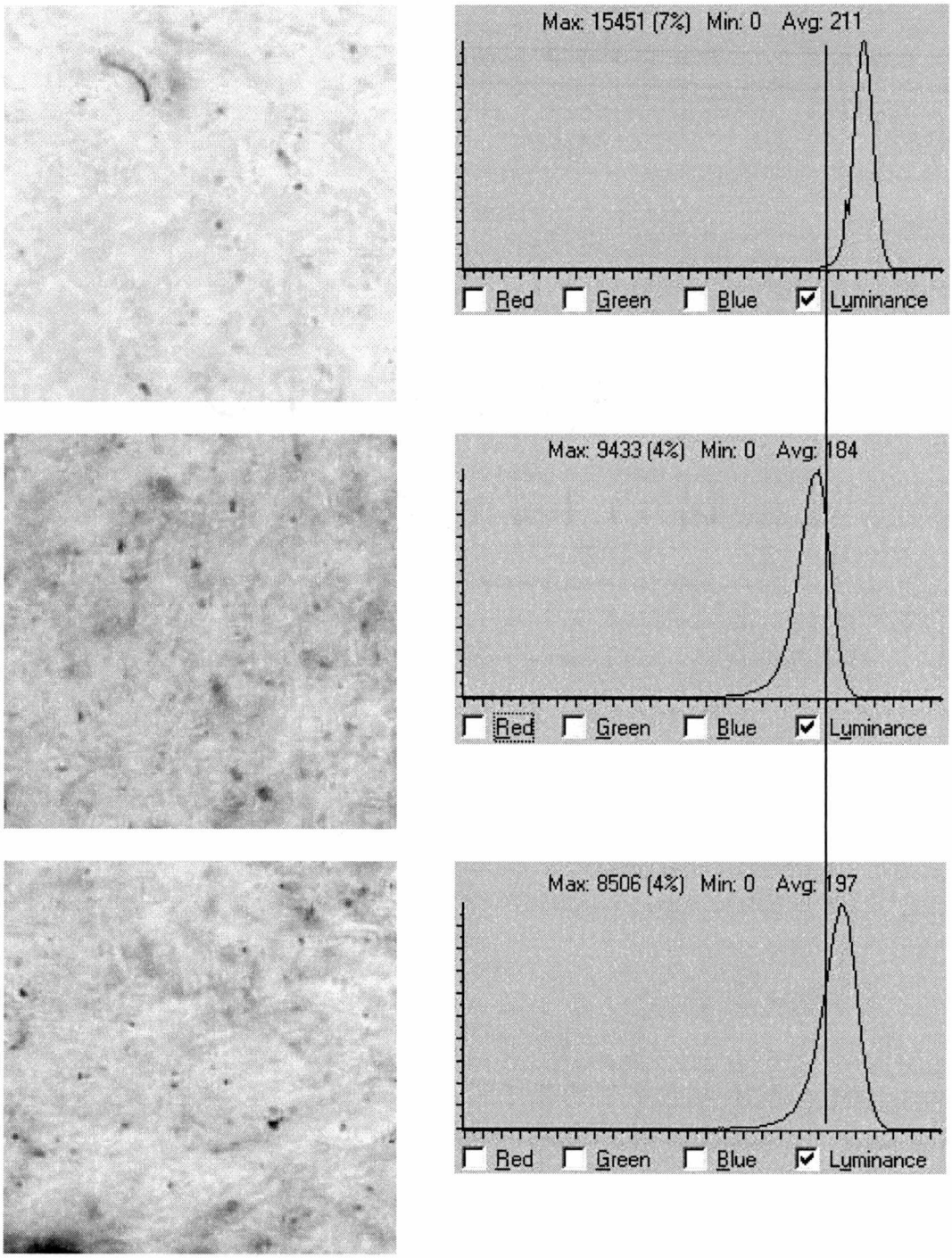


Figure 1.1 Cotton Image and Intensity Histogram

Many algorithms also exist which combine aspects of these techniques. Edge-based methods apply gradient operators to images in order to find local intensity or color boundaries [10] [11]. Regions enclosed by these boundaries are then grouped as distinct objects. For simple, low-noise, high contrast images, edge-based approaches perform well. However, for noisy, complex images with varying contrast, edge detection often produces extra edges or missing edges, resulting in undesired grouping. Occluded regions, such as walls behind objects, are also often incorrectly segmented. Edge-based techniques are not appropriate for the given segmentation task due to the varying contrast, or fuzziness, of the boundaries between lint and non-lint material in the images. Such techniques also allow limited use of the relative probability information available for the occurrence of lint and non-lint material.

Region-oriented approaches delineate pixel regions by either merging (or splitting) image regions based on similarity (or dissimilarity) of apparent intensity and/or color [12] [11]. While region-based methods are more resistant to the effects of random image noise than edge-based techniques, these algorithms still often mislabel occluded regions. Most region-based techniques also employ heuristic metrics of similarity, leading to under- or over-segmentation, and are computationally expensive. As a result, region-based methods are inappropriate for the trash measurement task.

Cluster-based approaches [14] employ techniques from statistics to identify and group image pixels with similar characteristics, or *features*, such as gradient operator

response, color, intensity, etc. Clustering algorithms represent a sub-problem of statistical pattern recognition, solutions to which are generally divided into *supervised* and *unsupervised* approaches. Supervised learning methods require large sets of training data and significant human guidance to construct decision surfaces, or boundaries between similar groups known as *clusters*, in feature spaces. Unsupervised learning methods automatically construct decision surfaces based on the similarities among the patterns without extra training data or human interference. If *a priori* statistical feature information is available, clustering algorithms are often computationally more efficient than region-based and edge-based approaches and more likely to properly group occluded objects. If such statistics are unavailable or inaccurate, however, clustering algorithms often produce undesired groupings. Because unsupervised cluster-based approaches are often relatively fast, can make use of the estimated probability information of non-lint and lint material, and require no human intervention or training data, they represent the best choice for use in a trash measurement system.

Bayesian Learning [1] [13] [15] [19] and K-means algorithms [1] [16] [17] [18] are two well-known unsupervised clustering techniques. Assuming knowledge of the *a priori* probabilities of feature classes and spherical Gaussian distributions for each cluster, the Bayesian Learning algorithm attempts to iteratively update the statistical parameters of each cluster, including mean vectors and covariance matrices, using the image feature data. After a stable statistical model is achieved, an optimal decision surface is calculated using Bayes Decision theory. This algorithm minimizes the probability of decision error

and yields the optimal solution for the classification problem assuming exact knowledge of the prior probability of the individual clusters. However, the Bayesian approach requires a significant amount of training data to find an accurate estimation of the *a priori* probabilities of classes and takes significant computation time when the feature space dimension is large.

The K-Means algorithm is a significantly faster classification method which yields the same optimal result as the Bayesian Learning algorithm when each cluster a) is modeled by a spherical Gaussian distribution, b) has equivalent probability of occurrence, and c) is nearly symmetric. For the task of cotton trash measurement, both the equivalent probability and the symmetry assumptions are not satisfied. On the other hand, the computation constraint is very important for this application. Thus, a new algorithm that is faster than Bayesian Learning and requires less restrictive assumptions than K-Means is needed.

To handle the reality of unequal probability and asymmetry present in the lint/non-lint segmentation problem while meeting the computation time constraint, I have developed a novel technique defined as the Bayesian Weighted K-Means algorithm. The Bayesian Weighted K-Means method uses the *a priori* probabilities of and covariance matrix between clusters to assign a weight to each class. Each image feature is then assigned to a given cluster based on a minimal weighted distance criterion. New weights are then computed using concepts from Bayes decision theory and the process is repeated

until a stable model is achieved. This weighting factor is directly correlated with both the statistical and the geometric properties of clusters. As the weight assigned to a given cluster increases the population and spatial scope of that class expands. Because only the relatively small set of weights, and not the entire set of image features, is updated using iterative Bayesian techniques, the new approach is much more efficient computationally and still produces close-to-optimal segmentation results.

The remainder of this paper is organized as follows. A brief overview of the K-Means and Bayesian Leaning algorithms is given in chapter 2. Chapter 3 discusses the Bayesian Weighted K-Means clustering algorithm in detail and presents experimental results on controlled images. Chapter 4 demonstrates the use of the new algorithm for cotton trash measurement. A final discussion of this work and future research directions is provided in Chapter 5.

Chapter 2

Background

As discussed in the previous chapter, the Bayesian Learning and K-Means clustering algorithms are two well-known unsupervised classification techniques. K-Means is widely used in many applications or is used as an initialization step for more computational expensive clustering methods due to its efficiency. The Bayesian Learning algorithm is often employed because it theoretically yields statistically optimal decision boundaries, despite its inefficiency. Due to their extensive use and positions at opposite ends of the computational efficiency/accuracy spectrum, these two algorithms serve as good references for evaluating the performance of new algorithms.

2.1 Bayesian Learning Algorithm

As discussed in chapter 1, the Bayesian learning algorithm is an unsupervised method that employs Bayes decision theory to iteratively learn the statistical model parameters for the classes in the given feature space. Given K feature classes, a set of K initial n-dimensional cluster centers, μ_k, a set of K initial $n \times n$ dimensional covariance matrices, Σ_k, a set of n-dimensional feature vectors, x_j, $j = 1 \ldots N$, and the *a priori* probability of each cluster, $p(k)$, each feature vector, x_j, is assigned to a cluster k such that the equation, *or decision function*,

$$d_k(x_j) = p(x_j \mid k) \cdot p(k) \tag{2.1}$$

is maximized. Assuming spherical Gaussian distributions for each cluster, the decision rule can be written as

$$d_k(\mathbf{x}_j) = \ln p(k) - \tfrac{1}{2} \ln |\Sigma_k| - \tfrac{1}{2}[(\mathbf{x}_j - \mu_k)^T \Sigma_k^{-1} (\mathbf{x}_j - \mu_k)] \quad (2.2)$$

New cluster centers, μ_k , and covariance matrices, Σ_k, are then calculated based on these feature assignments. This assigning-updating process is repeated until a stable model is established.

By maximizing the decision function equation (2.1), the Bayesian learning algorithm assigns patterns to classes based on a maximal *a posteriori* probability criterion. Thus, the expected classification error is both computable and optimal [1] for the given probability distributions and feature set. Again assuming spherical Gaussian classes, the expected classification error, $E(X)$, is given by

$$E(error) = \int \sum_{k=1..K, k \neq j} \frac{p(k)}{(2\pi)^{\frac{n}{2}} \sqrt{|\Sigma_k|}} \exp[-\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k)] \quad (2.3)$$

The Bayesian learning approach suffers from two significant limitations that often make it impractical for many applications despite its ability to provide theoretically optimal classification results for a given data set. First, the classification results it determines are highly sensitive to the initial *a priori* probability estimates given for each class. In general, in order to get an accurate estimation of *a priori* probabilities of the classes, a large amount of training data needs to be collected and analyzed. Second, the time complexity of the decision function computation step is on the order of $O(N \cdot K \cdot n^2)$.

For typical image segmentation problems where N is large, the 2nd order dependency on the feature space dimension *n* often leads to unacceptably long computation times.

2.2 K-Means Clustering Algorithms

The K-Means approach provides a simplified method for computing the optimal feature classification for a given data set under a more restrictive set of assumptions. In the Bayesian learning algorithm, each pattern, $\boldsymbol{x}_j$, is assigned to a cluster k by maximizing the decision function given in equation (2.2). Assuming the specific case in which all classes have equivalent *a priori* probability and are symmetric in feature space, the decision function, $d_k(\boldsymbol{x}_j)$, can be rewritten as

$$d_k(\boldsymbol{x}_j) = - (\boldsymbol{x}_j - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x}_j - \boldsymbol{\mu}_k) \tag{2.4}$$

Letting $\boldsymbol{\Sigma} = \sigma^2 \cdot \boldsymbol{I}$, where σ is the standard deviation in each dimensional of feature space and $\boldsymbol{I}$ is the $n \times n$ identity matrix, the assignment criterion, $d_k(\boldsymbol{x}_j)$, can be redefined as

$$\min_k (d_k(\boldsymbol{x}_j) = || \boldsymbol{x}_j - \boldsymbol{\mu}_k ||). \tag{2.5}$$

Clustering methods using this simplified decision function are known as K-Means algorithms.

As is apparent from equation 2.5, the goal of a K-Means algorithm is the minimization of the sum of squared error of all patterns in a cluster. Initial cluster centers, or seeds, are chosen in feature space, and then features are assigned to each cluster using minimum distance criteria of equation 2.5. New cluster centroids are then calculated based on the mean position of current members. As with the Bayesian approach, this assignment-update process is repeated until a stable model is established.

The primary advantage of the K-means approach over the Bayesian learning technique is reduced computational complexity. The complexity of the K-means decision function given by equation 2.5 is of the order *O(NKn)*. This linear dependence on the dimensionality of the feature space significantly reduces computation time in problems where either the size of the data set or the number of classes is large. Unfortunately, the performance of K-means algorithms degrades quickly for situations in which the equal probability and symmetry assumptions are not satisfied.

2.3 Algorithm Analysis

Figure 2.1 shows an example of a two-dimensional feature space with two clusters. The decision boundary found using the K-Means technique is the perpendicular bisector of the line joining the two cluster centers. In an *n*-dimensional feature space, the decision surface will be the hyper-plane that is the perpendicular bisector of the line joining the two cluster centers. In color-image segmentation applications, the feature space is typically a three-dimensional color space. Figure 2.2 shows a successful K-Means classification in RGB color space (see Appendix). As shown in the above two successful examples, the K-means approach provides desirable segmentation results if all the assumptions are satisfied and the clusters are well separated in the feature space.

For the example shown in Figure 2.3, when the two feature-space clusters are non-symmetric and very close to each other, the K-Means algorithm does not work well. For the same reason, the K-Means algorithm does not segment images of cotton and non-lint material well. Finally, Figure 2.4 shows a Bayesian Learning algorithm classification result to the same example as Figure 2.3. It is readily apparent that the classification

result of the Bayesian learning approach is more accurate than that of the K-Means algorithm. As discussed earlier, however, the processing time required for classification by the Bayesian Learning algorithm is much longer than that required for the K-Means algorithm.

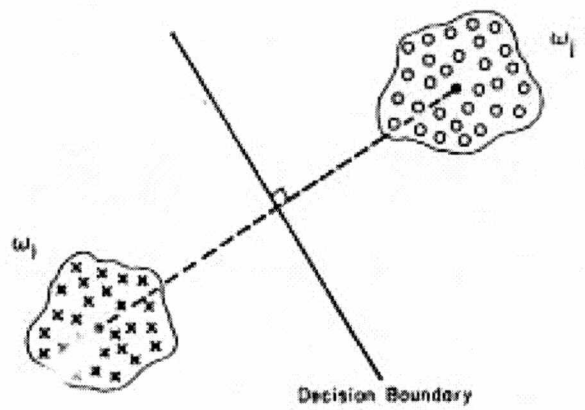


Figure 2.1 K-Means clustering algorithm result in two clusters and two-dimensional feature space example

From [1]

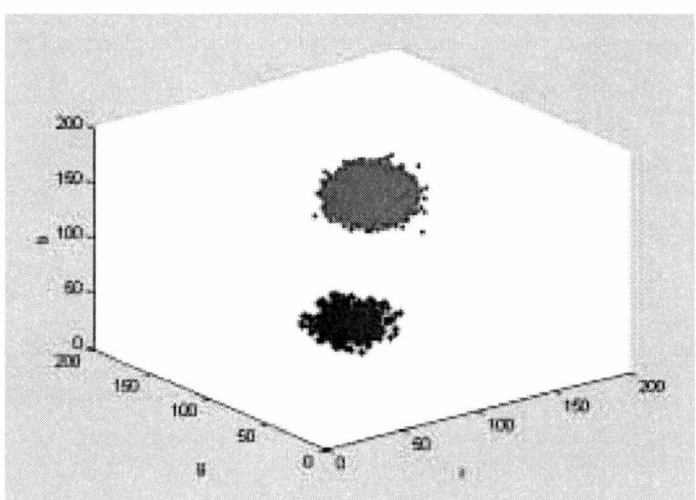


Figure 2.2 Successful K-Means classification of three-dimensional RGB color space.

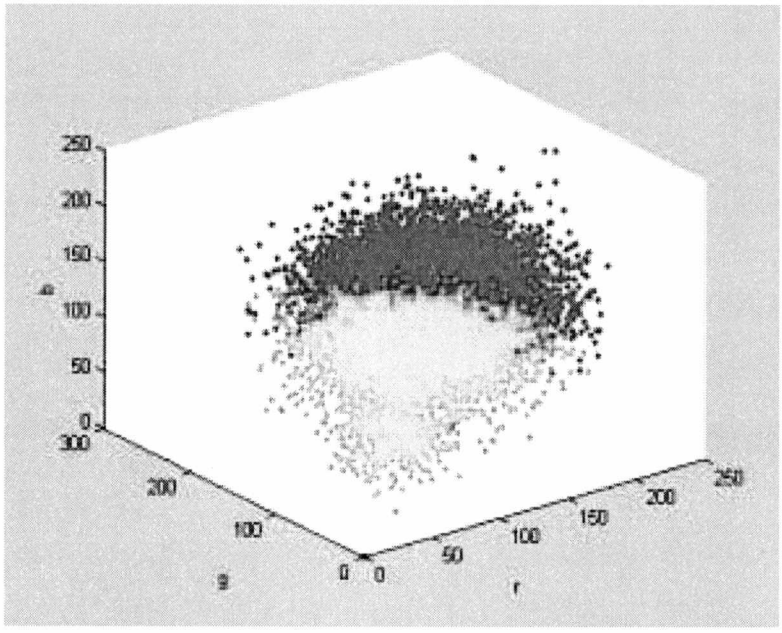


Figure 2.3 Unsuccessful Example of K-Means Classification.

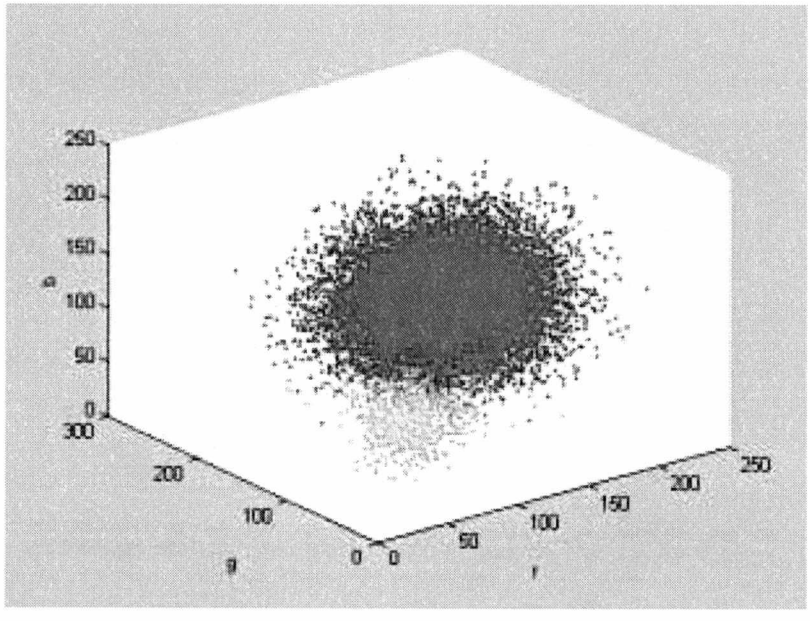


Figure 2.4 Bayesian Learning Algorithm Classification

Chapter 3

Bayesian Weighted K-Means Clustering Algorithm

As shown in the previous chapter, the K-Means algorithm is computationally efficient, but employs assumptions that are often unrealistic. Thus, K-Means does not work well in many practical applications. The Bayesian learning approach always approximates the optimal result, but is too slow for many practical applications, including the trash measurement system, due to its computational complexity, $O(N \cdot K \cdot n^2)$. To meet the demands imposed by the trash measurement task, I have developed a new algorithm, called Bayesian Weighted K-Means, which has similar computational efficiency to K-Means while yielding the same close-to-optimal result as the Bayesian learning approach.

3.1 Method

As discussed in section 2.3, computation of the decision function is the fundamental contributor to the time complexity of both the K-Means and Bayesian procedures. Thus, to maintain the computational efficiency of K-Means, the time complexity of the decision step in any new algorithm should remain linear in n, or $O(N \cdot K \cdot n)$. On the other hand, use of the *a priori* probability and covariance matrix information is important for acquiring desirable results, as shown in section 2.2. Thus, this knowledge should be considered in the decision step of a new algorithm. The Bayesian Weighted K-Means algorithm was designed by merging the computational simplicity of the K-means approach with the ability of the Bayesian technique to use *a priori* statistical data.

Given K feature space classes, a set of K initial n-dimensional cluster centers, μ_k, a set of n-dimensional feature vectors, x_j, $j = 1 \ldots N$, and a K-dimensional weighting vector $w = [\, 1/K \ldots 1/K \,]^T$, and assuming spherical Gaussian clusters, each feature vector, x_j, is first assigned to a cluster i such that the equation

$$\min_k \left(e_k(x_j) = \| x_j - \mu_k \| / w_k \right) \tag{3.1}$$

is satisfied. New cluster centers, μ_k, and covariance matrices, Σ_k, are then calculated based on these feature assignments. Note that as the weight assigned to a cluster grows, the likelihood of individuals features being assigned to that cluster increases, and the computational complexity is still of the order, $O(N \cdot K \cdot n)$.

The novel aspect of the algorithm involves the process of updating the cluster weight vector using Bayes decision theory. As noted in the previous section, the computationally expensive section of the Bayesian learning method involves the calculation of the n-dimensional Gaussian decision function, $d_k(x_j)$, over each cluster, k, $k = 1 \ldots K$, and each feature vector, x_j, $j = 1 \ldots N$. Thus, the time complexity is $O(N \cdot K \cdot n^2)$. However, instead of calculating the decision function for each feature vector and each cluster, the decision surface $D_{r,s}(x)$ between adjacent clusters r and s is calculated in the Bayesian Weighted K-Means algorithm. The time complexity for this additional step is $O(K^2 \cdot n^2)$, which is significantly smaller than $O(N \cdot K \cdot n^2)$ for the typical image processing task where N is large. The weight w_r of cluster r and w_s of cluster s are calculated based

on the distance $f(r,s)$ between cluster center μ_r and decision surface $D_{r,s}(x)$ and $f(s,r)$ between cluster center μ_s and decision surface $D_{r,s}(x)$ using

$$f(r,s) / w_r = f(s,r) / w_s \text{, where } r = 1 \ldots K \text{ and } r \neq s \tag{3.2}$$

where,

$$f(r,s) = \| \mu_r - D_{r,s}(x) \| \tag{3.3}$$

$$f(s,r) = \| \mu_s - D_{r,s}(x) \|. \tag{3.4}$$

The decision surface $D_{r,s}(x)$ for n-dimensional Gaussian distribution has the form

$$\begin{aligned} &\ln p_r - \frac{1}{2}\ln|\Sigma_r| - \frac{1}{2}[(x-\mu_r)^T \Sigma_r^{-1}(x-\mu_r)] \\ &- (\ln p_s - \frac{1}{2}\ln|\Sigma_s| - \frac{1}{2}[(x-\mu_s)^T \Sigma_s^{-1}(x-\mu_s)]) = 0 \end{aligned} \tag{3.5}$$

which can be written in general quadratic form as

$$x^T A x + b^T x + c = 0 \tag{3.6}$$

where A is $n \times n$ matrix, b is n-dimensional vector and c is a scalar. The distance $f(r,s)$ from each cluster center to the proper decision boundary can be calculated as below.

- Let $l(x)$ be the line that passes through μ_r and μ_s.
- Compute intersection z of $l(x)$ and decision surface $D_{r,sj}(x)$.
- Let $P(x)$ be the tangential hyper-plane of decision surface $D_{r,sj}(x)$ at point z.
- Compute the distance, $f(r,s)$, from μ_r to $P(x)$.

Because the weighting vector w has K unknowns, K constraints are needed for a solution. From Equation (3.2), for all the possible combinations of r and s, $K - 1$ of them are independent. Let $r = 1 \ldots K-1$ and $s = r + 1$, equation (3.2) can be written as

$$\boldsymbol{G \cdot w} = \boldsymbol{0} \tag{3.7}$$

where

$$G = \begin{bmatrix} f(2,1) & -f(1,2) & 0 & 0 & 0 \\ 0 & f(3,2) & -f(2,3) & 0 & 0 \\ 0 & 0 & \dots & \dots & 0 \\ 0 & 0 & 0 & f(K,K-1) & -f(K-1,K) \end{bmatrix}_{(K-1)\times K} \tag{3.8}$$

Equation 3.7 is a homogeneous system with *K-1* equations, *K* unknowns and rank*(G)* = *K-1*. A solution for the weighting vector up to a scale, ***w'***, can be found from equation 3.7 using SVD. The solution ***w'*** is the eigenvector corresponding to the smallest eigenvalue of $G^T G$. In addition,

$$\Sigma \boldsymbol{w} = \boldsymbol{1} \tag{3.9}$$

Thus, the actual weighting vector is determined by

$$\boldsymbol{w} = \boldsymbol{w'} / \Sigma(\boldsymbol{w'}) \tag{3.10}$$

Once the weighting vector has been updated, the entire assign-update procedure is repeated until the cluster centers and weights converge to a stable value.

Note that calculating the intersection of a line with a general quadratic can be computationally burdensome for high dimensional feature spaces. Because the decision surfaces are used for updating of the weighting vector instead of dividing the patterns into classes, dominant modes can be used to approximate a lower order, $p < n$, decision surface. Thus, in the current implementations of the algorithm, lower dimensional features, ***x'***, are used in Equations 3.3 to 3.6 to solve for the weighting vector. The one-dimensional and two-dimensional cases are discussed below.

In the case with one dominant discriminatory mode, the lower dimensional feature vector $\boldsymbol{x}'$ becomes a scalar x, $\boldsymbol{\mu}_r$ becomes a scalar u_r, $\boldsymbol{\mu}_s$ becomes a scalar u_s, and the decision surface $D_{r,s}(\boldsymbol{x})$ is given by the parabola

$$a \cdot x^2 + b \cdot x + c = 0 \tag{3.11}$$

Thus, the quadratic formula can be used to solve for the decision point between two adjacent clusters,

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{3.12}$$

Letting $T = x_{1,2}$ where T is chosen such that, $u_r < T < u_s$, the distance from each cluster center can be computed using

$$f(r,s) = \| \boldsymbol{\mu}_r - D_{r,j}(\boldsymbol{x}) \| = | u_r - T | \tag{3.14}$$

and

$$f(s,r) = \| \boldsymbol{\mu}_s - D_{r,s}(\boldsymbol{x}) \| = | u_s - T | \tag{3.15}$$

In the case of two discriminatory modes, the feature vector $\boldsymbol{x}'$ reduces to a point (x, y), $\boldsymbol{\mu}_r$ becomes a point (u_r, v_r); $\boldsymbol{\mu}_s$ becomes a point (u_s, v_s) and the decision surface $D_{r,s}(\boldsymbol{x})$ is given by the ellipsoid:

$$A \cdot x^2 + B \cdot x \cdot y + C \cdot y^2 + D \cdot x + E \cdot y + F = 0 \tag{3.16}$$

The distance $f(r,s) = \| \boldsymbol{\mu}_r - D_{r,s}(\boldsymbol{x}) \|$ can be calculated as below (see Figure 3.1):

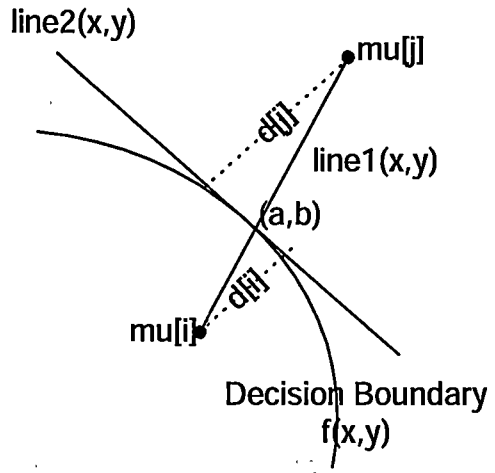


Figure 3.1 Compute the distance from center to decision surface

- Let $l_1(x,y)$ be the line that passes through μ_r and μ_s

$$y - v_r = (v_s - v_r)/(u_s - u_r)\cdot(x - u_r) \tag{3.17}$$

- Compute the intersection (a,b) of $l_1(x,y)$ and curve $D_{r,s}(x,y)$.
- Compute

$$\frac{\partial y}{\partial x} = -\frac{2\cdot A\cdot x + B\cdot y + D}{B\cdot x + 2\cdot C\cdot y + E} = -\frac{2A\cdot a + B\cdot b + D}{B\cdot a + 2\cdot C\cdot b + e} \tag{3.18}$$

- Let $l_2(x,y)$ be the line that passes through (a,b) and has the slope, $\partial y/\partial x$ such that

$$y - b = \partial y/\partial x \cdot (x - a). \tag{3.19}$$

- Compute the distance from (u_r,v_r) to $l_2(x,y)$ (see Figure 3.2) using

$$f(r,s) = |v_i - y| \cdot \cos(\pi - \alpha), \text{ where } \alpha = tan^{-1}(\partial y/\partial x). \tag{3.20}$$

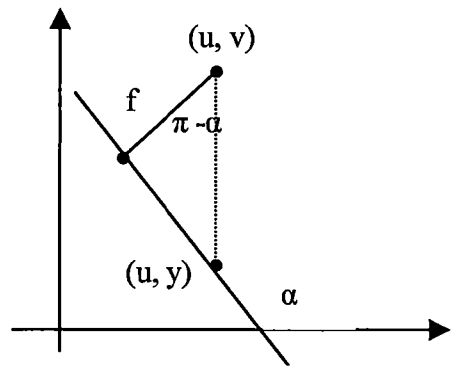


Figure 3.2 Distance from point to straight line

3.2 Algorithm Description

A complete listing of the algorithm is provided below. For complete details refer to section 3.1.

Assumptions:

- Each cluster is modeled by a spherical Gaussian distribution

Input Data:

- Pattern Set – $\boldsymbol{X} = \{\boldsymbol{x}\ in\ R^n \mid \boldsymbol{x}_j, j = 1\ ...\ .N\}$
- Number of Clusters – K
- Initial Cluster Centers – $\boldsymbol{\mu}_k\ in\ R^n,\ k = 1\ ..\ K$
- Initial K-dimensional weighting vector, $\boldsymbol{w} = [1/K\ ..\ 1/K]^T$
- *A priori* probability of cluster – p_k

Steps:

1. For each feature vector x_j, compute $e_k(x_j) = \| x_j - \mu_k \| / w_k$ for each cluster center, μ_i. Find the minimal value of $e_k(x_j)$ with respect to k for each feature vector. Assign pattern x_j to cluster k.
2. Compute new cluster centers μ_k and covariance matrices Σ_k for each cluster.
3. Update the weighting vector w.
   a) For cluster r and $s = r + 1$ where $r = 1 .. K-1$, compute the decision surface $D_{r,s}(x')$ using Bayes decision theory, where x' is a subset of size one or two selected from x to reduce the computational complexity of the decision surface.
   b) Compute the distance from each cluster center to the appropriate decision surface

$$f(r,s) = \| \mu_r - D_{r,s}(x') \| \tag{3.21}$$

and

$$f(r,s) = \| \mu_s - D_{r,s}(x') \| \tag{3.22}$$

   c) Let

$$f(r,s) / w_r = f(s,r) / w_s , \; r = 1 .. K - 1, \; s = r + 1. \tag{3.23}$$

   d) Use the expression

$$G \cdot w = 0 \tag{3.24}$$

where

$$G = \begin{bmatrix} f(2,1) & -f(1,2) & 0 & 0 & 0 \\ 0 & f(3,2) & -f(2,3) & 0 & 0 \\ 0 & 0 & \dots & \dots & 0 \\ 0 & 0 & 0 & f(K,K-1) & -f(K-1,K) \end{bmatrix}_{(K-1)\times K} \tag{3.25}$$

and the normalizing constraint

$$\Sigma(w) = 1 \tag{3.26}$$

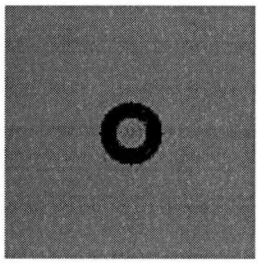
to solve for a new weighting vector w.

4. If the old model and new model parameters, $\{\mu,w\}$, are sufficiently close to each other, terminate, else return to step 1.

3.3 Experimental Results

The results of two experimental trials of the Bayesian Weighted K-Means algorithm are presented in this section. These experiments involved the use of synthetic color images with two ($K = 2$), and three ($K = 3$) distinct regions, respectively. These experiments were performed to investigate the ability of the Bayesian Weighted K-Means algorithm to provide desired segmentation in the presence of noise. Synthetic images are used so that both the correct segmentation and noise parameters can be controlled and quantified. The version of the algorithm used in this test was developed using MS Visual C++ 6.0, and was executed using a Pentium III 650MH/MS platform running Windows ME.

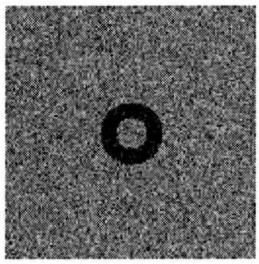
In the two clusters experiments, each *128 x 128* test image is generated by adding zero-mean Gaussian noise of differing variances to the synthetic image shown in Figure 3.3 (a). The background RGB color of this image is (128,128,128) with probability of occurrence, $P(b) = 0.8$, and the foreground RGB color is (64, 64, 64) with probability of



(a) Original Image

Background (128, 128, 128)

Foreground (64, 64, 64)



(b) Example Test Image

Background Noise = 32

Foreground Noise = 16

Figure 3.3 Input Image and Noisy Example for the Two Clusters Experiments

occurrence, $P(f) = 0.2$. The Bayesian Weighted K-Means, basic K-Means and Bayesian clustering algorithms are then applied to the test images. The classification result is compared to a standard classification based on knowledge of correct pixel grouping, and the actual classification error rate is calculated using the expression:

$$E = (\text{misclassified pixels}) / (\text{total pixels}) * 100\% \quad (3.27)$$

The expected error of the optimal segmentation case is calculated using equation 2.3. The error rates are compared with the theoretically expected error to demonstrate and analyze algorithmic performance.

Table 3.1 shows the results for a set of two cluster experiments. Note that the Bayesian Weighted K-Means algorithm provides error rates that closely approximate the expected optimal classification error from decision theory while maintaining computational times which are similar to and better than those of the traditional K-means and Bayesian learning approaches, respectively.

Table 3.1 Comparison for Two Clusters Experiments

(a) Error Rate Comparison for Two Clusters Experiments

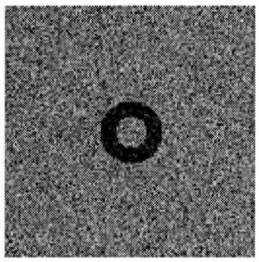
Std. Deviation Background Noise	Std. Deviation Foreground Noise	% Error BWKM	% Error K-Means	% Error BLA	Expected Error, E	Difference BWKM - E
32	32	1.459	33.722	1.147	0.95	0.509
32	24	1.227	33.691	1.227	0.861	0.366
32	16	0.47	33.582	1.404	0.464	0.006
32	8	0.092	33.716	1.898	0.077	0.015
24	32	0.824	25.079	1.105	0.684	0.14
24	24	0.458	24.268	0.507	0.516	-0.058
24	16	0.067	24.78	0.116	0.198	-0.131
24	8	0.024	24.445	0.427	0.016	0.008
16	32	0.159	0.214	1.434	0.229	-0.07
16	24	0.085	0.079	0.903	0.122	-0.037
16	16	0	0.037	0.189	0.022	-0.022
16	8	0	0.012	0	0	0
8	32	0.006	0.159	2.057	0.015	-0.009
8	24	0.006	0.024	1.678	0.004	0.002
8	16	0	0	0.708	0	0
8	8	0	0	0	0	0

Table 3.1 (Continued)

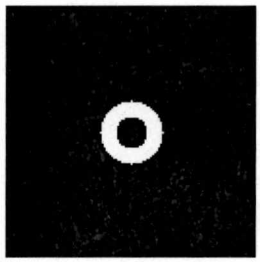
(b) Processing Time Comparison for Two Clusters Experiments

Std. Deviation Background Noise	Std. Deviation Foreground Noise	Processing Time, T1, BWKM (seconds)	Processing Time, T2, K-Means (seconds)	Ratio T1/T2	Processing Time, T3, BLA (seconds)	Ratio T1/T3
32	32	13	12	0.92	17	1.31
32	24	9	9	1	20	2.22
32	16	6	11	1.83	21	3.5
32	8	7	10	1.43	31	4.43
24	32	7	22	3.14	15	2.14
24	24	5	17	3.4	11	2.2
24	16	5	15	3	16	3.2
24	8	5	16	3.2	38	7.6
16	32	4	2	0.5	11	2.75
16	24	3	2	0.67	15	5
16	16	3	2	0.67	10	3.33
16	8	4	2	0.5	6	1.5
8	32	4	2	0.5	15	3.75
8	24	3	3	1	15	5
8	16	3	1	0.33	20	6.67
8	8	3	1	0.33	2	0.67

In order to further illustrate the operation of the Bayesian Weighted K-Means algorithm, two of the experiments listed in Table 3.1 will be presented in detail. Figure 3.4 shows the segmentation results for the case with background and foreground noise standard deviations of 32 and 16. Figure 3.5 shows the same results for the trial with background and foreground noise standard deviations of 32 and 8. Table 3.2 and Table 3.3 show the results comparison of different algorithms.

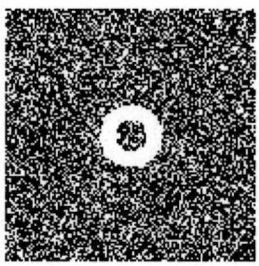


(a) Test Image

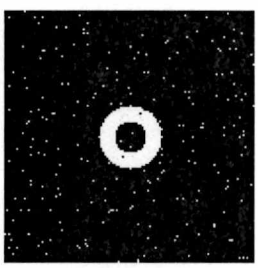


(b) Correct Classification

(c) Bayesian Weighted K-Means Classification



(d) Basic K-Means Classification

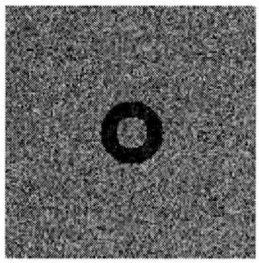


(e) Bayesian Learning Classification

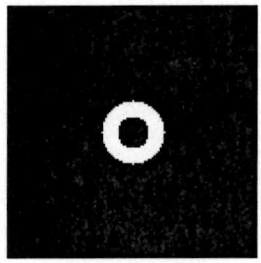
Background Cluster: Black (0)

Foreground Cluster: White (1)

Figure 3.4 Two Clusters Trial with Noise Deviations (BG = 32, FG = 16)

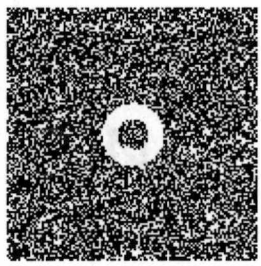


(a) Test Image

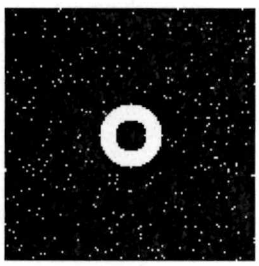


(b) Standard Classification

(c) Bayesian Weighted K-Means Classification



(d) Basic K-Means Classification



(e) Bayesian Learning Classification

Background Cluster: Black (0)

Foreground Cluster: White (1)

Figure 3.5 Two Clusters Trial with Noise Deviations (BG = 32, FG = 8)

Table 3.2 Two Clusters Trial – Noise Deviation (BG = 32, FG = 16)

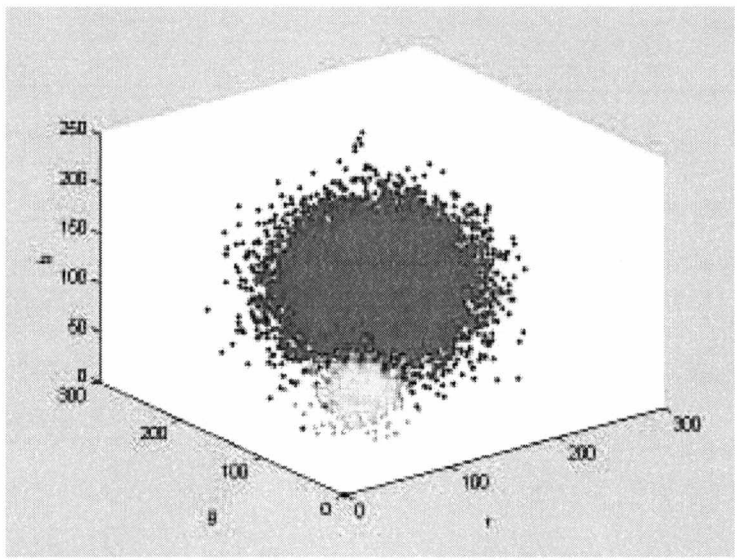
Algorithm	Error%	Processing Time (sec.)	Weight Vector	Final Background Cluster Center	Final Foreground Cluster Center
Bayesian K-Means	0.500	6	(0.730, 0.270)	(127.1, 128.1, 128.4)	(63.2, 61.9, 64.1)
Basic K-Means	33.307	11	(0.500, 0.500)	(138.0, 137.6, 138.1)	(103.7, 103.4, 103.7)
Bayesian Learning	1.404	21	(0.500, 0.500)	(128.9, 128.4, 129.1)	(68.0, 69.8, 67.2)
Expected	0.464			(128, 128, 128)	(64, 64, 64)

Table 3.3 Two Clusters Trial – Noise Deviation (BG = 32, FG = 8)

Algorithm	Error%	Processing Time	Weight Vector	Final Background Cluster Center	Final Foreground Cluster Center
Bayesian Weighted K-Means	0.061	7	(0.799, 0.201)	(128.1, 128.1, 128.2)	(64.5, 63.6, 64.0)
Basic K-Means	34.869	10	(0.500, 0.500)	(138.2, 138.5, 138.8)	(104.3, 104.2, 103.3)
Bayesian Learning	1.898	31	(0.500, 0.500)	(129.0,129.4, 129.1)	(70.4, 70.8, 69.8)
Expected	0.077			(128, 128, 128)	(64, 64, 64)

In both of these experiments, the ideal center of the background cluster is (128, 128, 128) and the ideal center of foreground cluster is (64, 64, 64). Once noise is added, the background and foreground colors could fall within +/-3 standard deviations from the cluster centers. Thus, background color values range from (32, 32, 32) to (224, 224, 224) and foreground color values from (16, 16, 16) to (112, 112, 112) in the experiment shown in Figure 3.4. Although the feature space area shared by both regions is large, the classification error of the Bayesian Weighted K-Means method is only 0.5%, which is very close to the expected error of 0.464%. This error rate is superior to both the Bayesian learning classification error of 1.404%, and the K-Means error of 33.3%. It should be noted that the classification error for the learning approach is higher than both that of the weighted technique and the optimal value expected due to inaccuracies in the estimated *a priori* probability. The feature space partitioning produced by each algorithm is shown if Figure 3.6.

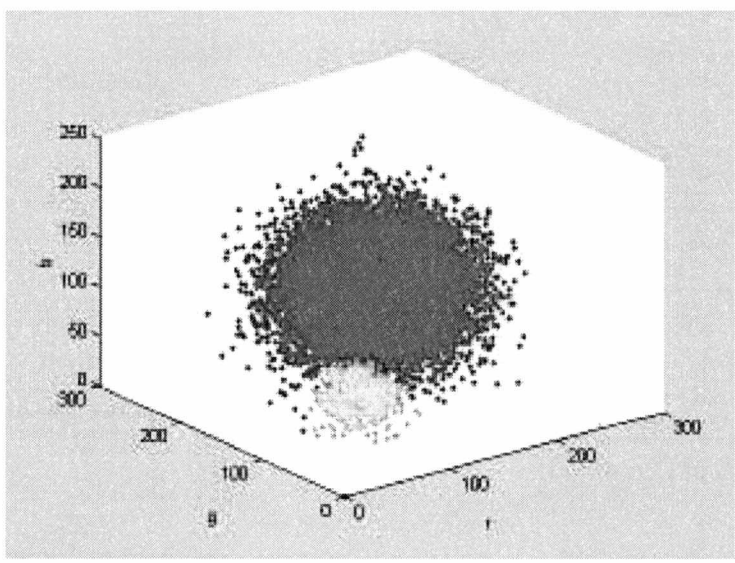
In the experiment of Figure 3.5, the background and foreground clusters have less overlap due to significantly different noise deviations, but are still close to each other in feature space. Note the final weighting vector of the Bayesian Weighted K-Means algorithm reflects this difference in cluster sizes. Figure 3.7 shows the classified feature space produced using the different algorithms.



(a) Correct Classification

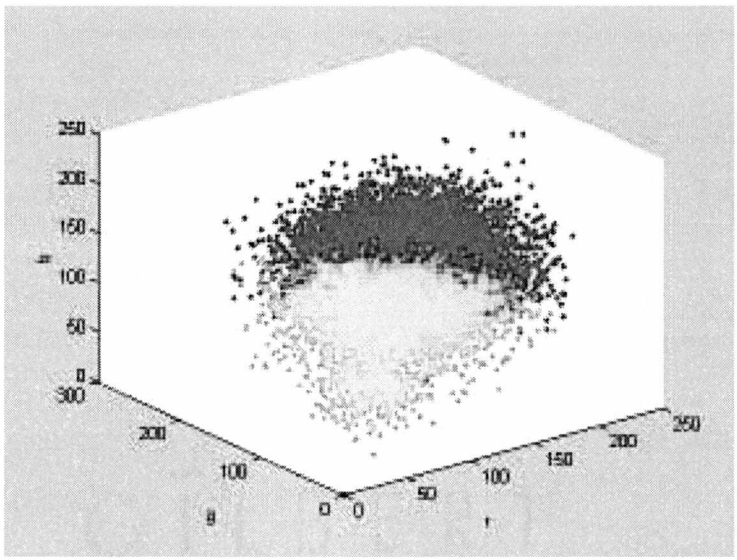
Background Cluster: Red

Foreground Cluster: Green

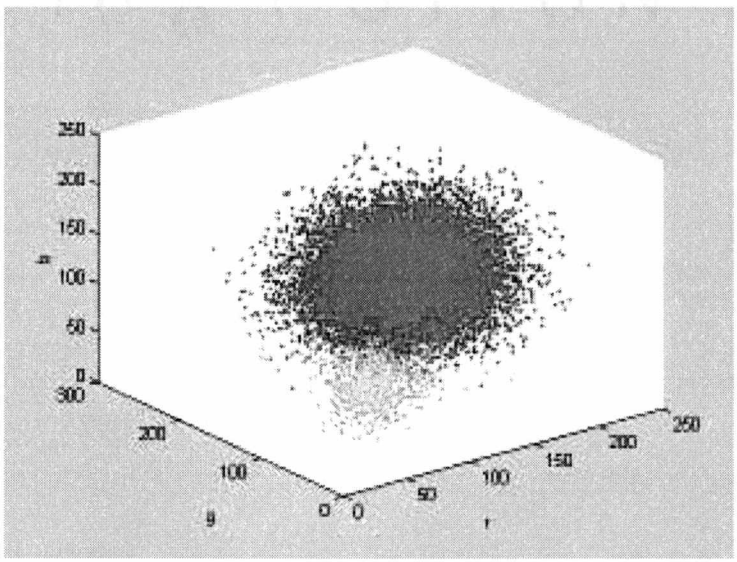


(b) Bayesian Weighted K-Means Classification

Figure 3.6 Feature Space of two cluster example with noise (32, 16)

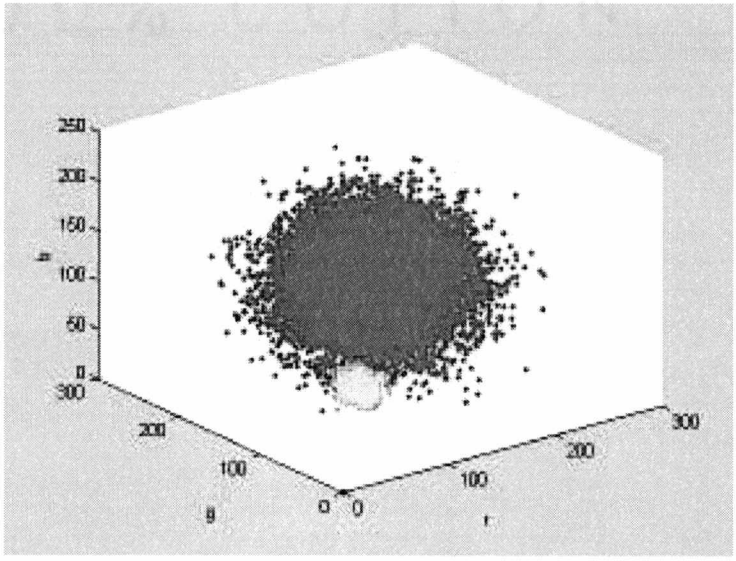


(c) Basic K-Means Classification



(d) Bayesian Learning Classification

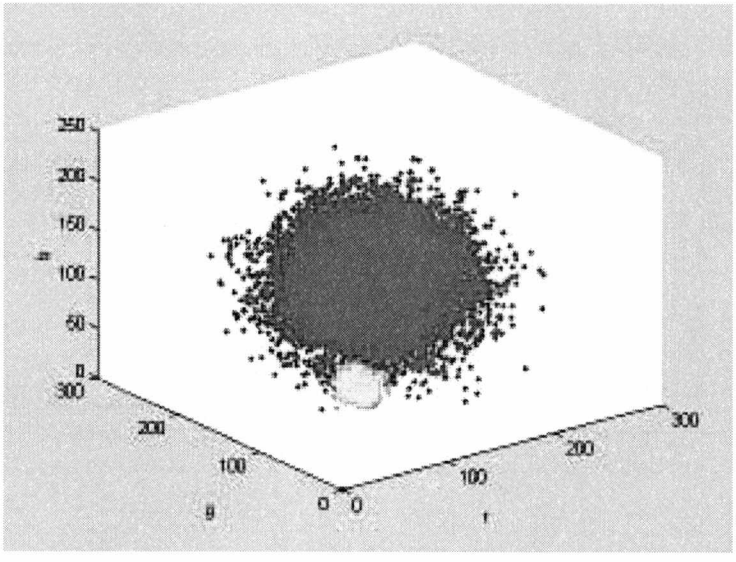
Figure 3.6 (Continued)



(a) Correct Classification

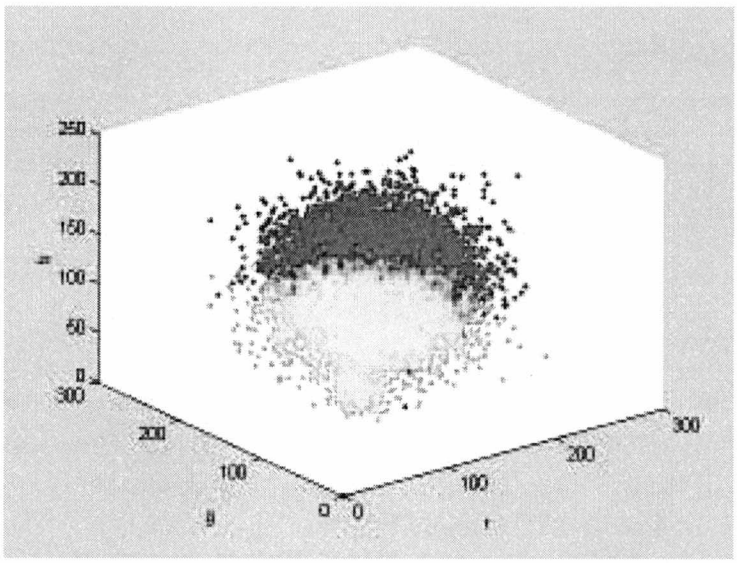
Background Cluster: Red

Foreground Cluster: Green

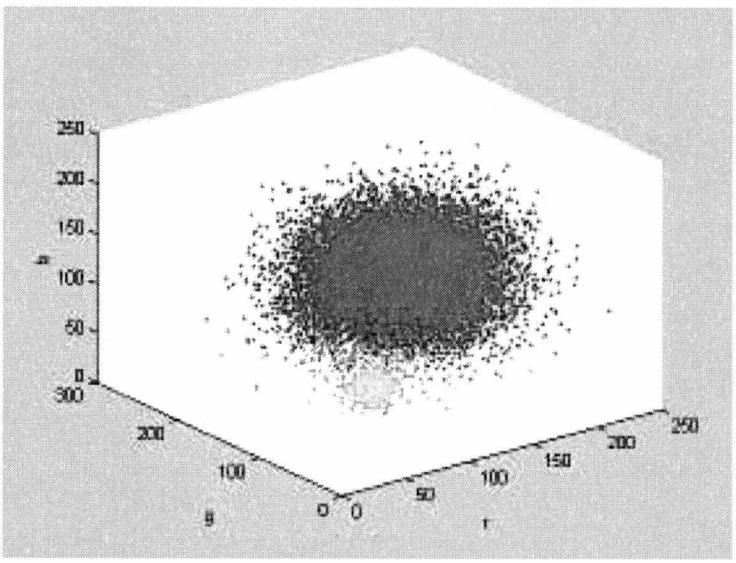


(b) Bayesian Weighted K-Means Classification

Figure 3.7 Feature Space of two clusters example with noise (32, 8)

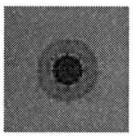


(c) Basic K-Means Classification



(d) Bayesian Learning Classification

Figure 3.7 (Continued)

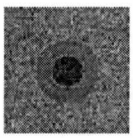


(a) Original Image

Background Gray(128, 128, 128)

Round Object Red(128, 64, 64)

Ring Object Green(64, 128, 64)



(b) Example Test Image

Background Noise = 24

Round Object Noise = 16

Ring Object Noise = 8

Figure 3.8 Input Image and Noisy Example for the Three Clusters Experiments

In the three clusters experiments, each *64 x 64* test image is generated by adding zero-mean Gaussian noise of differing variances to the synthetic image shown in Figure 3.8 (a). The background RGB color of this image is (128,128,128) with probability of occurrence, $P(b) = 0.5$, the ring object RGB color is (128, 64, 64) with probability of occurrence, $P(r) = 0.3$, and the circular object RGB color is (64,128,64) with probability of occurrence, $P(c) = 0.2$. The Bayesian Weighted K-Means, basic K-Means, and Bayesian clustering algorithms are then applied to the test images. The result is compared to a standard classification based on knowledge of correct pixel grouping, and the actual classification error rate is calculated using the equation 3.27. The expected error of the optimal segmentation case is calculated using equation 2.3. The error rates are compared with the expected error to demonstrate and compare algorithmic performance.

Table 3.4 shows the results for a set of three cluster experiments. This series of experiments again demonstrates that the Bayesian Weighted K-Means algorithm

combines a low classification error with computational speed. The average classification improvement over K-Means is 11.21%, while the computation speed is comparable. The average error rate produced by the weighted method is 2.39% lower than that displayed by the learning algorithm, while the computation speed is 5.46 times faster.

To further illustrate the operation of the Bayesian Weighted K-Means algorithm, the results of one of the experiments from table 3.4 are illustrated in detail. Figure 3.9 shows the segmentation results for the case with background, ring object, and round object noise standard deviations of 24, 8, and 16, respectively. Results comparison shows in Table 3.5. The correct feature space clustering and the classification produced using each of the three algorithms are given in Figure 3.10. The final weighting vector produced by the Bayesian Weighted K-Means algorithm reflects the visible variance in the three cluster radii.

Table 3.6 provides a summary of the average computation time and classification errors obtained throughout the course of both the two and three cluster experiments. Note that the Bayesian Weighted K-Means algorithm outperformed the other two algorithms in both speed and accuracy.

Table 3.4 Result Comparison for Three Clusters Experiments

(a) Error Rate Comparison - Three Clusters Experiments

Std. Dev. BG Noise	Std. Dev. Round Noise	Std. Dev. Ring Noise	% Error BWKM	% Error K-Means	% Error BLA	Expected Error, E	Difference BWKM - E
32	32	32	7.788	36.328	7.666	4.458	3.33
32	32	24	6.738	35.889	6.25	4.028	2.71
32	32	16	5.225	28.955	7.715	2.748	2.477
32	32	8	4.834	35.352	9.717	1.585	3.249
32	24	24	6.25	34.106	6.519	4.012	2.238
32	24	16	3.613	33.203	7.739	2.734	0.879
32	24	8	2.808	32.08	9.814	1.57	1.238
32	16	32	9.717	33.179	6.714	3.969	5.748
32	16	24	5.005	32.886	6.812	3.543	1.462
32	16	16	4.126	30.542	7.886	2.267	1.859
32	16	8	2.051	32.69	9.351	1.104	0.947
32	8	32	8.081	30.835	7.544	3.327	4.754
32	8	24	3.613	28.931	6.982	2.902	0.711
32	8	16	2.344	31.104	8.301	1.627	0.717
32	8	8	0.928	29.102	9.79	0.464	0.464
24	32	32	12.451	20.923	5.835	3.474	8.977
24	32	24	4.81	24.39	4.15	2.972	1.838
24	32	16	2.588	20.459	3.613	1.905	0.683
24	32	8	2.759	21.704	4.346	1.148	1.611
24	24	32	6.567	16.089	5.811	3.334	3.233
24	24	24	4.59	21.997	3.955	2.831	1.759
24	24	16	2.393	23.34	2.856	1.771	0.622
24	24	8	1.514	22.095	3.882	1.024	0.49
24	16	32	6.763	16.675	5.518	2.841	3.922
24	16	24	3.687	16.309	2.246	2.354	1.333
24	16	16	1.27	17.041	1.685	1.329	-0.059

Table 3.4 (a) (Continued)

24	16	8	0.952	19.922	3.564	0.596	0.356
24	8	32	6.323	16.992	5.371	2.38	3.943
24	8	24	4.541	17.847	2.515	1.918	2.623
24	8	16	0.83	18.335	1.758	0.916	-0.086
24	8	8	0.195	17.969	3.394	0.184	0.011
16	32	24	2.954	1.563	6.055	1.444	1.51
16	32	16	2.344	1.27	3.613	0.779	1.565
16	32	8	2.026	0.806	2.71	0.465	1.561
16	24	32	2.393	3.003	7.959	1.686	0.707
16	24	24	1.245	1.318	5.151	1.244	0.001
16	24	16	0.708	0.562	2.612	0.603	0.105
16	24	8	0.391	0.439	2.539	0.329	0.062
16	16	32	3.271	2.271	6.763	1.261	2.01
16	16	24	1.685	1.147	4.688	0.864	0.821
16	16	16	0.391	0.537	1.318	0.318	0.073
16	16	8	0.195	0.537	1.099	0.111	0.084
16	8	32	3.296	2.197	6.543	0.965	2.331
16	8	24	2.002	1.392	3.613	0.645	1.357
16	8	16	0.195	0.391	0.928	0.191	0.004
16	8	8	0	0.513	0.244	0.011	-0.011
8	32	24	1.758	1.416	8.398	0.715	1.043
8	32	8	0.488	0.903	3.589	0.08	0.408
8	24	24	0.708	1.318	8.032	0.542	0.166
8	24	16	0.244	0.293	5.029	0.192	0.052
8	24	8	0.024	0.269	3.052	0.035	-0.011
8	16	32	1.367	2.417	9.326	0.446	0.921
8	16	24	0.415	0.879	7.861	0.249	0.166
8	16	16	0	0.122	4.688	0.052	-0.052
8	16	8	0	0.049	2.637	0.003	-0.003
8	8	32	0.952	2.393	8.13	0.163	0.789
8	8	24	0.122	1.025	5.933	0.069	0.053
8	8	16	0	0.049	3.369	0.005	-0.005
8	8	8	0	0	0.342	0	0

Table 3.4 (Continued)

(b) Processing Time Comparison – Three Clusters Experiment

Std. Dev. BG Noise	Std. Dev. Round Noise	Std. Dev. Ring Noise	Processing Time, T1, *BWKM (seconds)	Processing Time, T2, K-Means (seconds)	Ratio T2/T1	Processing Time, T3, BLA (seconds)	Ratio T3/T1
32	32	32	2	4	2	7	3.5
32	32	24	2	3	1.5	7	3.5
32	32	16	3	3	1	7	2.33
32	32	8	3	3	1	7	2.33
32	24	24	2	3	1.5	5	2.5
32	24	16	3	3	1	5	1.67
32	24	8	2	3	1.5	7	3.5
32	16	32	4	3	0.75	8	2
32	16	24	3	2	0.67	7	2.33
32	16	16	3	2	0.67	5	1.67
32	16	8	3	2	0.67	7	2.33
32	8	32	3	2	0.67	7	2.33
32	8	24	7	3	0.43	10	1.43
32	8	16	3	2	0.67	15	5
32	8	8	2	3	1.5	13	6.5
24	32	32	4	4	1	7	1.75
24	32	24	2	4	2	7	3.5
24	32	16	1	4	4	7	7
24	32	8	2	4	2	10	5
24	24	32	3	4	1.33	5	1.67
24	24	24	2	3	1.5	6	3
24	24	16	2	5	2.5	6	3
24	24	8	2	3	1.5	31	15.5
24	16	32	2	4	2	7	3.5
24	16	24	2	3	1.5	7	3.5
24	16	16	2	3	1.5	5	2.5
24	16	8	2	4	2	18	9

Table 3.4 (b) (Continued)

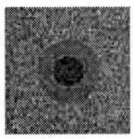
24	8	32	2	3	1.5	12	6
24	8	24	3	3	1	31	10.33
24	8	16	2	4	2	31	15.5
24	8	8	2	3	1.5	31	15.5
16	32	24	2	1	0.5	5	2.5
16	32	16	3	1	0.33	5	1.67
16	32	8	2	2	1	5	2.5
16	24	32	3	1	0.33	5	1.67
16	24	24	1	1	1	5	5
16	24	16	2	1	0.5	7	3.5
16	24	8	1	1	1	9	9
16	16	32	2	1	0.5	6	3
16	16	24	1	1	1	7	7
16	16	16	1	1	1	6	6
16	16	8	0	1	1	15	15
16	8	32	4	1	0.25	7	1.75
16	8	24	2	1	0.5	5	2.5
16	8	16	2	0	0	4	2
16	8	8	2	0	0	4	2
8	32	24	3	1	0.33	7	2.33
8	32	8	2	2	1	5	2.5
8	24	24	1	1	1	9	9
8	24	16	2	1	0.5	10	5
8	24	8	2	0	0	9	4.5
8	16	32	2	1	0.5	9	4.5
8	16	24	1	1	1	11	11
8	16	16	0	1	1	12	12
8	16	8	1	1	1	7	7
8	8	32	1	1	1	12	12
8	8	24	3	1	0.33	7	2.33
8	8	16	1	1	1	31	31
8	8	8	1	1	1	5	5

Table 3.5 Results - Three Clusters Trial – Noise Deviation (24, 8, 16)

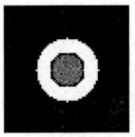
Algorithm	Error %	Time (secs.)	Weight Vector	Final Background Cluster Center	Final Round Object Cluster Center	Final Ring Object Cluster Center
Bayesian Weighted K-Means	1.099	2	(0.509, 0.318, 0.174)	(127.8, 128.3, 128.4)	(127.1, 67.5, 65.6)	(63.5, 128.1, 63.5)
Basic K-Means	15.649	4	(0.333, 0.333, 0.333)	(128.4, 134.1, 133.1)	(132.6, 91.1, 93.4)	(67.2, 128.4, 68.1)
Bayesian Learning	3.564	18	(0.333, 0.333, 0.333)	(129.0, 128.2, 129.2)	(128.5, 64.2, 62.9)	(68.6, 127.9, 68.7)
Expected	0.596			(128, 128, 128)	(128, 64, 64)	(64, 128, 64)

Table 3.6 Summary of Experiments

	Time Complexity	Error Rate	Processing Time
BWKM	$O(N \cdot K \cdot n)$	2.25%	2.83 s
K-Means	$O(N \cdot K \cdot n)$	14.13%	3.37 s
Bayesian Learning	$O(N \times K \cdot n^2)$	4.27%	11.07 s
Expected		1.18%	

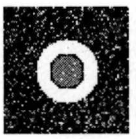


(a) Test Image



(b) Standard Classification

(c) Bayesian Weighted K-Means Classification



(d) Basic K-Means Classification

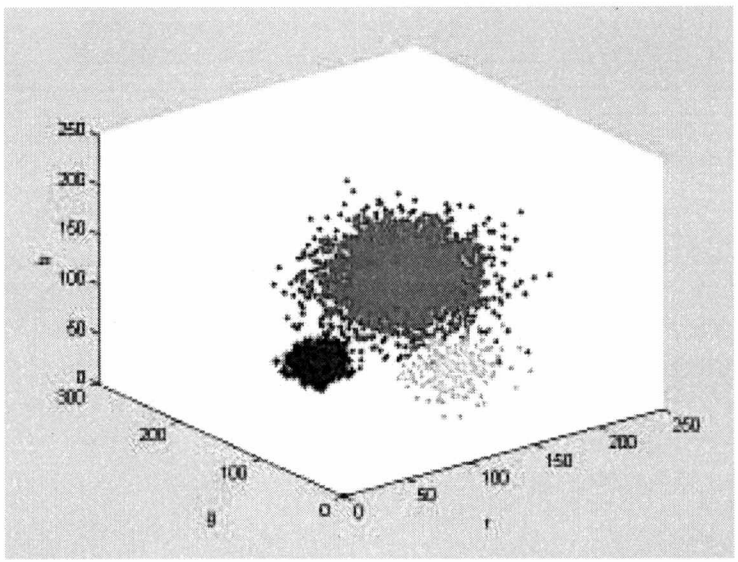
Background Cluster: Black (0)

Round Object Cluster: Gray (1)

Circle Object Cluster: White (2)

(e) Bayesian Learning Classification

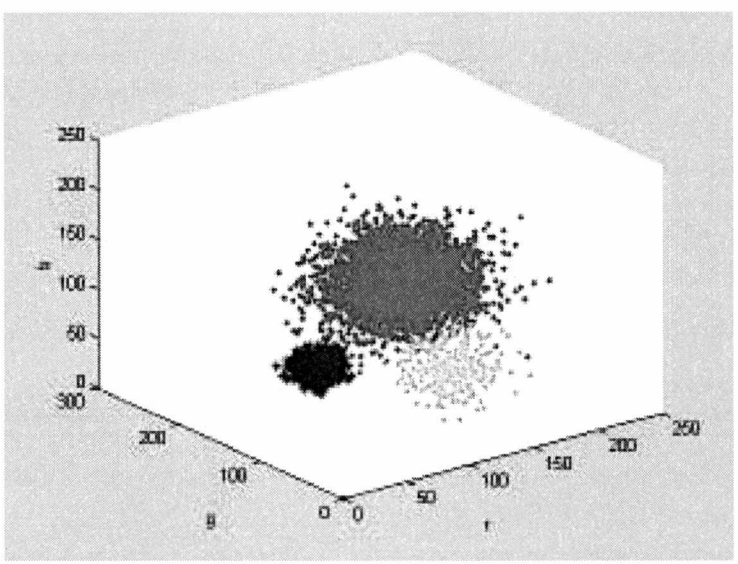
Figure 3.9 Output Images of three clusters example with noise (24, 16, 8)



(a) Correct Classification

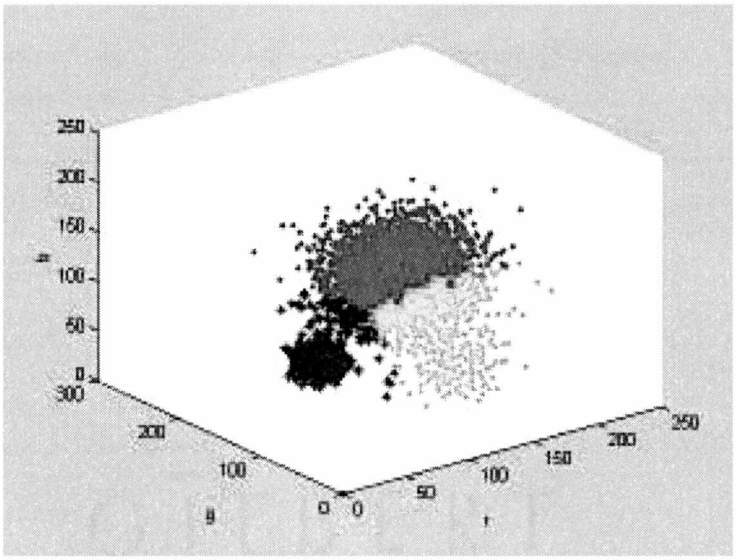
Background Cluster: Red, Round Object Cluster: Green

Ring Object Cluster: Blue

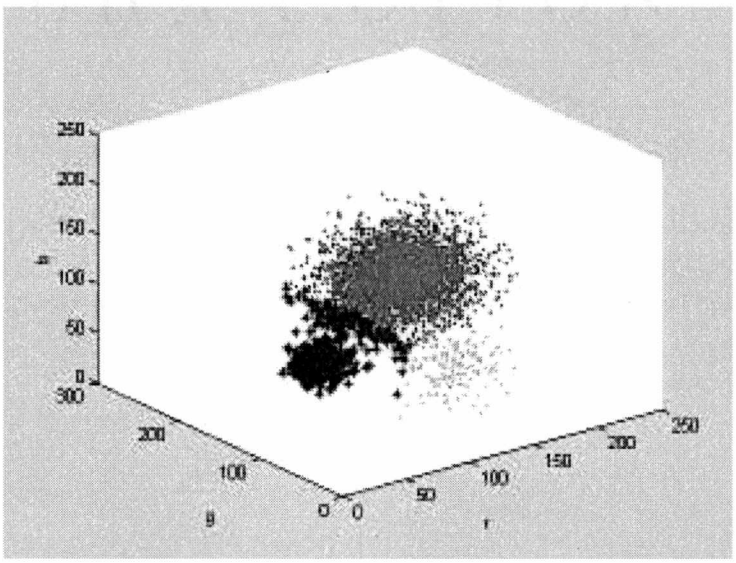


(b) Bayesian Weighted K-Means Classification

Figure 3.10 Feature Space of three clusters example with noise (24, 16, 8)



(c) Basic K-Means Classification



(d) Bayesian Learning Classification

Figure 3.10 (Continued)

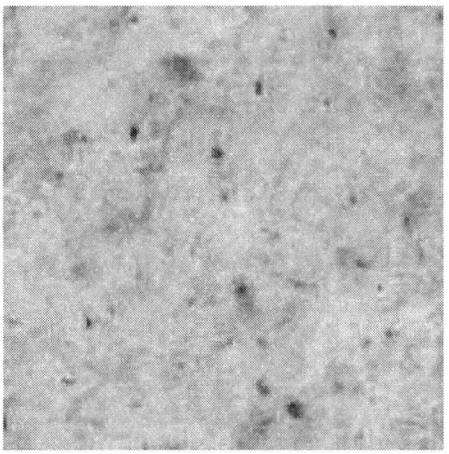
Chapter 4

Image-Based Cotton Trash Measurement

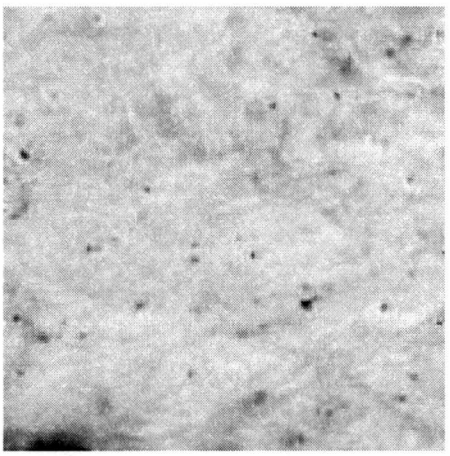
The motivation for development of the Bayesian Weighted K-Means algorithm described in the previous chapter is the construction of an automatic, color-image based system for trash measurement in cotton samples, such as those shown in Figures 4.1. Inspection of these sample images reveals that non-lint material is easy to discern based on color differences. It is also easy to notice the large variance in cotton color by comparing the various cotton images. As discussed in chapter 1, this color variance makes the application of fixed threshold segmentation techniques problematic for this task.

A self-adaptive system for identifying non-lint material in color images of cotton samples based on the Bayesian Weighted K-Means clustering algorithm has been developed. In this system, a color-calibrated scanner is used to return high-resolution RGB images of cotton samples. For other cotton quality measurements, the color data is transformed from the native RGB to the perceptually uniform CIELAB76 [23] color space (see Appendix). The pixels in the image are then classified as either lint or non-lint using the Bayesian Weighted K-Means algorithm and the percentage of total non-lint pixels is calculated.

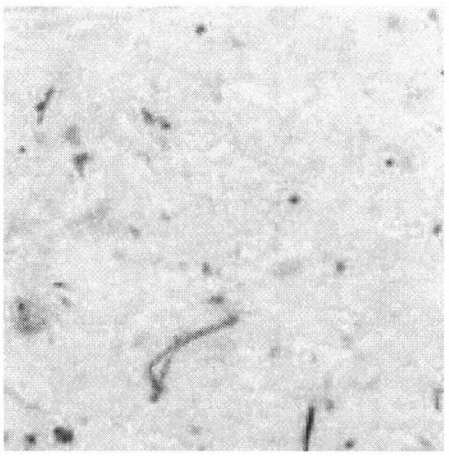
(a) White Cotton Image



(b) Dark Cotton Image



(c) Yellow Cotton Image



(d) White Cotton Image

Figure 4.1 Cotton Images

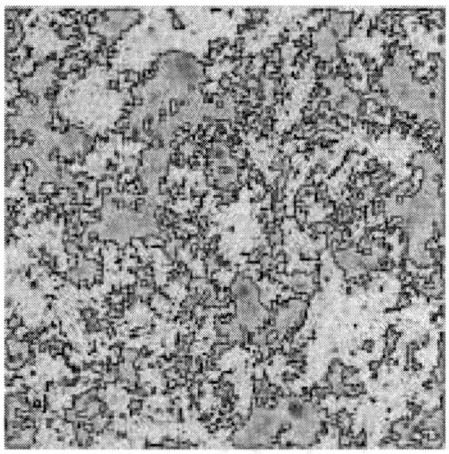
During the development of the trash measurement system, two primary factors had to be considered in addition to the choice of segmentation algorithms. First, the appropriate color space had to be chosen since both RGB and CIELAB data are available for each sample. Second, a method had to be developed for automatically initializing the clusters centers with appropriate model parameters, $\{\mu_k, p_k\}$.

4.1 Color Space Selection

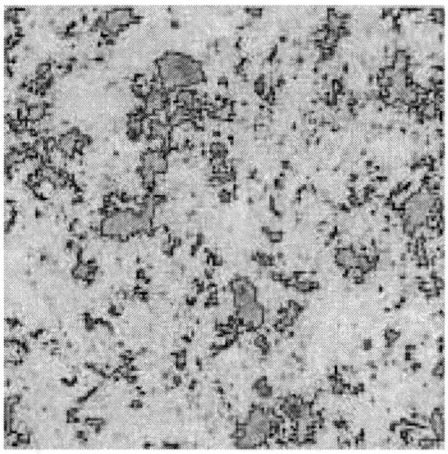
Cluster-based classification should provide results that are more human-like using the CIELAB space due to its perceptual uniformity, but increases the total trash measurement time since the process must wait for the image data transformation to be completed before testing.

Figure 4.2 shows the results of applying both the traditional and Bayesian Weighted K-Means algorithms to a typical cotton image sample stored in native RGB format. Pixels classified as non-lint are located within the blue boundaries. Note that many darker cotton pixels are misclassified as non-lint material and that the Bayesian Weighted approach produces significantly better results.

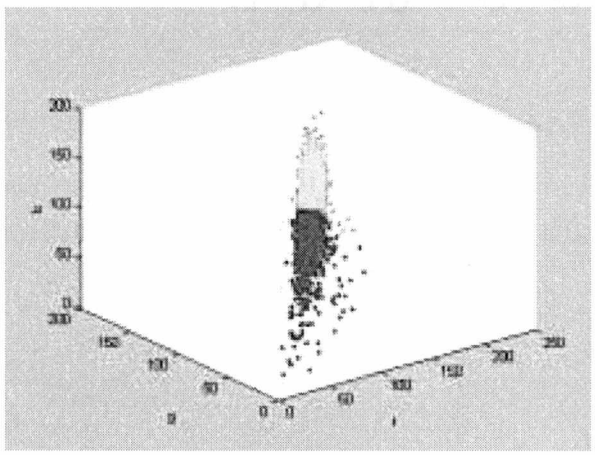
Figure 4.3 shows the results op applying both traditional and Bayesian Weighted K-Means algorithms to the same cotton image sample store in CIELAB space. Pixels classified as non-lint are again shown within the blue boundaries. Note that both algorithms provide much more visually coherent classification, with the Bayesian Weighted technique again out-performing the traditional approach. Based on experimental observations such as these, the current trash measurement system employs CIELAB color data during the segmentation process.



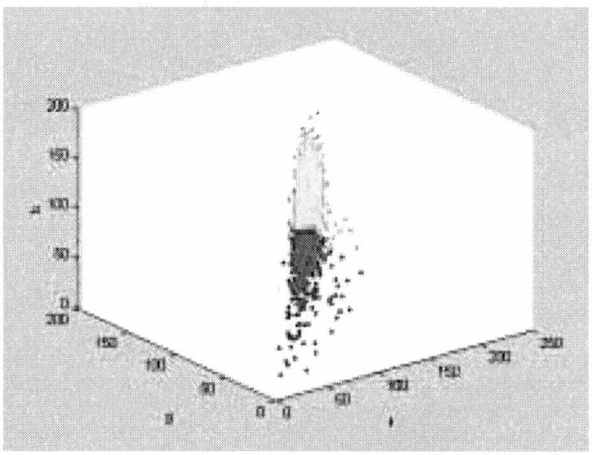
(a) K-Means Clustering Algorithm Classification



(b) Bayesian Weighted K-Means Algorithm Classification

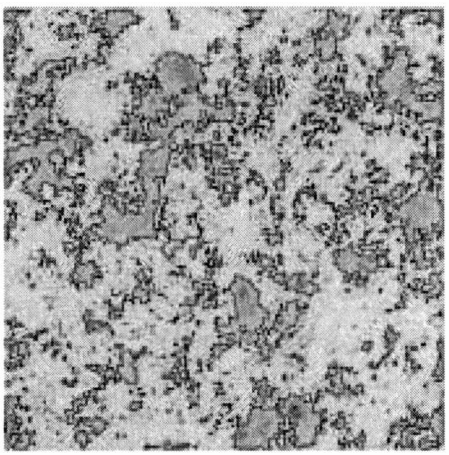


(c) K-Means Algorithm Feature Space

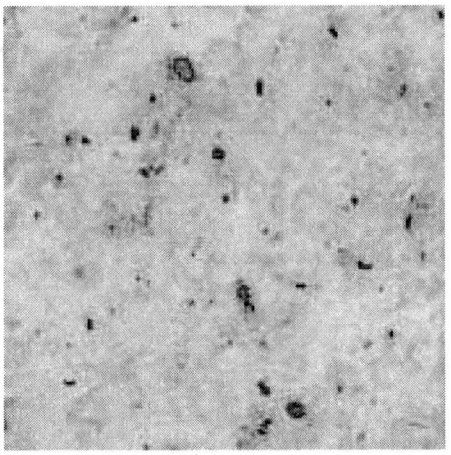


(d) Bayesian Weighted K-Means Algorithm Feature Space

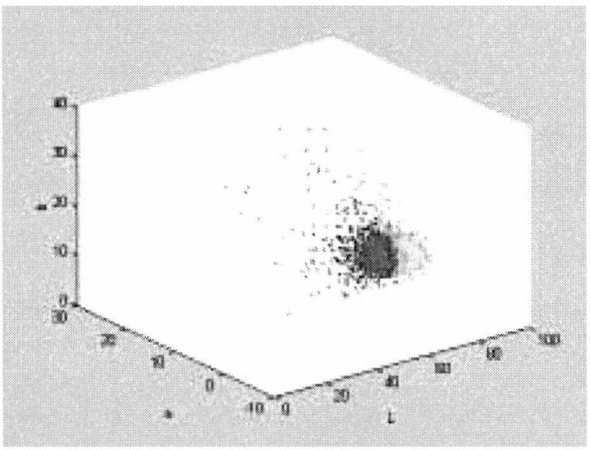
Figure 4.2 Classification Results with RGB Color Space



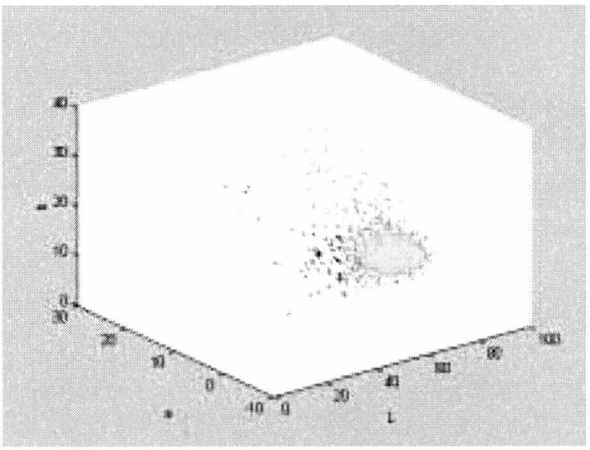
(c) K-Means Clustering Algorithm Classification



(d) Bayesian Weighted K-Means Algorithm Classification



(a) K-Means Algorithm Feature Space



(b) Bayesian Weighted K-Means Algorithm Feature Space

Figure 4.3 Classification Results with CIELAB Color Space

4.2 Cluster Initialization

Both traditional K-Means and Bayesian Weighted K-Means algorithms need the knowledge of initial cluster centers. The estimation of the initial cluster centers is very important for the success of the algorithm. Since there are only two clusters and one of them, the lint cluster, includes over 90% patterns, it is reasonable to use the center of all the patterns as the estimation of the lint cluster center. The initial center of the non-lint cluster is chosen to be the farthest point to the initial center of lint cluster in feature space.

4.3 Results

A complete example using Bayesian Weighted K-Means algorithm with CIELAB color space is showed in Figure 4.4, Figure 4.5. Test image is Yellow Cotton Image in Figure 4.1 (c). Table 4.1 shows the output of algorithm.

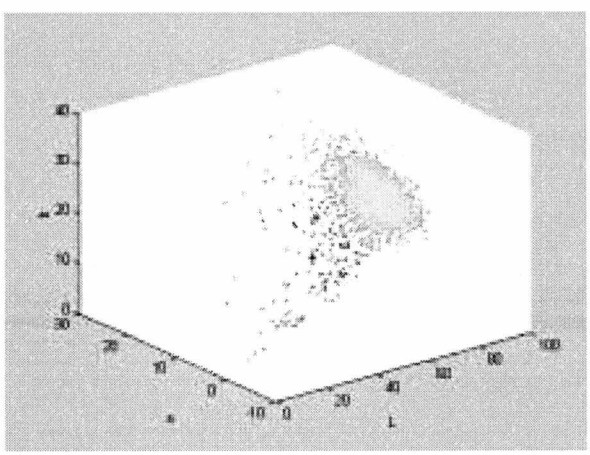


Figure 4.4 Classification of Bayesian Weighted K-Means Algorithm

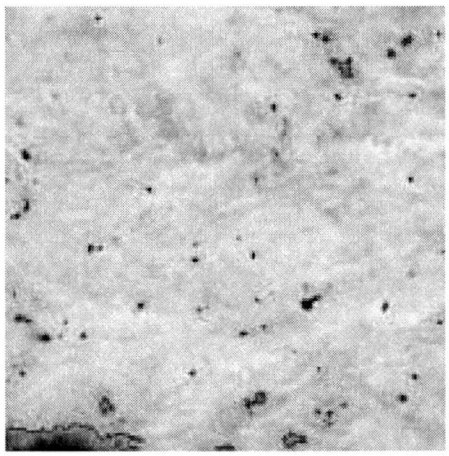


Figure 4.5 Result of Bayesian Weighted K-Means algorithm with CIELAB color space

Table 4.1 Result of Yellow Cotton Image Classification

	non-lint cluster	lint cluster
Initial Centers	23.96, 14.33, 11.83	78.81, 10.14, 20.96
Final Centers	49.30, 8.13, 14.23	78.55, 10.25, 21.47
Weight Vector	0.47	0.53
Iteration Times	8	

Chapter 5

Conclusions

Theoretically, the Bayes learning algorithm discussed in chapter 2 will always provide the optimal solution for any statistical pattern classification problem. However, the learning approach is very sensitive to the accuracy of the *a priori* class probability estimates, and does not always provide solutions with the expected optimal classification errors when applied to discrete data sets. The classification result is optimal only if the *a priori* cluster probabilities are very close to the real values and the pattern data set is extremely large. In most application scenarios, the *a priori* probabilities of clusters are obtained using statistical estimation methods and are not very accurate unless large training data sets are available.

The experimental results shown in chapter 3 demonstrate that pattern classification using the new Bayesian Weighted K-Means algorithm results in equivalent error rates as obtained using the Bayesian learning approach without the need for accurate initial parameters or training data. As shown in table 3.5, the mean classification error for large sets of two and three cluster experiments is actually 2.01% lower than the mean error resulting from application of the Bayesian Learning method. At the same time, the processing was increased by a factor of five. When compared with the traditional K-means method, the Bayesian Weighted K-Means algorithm require essentially the same processing time, while providing a mean classification error reduction of 11.87%. In

addition, the mean classification error of the Weighted K-means approach is only 1.07% higher than the theoretical optimum.

These experimental results are as expected from the detailed analysis of algorithmic complexity. For one iteration of each algorithm, the time complexity for both the traditional and Bayesian Weighted K-Means algorithms is of the order $O(N \cdot K \cdot n)$, while the Bayesian leaning approach is of the order $O(N \cdot K \cdot n^2)$. In the experiments of chapters 3 and 4, the feature space is of dimension, $n = 3$. Thus, the Bayesian Weighted K-means approach would minimally be three times faster than the learning method for the same feature sets. The larger than expected processing time required on average by the Bayesian learning process results from the additional iterations that were often required for convergence.

The Bayesian Weighted K-means algorithm does suffer from two potential scenarios which could result in poor classification performance. First, if the number of clusters K specified by the user is much greater than the actual real value and some of the cluster centers are specified in regions with sparse data, zero size clusters could be produced during the iteration process. In this situation, it will not be possible to find a unique solution for the weighting vector using SVD. Second, each cluster center must be associated with a specific *a priori* probability during algorithm initialization. Unexpected and/or undesirable classification results will be produced if these probability/cluster center pairs are not properly assigned. Thus, future work on the development of this

algorithm will focus on, 1) identifying zero size clusters and eliminating them, and 2) initial probability/cluster center pair assignment.

The Bayesian Weighted K-Means clustering algorithm described in this thesis has been designed for use in the case non-equivalent probability and non-symmetry of pattern classes in feature space. The only assumption required by the technique that each cluster is modeled by a spherical Gaussian distribution. Like traditional K-Means algorithms, the number of clusters and initial cluster centers must be specified prior to application. These choices will directly influence the classification produced by each algorithm. If the Gaussian assumption is satisfied, the number of clusters is not over-specified, and the initial cluster center estimates are close to their actual values, the Bayesian Weighted K-Means algorithm provides close-to-optimal classification solution with computational complexity that is linear with respect to the feature space dimensions.

References

REFERENCES

1. Tou, J. T. and Gonzalez, R. C., Pattern Recognition Principles, Addison-Wesley Publishing Company, Reading, Massachusetts, 1974.
2. R.C. Gonzalez and R.E. Woods. Digital Image Processing. Addison Wesley, 1993.
3. Agricultural Marketing Service. The Classification of Cotton. USDA, 1999.
4. Puzicha, T. Hofmann and J.M. Buhmann, "Histogram Clustering for Unsupervised Image Segmentation", CVPR99 , II, 602-608 ,1999.
5. Felzenszwalb, P., and Huttenlocher, D. 1998. Image segmentation using local variation. In Proceedings of CVPR.
6. P. Schroeter and J. Bigun. Hierarchical image segmentation by multi-dimensional clustering and orientation-adaptive boundary refinement. Pattern Recognition 28 (5) (1995) 695-709
7. Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. In Proceedings of the 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 731--737. IEEE Computer Society Press, June 1997
8. C. Carson, S. Belongie, H. Greenspan, and J. Mlik, "Color- and texture-based images segmentation using em and its application to image querying and classification," Submitted to PAMI, 1998.
9. Ohta, Y.-I.; T. Kanade; and T. Sakai. Color information for region segmentation. Computer Graphics and Image Processing, 13:222--241, 1980.

10. A. Cumani, "Edge detection in multispectral images," Computer Vision, Graphics, and Image Processing. Graphical Models and Image Processing, vol. 53, pp. 40-- 51, Jan. 1991.

11. Yu Xiaohan and Juha Yla-Jaaski. Image segmentation combining region growing and edge detection. In Proc. International Conference on Pattern Recognition, volume III, pages 481--484, 1992.

12. J.C. Tilton. Image Segmentation by Iterative Parallel Region Growing and Splitting. Proceedings of the International Geoscience and Remote Sensing Symposium (IGARSS89), pages 2235-2238, Vancouver, 1989.

13. P. Cheeseman, D. Freeman, J. Kelly, M. Self, J. Stutz, and W. Taylor. Autoclass: a bayesian classification system. In Proceedings of the Fifth International Conference on Machine Learning. Morgan Kaufmann, 1988.

14. M. Celenk, "Color clustering for image segmentation", IEEE Robotics and Automation Megazine, pp. 4--12, March, 1989.

15. E. A. Patrick. Fundamentals on Pattern Recognition. Prentice Hall, Englewood Cliffs N. J., 1972.

16. J.A. Hartigan and Wong M.A. A k-means clustering algorithm. Statistical Algorithms, pages 101--108, 1979.

17. P. S. Bradley and U. M. Fayyad. Refining initial points for k-means clustering. In J. Shavlik, editor, Proceedings of the Fifteenth International Conference on Machine Learning (ICML '98), pages 91--99, San Francisco, CA, 1998.

18. K. Alsabti, S. Ranka, and V. Singh, "An Efficient K-means Clustering Algorithm," Proc. First Workshop on High-Performance Data Mining, 1998.

19. A. Gammerman, editor, Computational Learning and Probabilistic Reasoning. Wiley, 1996.

20. Xu, B., C. Fang, R. Huang, and M.D. Watson. 1997. Chromatic image analysis for cotton trash and color measurements. Text. Res. J. 67:881-890.

21. B. Xu and C. fang; "Clustering Analysis For Cotton Trash Classification", Textile Research Journal, 69(9), 656-662, 1999

22. A. K. Jain, R. Duin, and J. Mao, "Statistical pattern recognition: A review," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 1, pp. 4--38, 2000.

23. S.J. Sangwine and R.E.N. Horne (editors), The Colour Image Processing Handbook, Chapman & Hill, 1998.

Appendix

APPENDIX

COLOR SPACE

Color space or color model is a three-dimensional coordinate system. Each color is represented by a point in the space.

A.1 RGB Color Space

The RGB color space is a basic color space since most image-capture devices and display devices provide RGB signal input or output. It is most frequently used in image processing applications. Each color is described by three components: red, green and blue. The disadvantages of RGB color space include: a) natural image is high correlated between RGB three components: about 0.78 for B-R, 0.98 for R-G and 0.94 for G-B. b) psychological non-intuitivity. c) non-uniformity. Thus, for some applications, such as image compression and image segmentation, it is necessary to convert RGB color space to the other color spaces like CIELAB.

A.2 CIELAB Color Space

The CIELAB color space is a perceptually uniform color space. In this model, the color differences, which are perceived by human eyes, correspond to distances measured colorimetrically. It is defined by following equations:

$$L = 116 \cdot (Y/Y0)^{1/3} - 16$$

$$a = 500 \cdot [\, (X/X0)^{1/3} - (Y/Y0)^{1/3} \,]$$

$$b = 500 \cdot [\,(Y/Y0)^{1/3} - (Z/Z0)^{1/3}\,]$$

where X, Y, Z can be get from RGB values using the following equations:

$$X = 0.490R + 0.310G + 0.200B$$

$$Y = 0.177R + 0.812G + 0.011B$$

$$Z = 0.000R + 0.010G + 0.990B$$

and (X0, Y0, Z0) is reference white point (R=255, G=255, B=255).

Figure A.1 shows the CIELAB color space. L stands for lightness, a for redness-greenness and b for yellowness-blueness. L is orthogonal to a and b. The color difference between two CIELAB coordinates (L_1, a_1, b_1) and (L_2, a_2, b_2) is equals to the Euclidean distance between these two point in CIELAB color space.

$$\Delta E_{Lab} = ((L_1 - L_2)^2 + (a_1 - a_2)^2 + (b_1 - b_2)^2)^{1/2}$$

The ΔE_{Lab} corresponds to human judgments of perceived color difference. Thus, the CIELAB color space is particularly useful in color image segmentation of natural image using clustering algorithms.

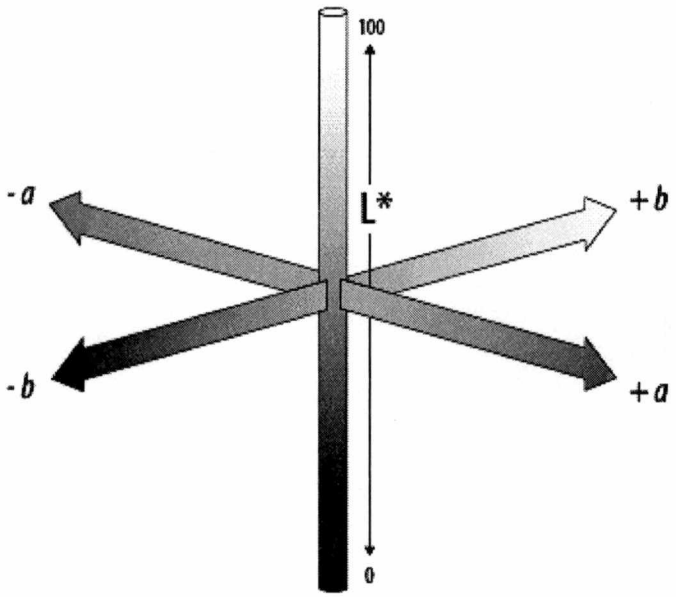


Figure A.1 A three-dimensional representation of the CIELAB color space

From http://www.adobe.com

Vita

Yupeng Zhang was born in Heilongjiang, China in 1974. In July 1997, he graduated from Beijing University of Posts and Telecommunications at Beijing, China with a Bachelor of Science degree in Computer Science. During his career at BUPT, he co-oped for System Architecture Lab, Computer Science Department as a programmer one and half years. He co-oped one semester for Signaling Development Department., Shanghai Bell Telecom Equipment Manufacture Joint Venture Inc., China and finished his senior thesis there. After graduation from college, he started work for Software Development Department, Dalian Telecom, Inc, Dalian, China as a software engineer. He married in December 1998. In June 1999, he came to USA and attended The University of Tennessee at Knoxville in January 2000. After the acceptance of this thesis, he will be leaving UT with his Master of Science degree in Electrical Engineering.