Chapter 3

Problem Definition

This chapter reviews the problems addressed in this thesis in sufficient detail. First the clustering problem is defined and different clustering concepts and approaches are discussed. This is followed by defining image segmentation in addition to presenting various image segmentation methods. A survey of color image quantization and approaches to quantization are then presented. This is followed by a brief introduction to spectral unmixing.

3.1 The Clustering Problem

Data clustering is the process of identifying natural groupings or clusters within multidimensional data based on some similarity measure (e.g. Euclidean distance) [Jain *et al*. 1999; Jain *et al*. 2000]. It is an important process in pattern recognition and machine learning [Hamerly and Elkan 2002]. Furthermore, data clustering is a central process in Artificial Intelligence (AI) [Hamerly 2003]. Clustering algorithms are used in many applications, such as image segmentation [Coleman and Andrews 1979; Jain and Dubes 1988; Turi 2001], vector and color image quantization [Kaukoranta *et al*. 1998; Baek *et al*. 1998; Xiang 1997], data mining [Judd *et al*. 1998], compression [Abbas and Fahmy 1994], machine learning [Carpineto and Romano 1996], etc. A cluster is usually identified by a cluster center (or *centroid*) [Lee and Antonsson 2000]. Data clustering is a difficult problem in unsupervised pattern recognition as the clusters in data may have different shapes and sizes [Jain *et al*. 2000].

3.1.1 Definitions

The following terms are used in this thesis:

- A *pattern* (or *feature vector*), $\boldsymbol{z}$, is a single object or data point used by the clustering algorithm [Jain *et al*. 1999].
- A *feature* (or *attribute*) is an individual component of a pattern [Jain *et al*. 1999].
- A *cluster* is a set of similar patterns, and patterns from different clusters are not similar [Everitt 1974].
- *Hard* (or *Crisp*) clustering algorithms assign each pattern to one and only one cluster.
- *Fuzzy* clustering algorithms assign each pattern to each cluster with some degree of membership.
- A *distance measure* is a metric used to evaluate the similarity of patterns [Jain *et al*. 1999].

The clustering problem can be formally defined as follows (Veenman *et al*. 2003): Given a data set $\boldsymbol{Z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_p, \ldots, \boldsymbol{z}_{N_p}\}$ where $\boldsymbol{z}_p$ is a pattern in the N_d-dimensional feature space, and N_p is the number of patterns in $\boldsymbol{Z}$, then the clustering of $\boldsymbol{Z}$ is the partitioning of $\boldsymbol{Z}$ into K clusters $\{\boldsymbol{C}_1, \boldsymbol{C}_2, \ldots, \boldsymbol{C}_K\}$ satisfying the following conditions:

- Each pattern should be assigned to a cluster, i.e.

  $\cup_{k=1}^{K} \boldsymbol{C}_k = \boldsymbol{Z}$

- Each cluster has at least one pattern assigned to it, i.e.

$\boldsymbol{C}_k \neq \phi, \quad k = 1, \ldots, K$

- Each pattern is assigned to one and only one cluster (in case of hard clustering only), i.e.

  $\boldsymbol{C}_k \cap \boldsymbol{C}_{kk} = \phi$ where $k \neq kk$

3.1.2 Similarity Measures

As previously mentioned, clustering is the process of identifying natural groupings or clusters within multidimensional data based on some similarity measure. Hence, similarity measures are fundamental components in most clustering algorithms [Jain *et al*. 1999].

The most popular way to evaluate a similarity measure is the use of distance measures. The most widely used distance measure is the Euclidean distance defined as

$$d(\boldsymbol{z}_u, \boldsymbol{z}_w) = \sqrt{\sum_{j=1}^{N_d} (z_{u,j} - z_{w,j})^2} = \|\boldsymbol{z}_u - \boldsymbol{z}_w\| \tag{3.1}$$

Euclidean distance is a special case (when $\alpha = 2$) of the Minkowski metric [Jain *et al.* 1999] defined as

$$d^{\alpha}(\boldsymbol{z}_u, \boldsymbol{z}_w) = (\sum_{j=1}^{N_d} (z_{u,j} - z_{w,j})^{\alpha})^{1/\alpha} = \|\boldsymbol{z}_u - \boldsymbol{z}_w\|^{\alpha} \tag{3.2}$$

When $\alpha = 1$, the measure is referred to as the Manhattan distance [Hamerly 2003].

Clustering data of high dimensionality using the Minkowski metric is usually not efficient because the distance between the patterns increases with increase in dimensionality. Hence, the concepts of near and far become weaker [Hamerly 2003]. Furthermore, for the Minkowski metric, the largest-scaled feature tends to dominate the other features. This can be solved by normalizing the features to a common range [Jain *et al.* 1999]. One way to do this is by using the cosine distance (or vector dot product) which is the sum of the product of each component from two vectors defined as

$$< \boldsymbol{z}_u , \boldsymbol{z}_w > = \frac{\sum_{j=1}^{N_d} z_{u,j} z_{w,j}}{\|\boldsymbol{z}_u\| \|\boldsymbol{z}_w\|} \tag{3.3}$$

where $< \boldsymbol{z}_u , \boldsymbol{z}_w > \in [-1,1]$.

The cosine distance is actually not a distance but rather a similarity metric. In other words, the cosine distance measures the difference in the angle between two vectors not the difference in the magnitude between two vectors. The cosine distance is suitable for clustering data of high dimensionality [Hamerly 2003].

Another distance measure is the Mahalanobis distance defined as

$$d_{\mathrm{M}}(\boldsymbol{z}_u , \boldsymbol{z}_w) = (\boldsymbol{z}_u - \boldsymbol{z}_w)\Sigma^{-1}(\boldsymbol{z}_u - \boldsymbol{z}_w)^{\mathrm{T}} \tag{3.4}$$

where Σ is the covariance matrix of the patterns. The Mahalanobis distance gives different features different weights based on their variances and pairwise linear correlations. Thus, this metric implicitly assumes that the densities of the classes are multivariate Gaussian [Jain *et al*. 1999].

3.1.3 Clustering Techniques

Most clustering algorithms are based on two popular techniques known as *hierarchical* and *partitional* clustering [Frigui and Krishnapuram 1999; Leung *et al*. 2000]. In the following, an overview of both techniques is presented with an elaborate discussion of popular hierarchical and partitional clustering algorithms.

3.1.3.1 Hierarchical Clustering Techniques

Algorithms in this category generate a cluster tree (or *dendrogram*) by using heuristic splitting or merging techniques [Hamerly 2003]. A cluster tree is defined as "a tree showing a sequence of clustering with each clustering being a partition of the data set" [Leung *et al*. 2000]. Algorithms that use splitting to generate the cluster tree are called *divisive*. On the other hand, the more popular algorithms that use merging to generate the cluster tree are called *agglomerative*. Divisive hierarchical algorithms start with all the patterns assigned to a single cluster. Then, splitting is applied to a cluster in each stage until each cluster consists of one pattern. Contrary to divisive hierarchical algorithms, agglomerative hierarchical algorithms start with each pattern assigned to one cluster. Then, the two most similar clusters are merged together. This step is repeated until all the patterns are assigned to a single cluster [Turi 2001]. Several agglomerative hierarchical algorithms were proposed in the literature which differ in the way that the two most similar clusters are calculated. The two most popular agglomerative hierarchical algorithms are the *single link* [Sneath and Sokal 1973] and *complete link* [Anderberg 1973] algorithms. Single link algorithms merge the clusters whose distance between their closest patterns is the smallest. Complete link algorithms, on the other hand, merge the clusters whose distance between their most distant patterns is the smallest [Turi 2001]. In general, complete link algorithms generate compact clusters while single link algorithms generate elongated clusters. Thus, complete link algorithms are generally more useful than single link algorithms [Jain *et al*. 1999]. Another less popular agglomerative hierarchical algorithm is the *centroid* method [Anderberg 1973]. The centroid algorithm merges the clusters whose distance between their centroids is the smallest. One disadvantage of the centroid algorithm is that the characteristic of a very small cluster is lost when merged with a very large cluster [Turi 2001]. More details about traditional hierarchical clustering techniques can be found in Everitt [1974].

Recently, a hierarchical clustering approach to simulate the human visual system by modeling the blurring effect of lateral retinal interconnections based on scale space theory has been proposed by Leung *et al*. [2000]. The following paragraph provides the reader with a good idea about this approach as described by Leung *et al*. [2000]:

> "In this approach, a data set is considered as an image with each light point located at a datum position. As we blur this image, smaller light blobs merge into larger ones until the whole image becomes one light blob at a low level of resolution. By identifying each blob with a cluster, the blurring process generates a family of clustering along the hierarchy."

According to Leung *et al*. [2000], this approach has several advantages, including:

- it is not sensitive to initialization,
- it is robust in the presence of noise in the data set, and
- it generates clustering that is similar to that perceived by human eyes.

In general, hierarchical clustering techniques have the following advantages [Frigui and Krishnapuram 1999]:

- the number of clusters need not to be specified *a priori*, and
- they are independent of the initial conditions.

However, hierarchical clustering techniques generally suffer from the following drawbacks:

- They are computationally expensive (time complexity is $O(N_p^2 \log N_p)$ and space complexity is $O(N_p^2)$ [Turi 2001]). Hence, they are not suitable for very large data sets.
- They are static, i.e. patterns assigned to a cluster cannot move to another cluster.
- They may fail to separate overlapping clusters due to a lack of information about the global shape or size of the clusters.

3.1.3.2 Partitional Clustering Techniques

Partitional clustering algorithms divide the data set into a specified number of clusters. These algorithms try to minimize certain criteria (e.g. a square error function) and can therefore be treated as optimization problems. However, these optimization problems are generally NP-hard and combinatorial [Leung *et al*. 2000]. The advantages of hierarchical algorithms are the disadvantages of the partitional algorithms and *vice versa*. Because of their advantages, partitional clustering techniques are more popular than hierarchical techniques in pattern recognition [Jain *et al*. 2000], hence, this thesis concentrates on partitional techniques.

Partitional clustering algorithms are generally iterative algorithms that converge to local optima [Hamerly and Elkan 2002]. Employing the general form of iterative clustering used by Hamerly and Elkan [2002], the steps of an iterative clustering algorithm are:

```
1. Randomly initialize the $K$ cluster centroids
2. **Repeat**
  (a) **For** each pattern, $\boldsymbol{z}_p$, in the data set **do**
        Compute its membership $u(\boldsymbol{m}_k \mid \boldsymbol{z}_p)$ to each centroid $\boldsymbol{m}_k$ and its weight $w(\boldsymbol{z}_p)$
      **endloop**

  (b) Recalculate the $K$ cluster centroids, using
        $\boldsymbol{m}_k = \frac{\sum_{\forall \boldsymbol{z}_p} u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) w(\boldsymbol{z}_p) \boldsymbol{z}_p}{\sum_{\forall \boldsymbol{z}_p} u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) w(\boldsymbol{z}_p)}$    (3.5)
  **until** a stopping criterion is satisfied.
```

In the above algorithm, $u(\boldsymbol{m}_k \mid \boldsymbol{z}_p)$ is the membership function which quantifies the membership of pattern $\boldsymbol{z}_p$ to cluster k. The membership function, $u(\boldsymbol{m}_k \mid \boldsymbol{z}_p)$, must satisfy the following constraints:

1) $u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) \geq 0,\ p = 1,\ldots, N_p$ and $k = 1,\ldots, K$

2) $\sum_{k=1}^{K} u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) = 1,\ p = 1,\ldots, N_p$

Crisp clustering algorithms use a *hard* membership function (i.e. $u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) \in \{0,1\}$), while fuzzy clustering algorithms use a *soft* member function (i.e. $u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) \in [0,1]$) [Hamerly and Elkan 2002].

The weight function, $w(\boldsymbol{z}_p)$, in equation (3.5) defines how much influence pattern $\boldsymbol{z}_p$ has in recomputing the centroids in the next iteration, where $w(\boldsymbol{z}_p) > 0$ [Hamerly and Elkan 2002]. The weight function was proposed by Zhang [2000].

Different stopping criteria can be used in an iterative clustering algorithm, for example:

- stop when the change in centroid values are smaller than a user-specified value,
- stop when the quantization error is small enough, or
- stop when a maximum number of iterations has been exceeded.

In the following, popular iterative clustering algorithms are described by defining the membership and weight functions in equation (3.5).

The K-means Algorithm

The most widely used partitional algorithm is the iterative K-means approach [Forgy 1965]. The objective function that the K-means optimizes is

$$J_{\text{K-means}} = \sum_{k=1}^{K} \sum_{\forall \boldsymbol{z}_p \in \boldsymbol{C}_k} d^2(\boldsymbol{z}_p, \boldsymbol{m}_k) \tag{3.6}$$

Hence, the K-means algorithm minimizes the intra-cluster distance [Hamerly and Elkan 2002]. The K-means algorithm starts with K centroids (initial values for the centroids are randomly selected or derived from *a priori* information). Then, each pattern in the data set is assigned to the closest cluster (i.e. closest centroid). Finally, the centroids are recalculated according to the associated patterns. This process is repeated until convergence is achieved.

The membership and weight functions for K-means are defined as

$$u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) = \begin{cases} 1 & \text{if } d^2(\boldsymbol{z}_p, \boldsymbol{m}_k) = \arg\min_k \{d^2(\boldsymbol{z}_p, \boldsymbol{m}_k)\} \\ 0 & \text{otherwise} \end{cases} \tag{3.7}$$

$$w(\boldsymbol{z}_p) = 1 \tag{3.8}$$

Hence, K-means has a hard membership function. Furthermore, K-means has a constant weight function, thus, all patterns have equal importance [Hamerly and Elkan 2002].

The K-means algorithm has the following main advantages [Turi 2001]:

- it is very easy to implement, and
- its time complexity is O(N_p) making it suitable for very large data sets.

However, the K-means algorithm has the following drawbacks [Davies 1997]:

- the algorithm is data-dependent,

- it is a greedy algorithm that depends on the initial conditions, which may cause the algorithm to converge to suboptimal solutions, and
- the user needs to specify the number of clusters in advance.

The K-medoids algorithm is similar to K-means with one major difference, namely, the centroids are taken from the data itself [Hamerly 2003]. The objective of K-medoids is to find the most centrally located patterns within the clusters [Halkidi *et al.* 2001]. These patterns are called *medoids*. Finding a single medoid requires $O(N_p^2)$. Hence, K-medoids is not suitable for moderately large data sets.

The Fuzzy C-means Algorithm

A fuzzy version of K-means, called Fuzzy C-means (FCM) (sometimes called fuzzy K-means), was proposed by Bezdek [1980; 1981]. FCM is based on a fuzzy extension of the least-square error criterion. The advantage of FCM over K-means is that FCM assigns each pattern to each cluster with some degree of membership (i.e. fuzzy clustering). This is more suitable for real applications where there are some overlaps between the clusters in the data set. The objective function that the FCM optimizes is

$$J_{\text{FCM}} = \sum_{k=1}^{K} \sum_{p=1}^{N_p} u_{k,p}^{q} d^2(\boldsymbol{z}_p, \boldsymbol{m}_k) \tag{3.9}$$

where q is the fuzziness exponent, with $q \geq 1$. Increasing the value of q will make the algorithm more fuzzy; $u_{k,p}$ is the membership value for the pth pattern in the kth cluster satisfying the following constraints:

1) $u_{k,p} \geq 0$, $p = 1,\ldots, N_p$ and $k = 1,\ldots, K$

2) $\sum_{k=1}^{K} u_{k,p} = 1$, $p = 1,\ldots, N_p$

The membership and weight functions for FCM are defined as [Hamerly and Elkan 2002]

$$u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) = \frac{\|\boldsymbol{z}_p - \boldsymbol{m}_k\|^{-2/(q-1)}}{\sum_{k=1}^{K} \|\boldsymbol{z}_p - \boldsymbol{m}_k\|^{-2/(q-1)}} \tag{3.10}$$

$$w(\boldsymbol{z}_p) = 1 \tag{3.11}$$

Hence, FCM has a soft membership function and a constant weight function. In general, FCM performs better than K-means [Hamerly 2003] and it is less affected by the presence of uncertainty in the data [Liew *et al*. 2000]. However, as in K-means it requires the user to specify the number of clusters in the data set. In addition, it may converge to local optima [Jain *et al*. 1999].

Krishnapuram and Keller [1993; 1996] proposed a possibilistic clustering algorithm, called *possibilistic* C-means. Possibilistic clustering is similar to fuzzy clustering; the main difference is that in possibilistic clustering the membership values may not sum to one [Turi 2001]. Possibilistic C-means works well in the presence of noise in the data set. However, it has several drawbacks, namely [Turi 2001],

- it is likely to generate coincident clusters,
- it requires the user to specify the number of clusters in advance,
- it converges to local optima, and
- it depends on initial conditions.

The Gaussian Expectation-Maximization Algorithm

Another popular clustering algorithm is the Expectation-Maximization (EM) algorithm [McLachlan and Krishnan 1997; Rendner and Walker 1984; Bishop 1995]. EM is used for parameter estimation in the presence of some unknown data [Hamerly 2003]. EM partitions the data set into clusters by determining a mixture of Gaussians fitting the data set. Each Gaussian has a mean and covariance matrix [Alldrin *et al.* 2003]. The objective function that the EM optimizes as defined by Hamerly and Elkan [2002] is

$$J_{\text{EM}} = -\sum_{p=1}^{N_p} \log\left(\sum_{k=1}^{K} p(\boldsymbol{z}_p \mid \boldsymbol{m}_k)\, p(\boldsymbol{m}_k)\right) \tag{3.12}$$

where $p(\boldsymbol{z}_p \mid \boldsymbol{m}_k)$ is the probability of $\boldsymbol{z}_p$ given that it is generated by a Gaussian distribution with centroid $\boldsymbol{m}_k$, and $p(\boldsymbol{m}_k)$ is the prior probability of centroid $\boldsymbol{m}_k$.

The membership and weight functions for EM are defined as [Hamerly and Elkan 2002]

$$u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) = \frac{p(\boldsymbol{z}_p \mid \boldsymbol{m}_k)\, p(\boldsymbol{m}_k)}{p(\boldsymbol{z}_p)} \tag{3.13}$$

$$w(\boldsymbol{z}_p) = 1 \tag{3.14}$$

Hence, EM has a soft membership function and a constant weight function. The algorithm starts with an initial estimate of the parameters. Then, an *expectation* step is applied where the known data values are used to compute the expected values of the unknown data [Hamerly 2003]. This is followed by a *maximization* step where the known and expected values of the data are used to generate a new estimate of the parameters. The expectation and maximization steps are repeated until convergence.

Results from Veenman *et al.* [2002] and Hamerly [2003] showed that K-means performs comparably to EM. Furthermore, Aldrin *et al.* [2003] stated that EM fails on high-dimensional data sets due to numerical precision problems. They also observed that Gaussians often collapsed to delta functions [Alldrin *et al.* 2003]. In addition, EM depends on the initial estimate of the parameters [Hamerly 2003; Turi 2001] and it requires the user to specify the number of clusters in advance. Moreover, EM assumes that the density of each cluster is Gaussian which may not always be true [Ng *et al.* 2001].

The K-harmonic Means Algorithm

Recently, Zhang and colleagues [1999; 2000] proposed a novel algorithm called K-harmonic means (KHM), with promising results. In KHM, the harmonic mean of the distance of each cluster center to every pattern is computed. The cluster centroids are then updated accordingly. The objective function that the KHM optimizes is

$$J_{\text{KHM}} = \sum_{p=1}^{N_p} \frac{K}{\sum_{k=1}^{K} \frac{1}{\left\| \boldsymbol{z}_p - \boldsymbol{m}_k \right\|^{\alpha}}} \tag{3.15}$$

where α is a user-specified parameter, typically $\alpha \geq 2$.

The membership and weight functions for KHM are [Hamerly and Elkan 2002]

$$u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) = \frac{\left\| \boldsymbol{z}_p - \boldsymbol{m}_k \right\|^{-\alpha-2}}{\sum_{k=1}^{K} \left\| \boldsymbol{z}_p - \boldsymbol{m}_k \right\|^{-\alpha-2}} \tag{3.16}$$

$$w(\boldsymbol{z}_p) = \frac{\sum_{k=1}^{K} \left\| \boldsymbol{z}_p - \boldsymbol{m}_k \right\|^{-\alpha-2}}{\left(\sum_{k=1}^{K} \left\| \boldsymbol{z}_p - \boldsymbol{m}_k \right\|^{-\alpha} \right)^2} \tag{3.17}$$

Hence, KHM has a soft membership function and a varying weight function. KHM assigns higher weights for patterns that are far from all the centroids to help the centroids in covering the data [Hamerly and Elkan 2002].

Contrary to K-means, KHM is less sensitive to initial conditions and does not have the problem of collapsing Gaussians exhibited by EM [Alldrin *et al*. 2003]. Experiments conducted by Zhang *et al*. [1999], Zhang [2000] and Hamerly and Elkan [2002] showed that KHM outperformed K-means, FCM (according to Hamerly and Elkan [2002]) and EM.

Hybrid 2

Hamerly and Elkan [2002] proposed a variation of KHM, called Hybrid 2 (H2), which uses the soft membership function of KHM (i.e. equation (3.16)) and the constant weight function of K-means (i.e. equation (3.8)). Hamerly and Elkan [2002] showed that H2 outperformed K-means, FCM and EM. However, KHM, in general, performed slightly better than H2.

K-means, FCM, EM, KHM and H2 are linear time algorithms (i.e. their time complexity is O(N_p)) making them suitable for very large data sets. According to

Hamerly [2003], FCM, KHM and H2 - all use soft membership functions - are the best available clustering algorithms.

Non-iterative Partitional Algorithms

Another category of unsupervised partitional algorithms includes the non-iterative algorithms. The most widely used non-iterative algorithm is MacQueen's K-means algorithm [MacQueen 1967]. This algorithm works in two phases: the first phase finds the centroids of the clusters, and the second clusters the patterns. Competitive Learning (CL) updates the centroids sequentially by moving the closest centroid toward the pattern being classified [Scheunders, A Comparison 1997]. These algorithms suffer the drawback of being dependent on the order in which the data points are presented. To overcome this problem, data points are presented in a random order [Davies 1997]. In general, iterative algorithms are more effective than non-iterative algorithms, since they are less dependent on the order in which data points are presented.

3.1.3.3 Other Clustering Techniques

Another type of clustering algorithms includes the *Nearest Neighbor* clustering algorithm proposed by Lu and Fu [1978]. For each unclassified pattern, the algorithm finds the nearest classified pattern whose distance from the unclassified pattern is less than a pre-specified threshold. The unclassified pattern is then assigned to the cluster of the classified pattern. This process is repeated until all the patterns become classified or no further assignments can occur [Jain *et al.* 1999].

Recently, a new type of clustering algorithms called *spectral* clustering algorithms [Ng *et al*. 2001; Bach and Jordan 2003] has been proposed by computer vision researchers and graph theorists. Spectral clustering is based on spectral graph theory [Chung 1997] where a graph representing the data (the graph is analogous to a matrix of the distance between the patterns in the data set) is searched by the spectral clustering algorithm for globally optimal cuts [Hamerly 2003]. One major advantage of spectral clustering is that it can generate arbitrary-shaped clusters. However, spectral clustering suffers from two major drawbacks [Hamerly 2003]:

- It is computationally expensive (its time complexity is $O(N_p^3 + N_d N_p^2)$). Hence, they are not suitable for moderately large data sets.
- It requires the user to specify a kernel width parameter which has a profound effect on the result of the spectral clustering algorithm. Choosing a good value for this parameter is usually difficult.

The *mean shift* algorithm [Comaniciu and Meer 2002] also automatically finds the number of clusters in a data set and can work with arbitrary shaped clusters. The mean shift algorithm starts with a number of kernel estimators in the input space. These estimators are then repeatedly moved towards areas of higher density. When all the kernels reached stability, all the kernels that are near to each other are grouped together. The data is then segmented based on where each kernel started.

The mean shift algorithm has the following problems, [Hamerly 2003]:

- it has to find a way to group kernels and patterns, and

- as in spectral clustering, the mean shift algorithm requires the user to specify a kernel width parameter which has a profound effect on the result of the algorithm.

3.1.4 Clustering Validation Techniques

The main objective of cluster validation is to evaluate clustering results in order to find the best partitiong of a data set [Halkidi *et al*. 2001]. Hence, cluster validity approaches are used to quantitatively evaluate the result of a clustering algorithm [Halkidi *et al*. 2001]. These approaches have representative indices, called *validity indices*. The traditional approach to determine the "optimum" number of clusters is to run the algorithm repetitively using different input values and to select the partitioning of data resulting in the best validity measure [Halkidi and Vazirgiannis 2001].

Two criteria that have been widely considered sufficient in measuring the quality of data partitioning, are [Halkidi *et al*. 2001]

- *Compactness*: patterns in one cluster should be similar to each other and different from patterns in other clusters. The variance of patterns in a cluster gives an indication of compactness.
- *Separation*: clusters should be well-separated from each other. The Euclidean distance between cluster centroids gives an indication of cluster separation.

There are several validity indices; a thorough survey of validity indices can be found in Halkidi *et al*. [2001]. In the following, some representative indices are discussed.

Dunn [1974] proposed a well known cluster validity index that identifies compact and well separated clusters. The main goal of Dunn's index is to maximize inter-cluster distances (i.e. separation) while minimizing intra-cluster distances (i.e. increase compactness). The Dunn index is defined as

$$D = \min_{k=1,\ldots,K} \left\{ \min_{kk=k+1,\ldots,K} \left(\frac{dist(\mathbf{C}_k, \mathbf{C}_{kk})}{\max\limits_{a=1,\ldots,K} diam(\mathbf{C}_a)} \right) \right\} \tag{3.18}$$

where $dist(\mathbf{C}_k, \mathbf{C}_{kk})$ is the dissimilarity function between two clusters $\mathbf{C}_k$ and $\mathbf{C}_{kk}$ defined as

$$dist(\mathbf{C}_k, \mathbf{C}_{kk}) = \min_{\mathbf{u} \in \mathbf{C}_k, \mathbf{w} \in \mathbf{C}_{kk}} d(\mathbf{u}, \mathbf{w}),$$

where $d(\mathbf{u}, \mathbf{w})$ is the Euclidean distance between $\mathbf{u}$ and $\mathbf{v}$; $diam(\mathbf{C})$ is the diameter of a cluster, defined as

$$diam(\mathbf{C}) = \max_{\mathbf{u}, \mathbf{w} \in \mathbf{C}} d(\mathbf{u}, \mathbf{w})$$

An "optimal" value of K is the one that maximizes the Dunn's index. Dunn's index suffers from the following problems [Halkidi *et al*. 2001]:

- it is computationally expensive, and
- it is sensitive to the presence of noise.

Several Dunn-like indices were proposed in Pal and Biswas [1997] to reduce the sensitivity to the presence of noise.

Another well known index, proposed by Davies and Bouldin [1979], minimizes the average similarity between each cluster and the one most similar to it. The Davies and Bouldin index is defined as

$$DB = \frac{1}{K}\sum_{k=1}^{K} \max_{\substack{kk=1,\ldots,K\\ k\neq kk}} \left(\frac{diam(\boldsymbol{C}_k) + diam(\boldsymbol{C}_{kk})}{dist(\boldsymbol{C}_k, \boldsymbol{C}_{kk})} \right) \tag{3.19}$$

An "optimal" value of K is the one that minimizes the DB index.

Recently, Turi [2001] proposed an index incorporating a multiplier function (to penalize the selection of a small number of clusters) to the ratio between intra-cluster and inter-cluster distances, with some promising results. The index is defined as

$$V = (c \times \mathrm{N}(2,1) + 1) \times \frac{\text{intra}}{\text{inter}} \tag{3.20}$$

where c is a user specified parameter and N(2,1) is a Gaussian distribution with mean 2 and standard deviation of 1. The "intra" term is the average of all the distances between each data point and its cluster centroid, defined as

$$\text{intra} = \frac{1}{N_p}\sum_{k=1}^{K}\sum_{\forall \boldsymbol{u}\in \boldsymbol{C}_k} \left\| \boldsymbol{u} - \boldsymbol{m}_k \right\|^2$$

This term is used to measure the compactness of the clusters. The "inter" term is the minimum distance between the cluster centroids, defined as

$$\text{inter} = min\{\left\| \boldsymbol{m}_k - \boldsymbol{m}_{kk} \right\|^2\}, \forall\, k = 1,\ldots,K-1 \text{ and } kk = k+1,\ldots,K.$$

This term is used to measure the separation of the clusters. An "optimal" value of K is the one that minimizes the V index.

According to Turi [2001], this index performed better than both Dunn's index and the index of Davies and Bouldin on the tested cases.

Two recent validity indices are *S_Dbw* [Halkidi and Vazirgiannis 2001] and *CDbw* [Halkidi and Vazirgiannis 2002]. *S_Dbw* measures the compactness of a data set by the cluster variance, whereas separation is measured by the density between clusters. The *S_Dbw* index is defined as

$$S\_Dbw = scat(K) + Dens\_bw(K) \tag{3.21}$$

The first term is the average scattering of the clusters which is a measure of compactness of the clusters, defined as

$$scat(K) = \frac{1}{K}\sum_{k=1}^{K} \|\sigma(\boldsymbol{C}_k)\| / \|\sigma(\boldsymbol{Z})\|$$

where $\sigma(\boldsymbol{C}_k)$ is the variance of cluster $\boldsymbol{C}_k$ and $\sigma(\boldsymbol{Z})$ is the variance of data set $\boldsymbol{Z}$; $||\boldsymbol{z}||$ is defined as $||\boldsymbol{z}|| = (\boldsymbol{z}^{\mathrm{T}}\boldsymbol{z})^{1/2}$, where $\boldsymbol{z}$ is a vector.

The second term in equation (3.21) evaluates the density of the area between the two clusters in relation to the density of the two clusters. Thus, the second term is a measure of the separation of the clusters, defined as

$$Dens\_bw(K) = \frac{1}{K(K-1)}\sum_{k=1}^{K}\left[\sum_{\substack{kk=1\\ k\neq kk}}^{K} \frac{density(\boldsymbol{b}_{k,kk})}{max\{density(\boldsymbol{C}_k), density(\boldsymbol{C}_{kk})\}}\right]$$

where $\boldsymbol{b}_{k,kk}$ is the middle point of the line segment defined by $\boldsymbol{m}_k$ and $\boldsymbol{m}_{kk}$. The term *density*($\boldsymbol{b}$) is defined as

$$density(\boldsymbol{b}) = \sum_{ll=1}^{n_{k,kk}} f(\boldsymbol{z}_{ll}, \boldsymbol{b})$$

where $n_{k,kk}$ is the total number of patterns in clusters $\boldsymbol{C}_k$ and $\boldsymbol{C}_{kk}$ (i.e. $n_{k,kk} = n_k + n_{kk}$). The function $f(\boldsymbol{z},\boldsymbol{b})$ is defined as

$$f(\boldsymbol{z},\boldsymbol{b}) = \begin{cases} 0 & \text{if } d(\boldsymbol{z},\boldsymbol{b}) > \sigma \\ 1 & \text{otherwise} \end{cases}$$

where

$$\sigma = \frac{1}{K}\sqrt{\sum_{k=1}^{K}\|\sigma(\boldsymbol{C}_k)\|}$$

An "optimal" value of *K* is the one that minimizes the *S_Dbw* index. Halkidi and Vazirgiannis [2001] showed that, in tested cases, *S_Dbw* successfully found the "optimal" number of clusters whereas other well-known indices often failed to do so. However, *S_Dbw* does not work properly for arbitrary shaped clusters.

To address this problem, Halkidi and Vazirgiannis [2002] proposed a multi-representative validity index, *CDbw*, in which each cluster is represented by a user-specified number of points, instead of one representative as is done in *S_Dbw*. Furthermore, *CDbw* uses intra-cluster density to measure the compactness of a data set, and uses the density between clusters to measure their separation.

More recently, Veenman *et al.* [2002; 2003] proposed a validity index that minimizes the intra-cluster variability while constraining the intra-cluster variability of the union of the two clusters. The sum of squared error is used to minimize the intra-cluster variability while a minimum variance for the union of two clusters is used to implement the joint intra-cluster variability. The index is defined as

$$IV = min\sum_{k=1}^{K} n_k Var(\boldsymbol{C}_k) \tag{3.22}$$

where n_k is the number of patterns in cluster $\boldsymbol{C}_k$ and

$$Var(\boldsymbol{C}_k) = \frac{1}{n_k}\sum_{\boldsymbol{z}_p \in \boldsymbol{C}_k}\|\boldsymbol{z}_p - \boldsymbol{m}_k\|^2$$

such that

$$Var(\boldsymbol{C}_k \cup \boldsymbol{C}_{kk}) \geq \sigma^2_{\max}, \quad \forall \boldsymbol{C}_k, \boldsymbol{C}_{kk}, k \neq kk$$

where $\sigma^2_{\max}$ is a user-specified parameter. This parameter has a profound effect on the final result.

The above validity indices are suitable for hard clustering. Validity indices have been developed for fuzzy clustering. The interested reader is referred to Halkidi *et al*. [2001] for more information.

3.1.5 Determining the Number of Clusters

Most clustering algorithms require the number of clusters to be specified in advance [Lee and Antonsson 2000; Hamerly and Elkan 2003]. Finding the "optimum" number of clusters in a data set is usually a challenge since it requires *a priori* knowledge, and/or ground truth about the data, which is not always available. The problem of finding the optimum number of clusters in a data set has been the subject of several research efforts [Halkidi *et al*. 2001; Theodoridis and Koutroubas 1999], however, despite the amount of research in this area, the outcome is still unsatisfactory [Rosenberger and Chehdi 2000]. In the literature, many approaches to dynamically find the number of clusters in a data set were proposed. In this section, several dynamic clustering approaches are presented and discussed.

ISODATA (Iterative Self-Organizing Data Analysis Technique), proposed by Ball and Hall [1967], is an enhancement of the K-means algorithm (K-means is sometimes referred to as *basic* ISODATA [Turi 2001]). ISODATA is an iterative procedure that assigns each pattern to its closest centroids (as in K-means). However, ISODATA has the ability to merge two clusters if the distance between their centroids is below a user-specified threshold. Furthermore, ISODATA can split elongated clusters into two clusters based on another user-specified threshold. Hence, a major advantage of ISODATA compared to K-means is the ability to determine the number of clusters in a data set. However, ISODATA requires the user to specify the values of several parameters (e.g. the merging and splitting thresholds). These parameters have a profound effect on the performance of ISODATA making the result subjective [Turi 2001].

Dynamic Optimal Cluster-seek (DYNOC) [Tou 1979] is a dynamic clustering algorithm which is similar to ISODATA. DYNOC maximizes the ratio of the minimum inter-cluster distance to the maximum intra-cluster distance. This is done by an iterative procedure with the added capability of splitting and merging. However, as in ISODATA, DYNOC requires the user to specify a value for a parameter that determines whether splitting is needed [Turi 2001].

Snob [Wallace 1984; Wallace and Dowe 1994] uses various methods to assign objects to clusters in an intelligent manner [Turi 2001]. After each assignment, a means of model selection called the Wallace Information Measure (also known as the Minimum Message Length) [Wallace and Boulton 1968; Oliver and Hand 1994] is calculated and based on this calculation the assignment is accepted or rejected. Snob can split/merge and move points between clusters, thereby allowing it to determine the number of clusters in a data set.

Bischof *et al*. [1999] proposed an algorithm based on K-means which uses a similar concept to the Wallace Information Measure called the Minimum Description Length [Rissanen 1978] framework. The algorithm starts with a large value for K and proceeds to remove centroids when this removal results in a reduction of the description length. K-means is used between the steps that reduce K.

Modified Linde-Buzo-Gray (MLBG), proposed by Rosenberger and Chehdi [2000], improves K-means by automatically finding the number of clusters in data set by using intermediate results. MLBG is an iterative procedure that starts with K clusters. In each iteration, a cluster, $\boldsymbol{C}_k$, maximizing an intra-cluster distance measure is chosen for splitting. Two centroids are generated from the splitting process. The first centroid, $\boldsymbol{m}_1$, is initialized to the centroid of the original cluster, $\boldsymbol{C}_k$. The second cluster centroid, $\boldsymbol{m}_2$, is chosen to be the pattern in $\boldsymbol{C}_k$ which is the most distant from $\boldsymbol{m}_1$. K-means is then applied on the new K+1 centroids. The new set of centroids is accepted if it satisfies an evaluation criterion based on a dispersion measure. This process is repeated until no valid partition of the data can be obtained. One of the main problems with MLBG is that it requires the user to specify the values of four parameters, which have a profound effect on the resultant number of clusters.

Pelleg and Moore [2000] proposed another K-means based algorithm, called X-means that uses model selection. X-means starts by setting the number of clusters, K, to be the minimum number of clusters in the data set (e.g. K = 1). Then, K-means is applied on the K clusters. This is followed by a splitting process based on the Bayesian Information Criterion (BIC) [Kass and Wasserman 1995] defined as

$$BIC(\boldsymbol{C} \mid \boldsymbol{Z}) = \hat{l}(\boldsymbol{Z} \mid \boldsymbol{C}) - \frac{K(N_d + 1)}{2} \log N_p \tag{3.23}$$

where $\hat{l}(\boldsymbol{Z} \mid \boldsymbol{C})$ is the log-likelihood of the data set $\boldsymbol{Z}$ according to model $\boldsymbol{C}$. If the splitting process improves the BIC score the resulting split is accepted, otherwise it is rejected. Other scoring functions can also be used.

These two steps are repeated until a user-specified upper bound of K is reached. X-means searches over the range of values of K and reports the value with the best BIC score.

Recently, Huang [2002] proposed SYNERACT as an alternative approach to ISODATA. SYNERACT combines K-means with hierarchical descending approaches to overcome the drawbacks of K-means mentioned previously. Three concepts used by SYNERACT are:

- a hyperplane to split up a cluster into two smaller clusters and compute their centroids,
- iterative clustering to assign pixels into available clusters, and
- a binary tree to store clusters generated from the splitting process.

According to Huang [2002], SYNERACT is faster than and almost as accurate as ISODATA. Furthermore, it does not require the number of clusters and initial location of centroids to be specified in advance. However, SYNERACT requires the user to specify the values of two parameters that affect the splitting process.

Veenman *et al*. [2002] proposed a partitional clustering algorithm that finds the number of clusters in a data set by minimizing the clustering validity index defined in equation (3.22). This algorithm starts by initializing the number of clusters equal to the number of patterns in the data set. Then, iteratively, the clusters are split or merged according to a series of tests based on the validity index. According to Veenman *et al*. [2002], the proposed approach performed better than both K-means and EM algorithms. However, the approach suffers from the following drawbacks, namely

- it is computationally expensive, and
- it requires the user to specify a parameter for the validity index (already discussed in Section 3.1.4) which has a significant effect on the final results (although the authors provide a method to help the user in finding a good value for this parameter).

More recently, Hamerly and Elkan [2003] proposed another approach based on K-means, called G-means. G-means starts with a small value for *K*, and with each iteration splits up the clusters whose data do not fit a Gaussian distribution. Between each round of splitting, K-means is applied to the entire data set in order to refine the current solution. According to Hamerly and Elkan [2003], G-means works better than X-means, however, it works only for data having spherical and/or elliptical clusters. G-means is not designed to work for arbitrary-shaped clusters [Hamerly 2003].

Gath and Geva [1989] proposed an unsupervised fuzzy clustering algorithm based on a combination of FCM and fuzzy maximum likelihood estimation. The algorithm starts by initializing *K* to a user-specified lower bound of the number of clusters in the data set (e.g. $K = 1$). A modified FCM (that uses an unsupervised learning process to initialize the *K* centroids) is first applied to cluster the data. Using the resulting centroids, a fuzzy maximum likelihood estimation algorithm is then applied. The fuzzy maximum likelihood estimation algorithm uses an "exponential" distance measure based on maximum likelihood estimation [Bezdek 1981] instead of the Euclidean distance measure, because the exponential distance measure is more suitable for hyper-ellipsoidal clusters. The quality of the resulting clusters is then evaluated using a clustering validity index that is mainly based on a hyper-volume criterion which measures the compactness of a cluster. *K* is then incremented and the algorithm is repeated until a user-specified upper bound of *K* is reached. The value of *K* resulting in the best value of the validity index is considered to be the "optimal" number of clusters in the data set. Gath and Geva [1989] stated that their algorithm works well in cases of large variability of cluster shapes. However, the algorithm becomes more sensitive to local optima as the complexity increases. Furthermore, because of the exponential function, floating point overflows may occur [Su 2002].

Lorette *et al.* [2000] proposed an algorithm based on fuzzy clustering to dynamically determine the number of clusters in a data set. In this thesis, the proposed algorithm is referred as the Unsupervised Fuzzy Clustering (UFC) algorithm. A new objective function was proposed for this purpose, defined as

$$J_{\text{UFC}} = \sum_{k=1}^{K} \sum_{p=1}^{N_p} u_{k,p}^{q} d^2(\boldsymbol{z}_p, \boldsymbol{m}_k) - \beta \sum_{k=1}^{K} p_k \log(p_k) \qquad (3.24)$$

where q is the fuzziness exponent, $u_{k,p}$ is the membership value for the pth pattern in the kth cluster, β is a parameter that decreases as the run progresses, and p_k is the *a priori* probability of cluster $\boldsymbol{C}_k$ defined as

$$p_k = \frac{1}{N_p} \sum_{p=1}^{N_p} u_{k,p} \qquad (3.25)$$

The first term of equation (3.24) is the objective function of FCM which is minimized when each cluster consists of one pattern. The second term is an entropy term that is minimized when all the patterns are assigned to one cluster. Lorette *et al.* [2000] use this objective function to derive new update equations for the membership and centroid parameters.

The algorithm starts with a large number of clusters. Then, the membership values and centroids are updated using the new update equations. This is followed by applying equation (3.25) to update the *a priori* probabilities. If $p_k < \varepsilon$ then cluster k is discarded; ε is a user-specified parameter. This procedure is repeated until convergence. The drawback of this approach is that it requires the parameter ε to be specified in advance. The performance of the algorithm is sensitive to the value of ε.

Similar to UFC, Boujemaa [2000] proposed an algorithm, based on a generalization of the competitive agglomeration clustering algorithm introduced by Frigui and Krishnapuram [1997].

The fuzzy algorithms discussed above modify the objective function of FCM. In general, these approaches are sensitive to initialization and other parameters [Frigui and Krishnapuram 1999]. Frigui and Krishnapuram [1999] proposed a robust competitive clustering algorithm based on the process of competitive agglomeration. The algorithm starts with a large number of small clusters. Then, during the execution of the algorithm, adjacent clusters compete for patterns. Clusters losing the competition will eventually disappear [Frigui and Krishnapuram 1999]. However, this algorithm also requires the user to specify a parameter that has a significant effect on the generated result.

3.1.6 Clustering using Self-Organizing Maps

Kohonen's Self Organizing Maps (SOM) [Kohonen 1995] can be used to automatically find the number of clusters in a data set. The objective of SOM is to find regularities in a data set without any external supervision [Pandya and Macy 1996]. SOM is a single-layered unsupervised artificial neural network where input patterns are associated with output nodes via weights that are iteratively modified until a stopping criterion is met [Jain *et al.* 1999]. SOM combines competitive learning (in which different nodes in the Kohonen network compete to be the winner when an input pattern is presented) with a topological structuring of nodes, such that adjacent nodes tend to have similar weight vectors (this is done via lateral feedback)

[Mehrotra *et al.* 1997; Pandya and Macy 1996]. A general pseudo-code of SOM [Pandya and Macy 1996] is shown in Figure 3.1.

```
Let $\eta(t)$ be the learning rate parameter and $\Delta_w(t)$ be the neighborhood function
Randomly initialize the weight vectors, $\boldsymbol{w}_k(0)$
Initialize the learning rate $\eta(0)$ and the neighborhood function $\Delta_w(0)$
Repeat
  For each input pattern $\boldsymbol{z}_p$ do
    Select the node whose weight vector is closest (in terms of Euclidean distance) to
    $\boldsymbol{z}_p$ as the winning node

    Use competitive learning to train the weight vectors such that all the nodes within
    the neighborhood of the winning node are moved toward $\boldsymbol{z}_p$:

    $\boldsymbol{w}_k(t+1) = \begin{cases} \boldsymbol{w}_k(t) + \eta(t)[\boldsymbol{z}_p - \boldsymbol{w}_k(t)] & k \in \Delta_w(t) \\ \boldsymbol{w}_k(t) & \text{otherwise} \end{cases}$
  Endloop
  Linearly decrease $\eta(t)$ and reduce $\Delta_w(t)$
Until some convergence criteria are satisfied
```

Figure 3.1: General pseudo-code for SOM

In Figure 3.1, $\eta(t)$ starts relatively large (e.g. close to 1) then linearly decreases until it reaches a small user-specified value. The neighborhood function $\Delta_w(t)$ defines the neighborhood size surrounding the winning node. A large value of $\Delta_w(t)$ is used at the beginning of the training. This value is then reduced as the training progresses in order to get sharper clusters [Pandya and Macy 1996]. A typical neighborhood arrangement is the rectangular lattice shown in Figure 3.2 [Pandya and Macy 1996].

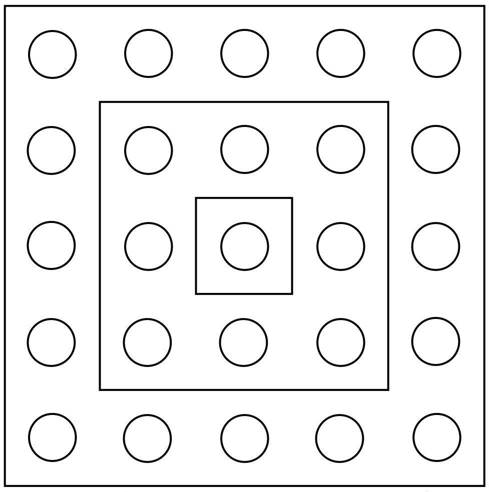
Figure 3.2: Rectangular Lattice arrangement of neighborhoods

SOM suffers from the following drawbacks [Jain *et al*. 1999]:

- It depends on the initial conditions.
- Its performance is affected by the learning rate parameter and the neighborhood function.
- It works well with hyper-spherical clusters only.
- It uses a fixed number of output nodes.
- It depends on the order in which the data points are presented. To overcome this problem, the choice of data points can be randomized during each iteration [Pandya and Macy 1996].

3.1.7 Clustering using Stochastic Algorithms

Simulated annealing (discussed in Section 2.3) has been used for clustering [Klein and Dubes 1989]. In general, a simulated annealing based clustering algorithm works as shown in Figure 3.3 [Jain *et al*. 1999].

```
An initial partition $P_0$ of the data set is randomly chosen
Repeat
  A neighbor of $P_0$ is chosen
  If the new partition is better than $P_0$ then
    move to the new partition
  Else
    move to the new partition with a probability that decreases as the algorithm
    progresses.
Until a stopping criterion is satisfied
```

Figure 3.3: General simulated annealing based clustering algorithm

One problem with simulated annealing is that it is very slow in finding an optimal solution [Jain *et al*. 1999].

Tabu search (discussed in Section 2.3) has also been used for hard clustering [Al-Sultan 1995] and fuzzy clustering [Delgado *et al*. 1997] with encouraging results. A hybrid approach combining both K-means and tabu search that performs better than both K-means and tabu search was proposed by Frnti *et al*. [1998]. Recently, Chu and Roddick [2003] proposed a hybrid approach combining both tabu search and simulated annealing that outperforms the hybrid proposed by Frnti *et al*. [1998].

However, the performance of simulated annealing and tabu search depends on the selection of several control parameters [Jain *et al*. 1999].

Most clustering approaches discussed so far perform local search to find a solution to a clustering problem. Evolutionary algorithms (discussed in Section 2.4) which perform global search have also been used for clustering [Jain *et al*. 1999]. Raghavan and Birchand [1979] used GAs to minimize the squared error of a clustering solution. In this approach, each chromosome represents a partition of N_p patterns into K clusters. Hence, the size of each chromosome is N_p. This representation has a major drawback in that it increases the search space by a factor of $K!$. The crossover operator may also result in inferior offspring [Jain *et al*. 1999].

Babu and Murty [1993] proposed a hybrid approach combining K-means and GAs that performed better than the GA. In this approach, a GA is only used to feed K-means with good initial centroids [Jain *et al*. 1999].

Recently, Maulik and Bandyopadhyay [2000] proposed a GA-based clustering where each chromosome represents K centroids. Hence, a floating point representation is used. The fitness function is defined as the inverse of the objective function of K-means (refer to equation (3.6)). The GA-based clustering algorithm is summarized in Figure 3.4.

According to Maulik and Bandyopadhyay [2000], this approach outperformed K-means on the tested cases. One drawback of this approach is that it requires the user to specify the number of clusters in advance.

```
1. Initialize each chromosome to contain $K$ randomly chosen centroids from the
   data set
2. For $t = 1$ to $t_{\max}$
(a) For each chromosome $i$
    (i) Assign each pattern to the cluster with the closest centroid
    (ii) Recalculate the $K$ cluster centroids of chromosome $i$ as the means of their
         patterns
    (iii) Calculate the fitness of chromosome $i$

(b) Apply roulette wheel selection
(c) Apply single point crossover with probability $p_c$
(d) Apply mutation with probability $p_m$. The mutation operator is defined as
    $\boldsymbol{x} = \boldsymbol{x} \pm (r + \gamma)\boldsymbol{x}$
    where $r \sim U(0,1)$ and $\gamma$ is a user-specified parameter such that $\gamma \in (0,1)$
```

Figure 3.4: General pseudo-code for GA-based clustering algorithm

Lee and Antonsson [2000] used an evolution strategy (ES) to dynamically cluster a data set. The proposed ES implemented variable length individuals to search for both the centroids and the number of clusters. Each individual represents a set of centroids. The length of each individual is randomly chosen from a user-specified range of cluster numbers. The centroids of each individual are then randomly initialized. Mutation is applied to the individuals by adding/subtracting a Gaussian random variable with zero mean and unit standard deviation. Two point crossover is also used as a "length changing operator". A (10+60) ES selection is used where 10 is the number of parents and 60 is the number of offspring generated in each generation. The best ten individuals from the set of parents and offspring are used for the next generation. A modification of the mean square error is used as the fitness function, defined as

$$J_{\text{ES}} = \sqrt{K+1}\sum_{k=1}^{K}\sum_{\forall z_p \in C_k} d(\boldsymbol{z}_p, \boldsymbol{m}_k) \tag{3.26}$$

The modification occurs by multiplying the mean square error by a constant corresponding to the square root of the number of clusters. This constant is used to penalize a large value of *K*. According to Lee and Antonsson [2000], the results are promising. However, the proposed algorithm needs to be compared with other dynamic clustering approaches and its performance needs to be investigated as the dimension increases.

In general, evolutionary approaches have several advantages, namely [Jain *et al.* 1999]:

- they are global search approaches,
- they are suitable for parallel processing, and
- they can work with a discontinuous criterion function.

However, evolutionary approaches generally suffer from the following drawbacks [Jain *et al.* 1999]:

- they require the user to specify the values of a set of parameters (e.g. population size, p_c, p_m, etc.) for each specific problem, and

- the execution time of EAs is significantly higher than the execution time of other traditional clustering algorithms (e.g. K-means and FCM), especially when applied to large data sets.

3.1.8 Unsupervised Image Classification

Image classification is the process of identifying groups of similar image primitives [Puzicha *et al*. 2000]. These image primitives can be pixels, regions, line elements and so on, depending on the problem encountered.

There are two main approaches to image classification: supervised and unsupervised. In the supervised approach, the number and the numerical characteristics (e.g. mean and variance) of the classes in the image are known in advance (by the analyst) and used in the training step, which is followed by the classification step. There are several popular supervised algorithms such as the minimum-distance-to-mean, parallelepiped and the Gaussian maximum likelihood classifiers [Lillesand and Kiefer 1994]. In the unsupervised approach the classes are unknown and the approach starts by partitioning the image data into groups (or clusters), according to a similarity measure, which can be compared with reference to data by an analyst and used to segment the image.

Therefore, unsupervised classification is a special case of the general clustering problem where the data set is an image (or a set of images) and the patterns are the pixels of the image(s).

In general, the unsupervised approach has several advantages over the supervised approach, namely [Davies 1997]

- For unsupervised approaches, there is no need for an analyst to specify in advance all the classes in the image data set. The clustering algorithm automatically finds distinct clusters, which dramatically reduces the work of the analyst.
- The characteristics of the objects being classified can vary with time; the unsupervised approach is an excellent way to monitor these changes.
- Some characteristics of objects may not be known in advance. The unsupervised approach automatically flags these characteristics.

3.2 Image Segmentation using Clustering

Image segmentation is a fundamental process in several image processing and computer vision applications. It can be considered as the first low-level processing step in image processing and pattern recognition [Cheng *et al.* 2001]. Image segmentation is defined as the process of dividing an image into disjoint homogenous regions. These homogenous regions should represent objects or parts of them [Lucchese and Mitra 2001]. The homogeneity of the regions is measured using some image property (e.g. pixel intensity) [Jain *et al.* 1999]. Image segmentation can be formally defined as follows:

Given an image I and a homogeneity predicate P. The segmentation of image I is the partitioning of I into K regions, $\{R_1, R_2,\ldots,R_K\}$, satisfying the following conditions:

- Each pixel in the image should be assigned to a region, i.e.

  $$\cup_{k=1}^{K} R_k = I$$

- Each pixel is assigned to one and only one region, i.e.

$R_k \cap R_{kk} = \phi$ where $k \neq kk$

- Each region satisfies homogeneity predicate *P*, i.e.

  $P(R_k) = \text{True}, \quad \forall\, k = 1,\ldots,K$

- Two different regions can not satisfy *P*, i.e.

  $P(R_k \cup R_{kk}) = \text{False}$ where $k \neq kk$

There are many techniques for image segmentation in the literature; details can be found in Fu and Mui [1981], Pal and Pal [1993], Cheng *et al.* [2001], Lucchese and Mitra [2001] and Turi [2001]. In general, these techniques can be categorized into thresholding, edge-based, region growing and clustering techniques [Turi 2001]. Each of these categories are discussed in the following sections.

3.2.1 Thresholding Techniques

Thresholding [Gonzalez and Woods 1992; Jain *et al.* 1995] is the simplest image segmentation technique. In its simplest version an image is divided into two segments: object and background by specifying a threshold. A pixel above the threshold is assigned to one segment and a pixel below the threshold is assigned to the other segment. For more sophisticated images multiple thresholds can be used.

3.2.2 Edge-based Techniques

In edge-based techniques [Gonzalez and Woods 1992; Jain *et al.* 1995; Kwok and Constantinides 1997], segmentation is achieved by finding the edges of the regions.

This is usually accomplished by moving a mask (e.g. a 3×3 window) over the image to detect local changes in the image intensity.

3.2.3 Region growing Techniques

In region growing [Gonzalez and Woods 1992; Jain *et al*. 1995; Fuh *et al*. 2000], a set of seed pixels are chosen. Neighboring pixels of a seed are agglomerated if they satisfy a homogeneity criterion. This is repeated until no more pixels can be added to the region. This approach has some problems [Turi 2001]:

- The selection of the seed pixels which is not a straightforward task.
- The selection of the homogeneity criterion.

Region splitting and merging divide the image into regions. A region is then split if it does not satisfy a homogeneity condition. Regions can also be merged if their merging results in a region that satisfies some condition. This is repeated until no more splitting and merging can occur [Gonzalez and Woods 1992].

3.2.4 Clustering Techniques

Image segmentation can be treated as a clustering problem where features describing each pixel correspond to a pattern and an image region (i.e. segment) corresponds to a cluster [Jain *et al*. 1999]. This similarity is obvious by comparing the clustering problem definition (refer to section 3.1.1) and the image segmentation problem definition (refer to section 3.2). Therefore, clustering algorithms have been widely used to solve the problem of image segmentation (e.g. K-means [Tou and Gonzalez

1974], FCM [Trivedi and Bezdek 1986], ISODATA [Tou and Gonzalez 1974] and snob [Wallace and Dowe 1994]). However, it should be noted that the number of clusters is usually not known *a priori* in image segmentation. Therefore, clustering algorithms that do not require the user to specify the number of clusters are usually preferred.

In this thesis, the clustering problem and the image segmentation problem are considered to be similar. Thus, algorithms are proposed for both problems interchangeably. In the following, several representative clustering-based techniques are presented.

A hybrid approach combining agglomerative hierarchical clustering and region-based segmentation was proposed by Amadasun and King [1988]. The image is first divided into regions. Homogenous regions are specified and mean feature vectors are then determined for each homogenous region. The most similar mean feature vectors are merged. This process is repeated until the specified number of clusters is reached. One advantage of this approach is that it is computationally efficient, because hierarchical clustering is applied on the mean feature vectors instead of the image pixels. However, this approach has several drawbacks, namely [Turi 2001],

- it requires the user to specify the number of clusters in advance,
- it depends on the region size, and
- it depends on the used homogeneity criterion.

Clustering algorithms are usually applied to feature space, and as such they do not use any spatial information (e.g. the relative location of the patterns in the feature space). However, for image segmentation spatial information is important because pixels with similar features are usually found near each other in the spatial domain [Liew *et al.* 2000]. To address this issue, a generalization of K-means that is adaptive and includes spatial information was proposed by Pappas [1992]. In this approach, *a posteriori* probability function is defined which constrains the region intensity and imposes spatial continuity [Turi 2001]. The iterative algorithm alternates between maximizing the *a posteriori* probability function and calculating the cluster centroids. The cluster centroids are initially equal to the K-means cluster centroids. The centroids are updated by averaging them over a sliding window. The size of the sliding window is progressively decreases [Lucchese and Mitra 2001]. Chang *et al.* [1994] extends this algorithm to color image segmentation. Saber *et al.* [1996] extends the approach of Chang *et al.* by proposing a hybrid approach combining color image segmentation and edge linking. Chen *et al.* [1998] applied an approach similar to Pappas [1992] to biomedical images. A drawback of the generalization of K-means approaches is that they require the user to specify the number of clusters in advance [Turi 2001].

A color map image segmentation algorithm combining FCM and a supervised neural network was proposed by Wu *et al.* [1994]. FCM is first applied giving a set of prototypes satisfying some validation criteria. A neural network with supervised learning is then used to optimize these prototypes. The optimized prototypes are used to segment the image using the nearest neighbor rule [Turi 2001].

A fuzzy image clustering algorithm which incorporates spatial contextual information was proposed by Liew *et al.* [2000]. A dissimilarity measure which considers the eight neighboring pixels of each pixel was proposed. The dissimilarity measure is adaptive in the sense that the effect of the neighboring pixels is suppressed in nonhomogenous image regions. In addition, a merging process that merges clusters based on their closeness and their degree of overlap is also used to determine the "optimal" number of clusters. According to Liew *et al.* [2000], due to the incorporation of spatial information, this approach is faster, less sensitive to noise and more suitable for arbitrary shaped clusters than FCM.

Lim and Lee [1990] proposed a two-stage process called *thresholding and FCM*. In the first stage, a *coarse* segmentation is obtained by smoothing the histogram of each color component by a Gaussian convolution. Thresholds are set as the valleys of the smoothed histograms (the valleys are obtained using the first and second derivative of the smoothed histograms). A safe area around each threshold is determined. Each pixel outside these safe areas is assigned to a cluster according to its red, green and blue values. Cluster centroids are then calculated. In the second stage, a *fine* segmentation is obtained by assigning pixels in safe areas to their closest clusters as determined from the fuzzy membership functions. One advantage of this approach is that it dynamically determines the number of clusters. However, the number of clusters obtained is significantly affected by the smoothing function parameter and the size of the safe area [Turi 2001].

Color image segmentation using competitive learning based on the least-squares criterion was proposed by Uchiyama and Arbib [1994]. An image segmentation approach based on the mean shift algorithm was proposed by Comaniciu and Meer [1997]. Shi and Malik [1997] addressed image segmentation using clustering as a graph partitioning problem.

Zhang *et al.* [2001] proposed a hybrid approach combining hidden Markov random field (HMRF) and the EM algorithm to segment brain magnetic resonance (MR) images. A HMRF model is a stochastic process generated by a MRF. The HMRF state sequence can be observed through a field of observations [Zhang *et al.* 2001]. An advantage of HMRF is that it encodes spatial information, which is very useful in image segmentation since it reduces the sensitivity to the presence of noise. A parameter estimation method is required to approximate the parameters of the HMRF model. In this approach, the EM algorithm is used to estimate the parameters. One drawback of this approach is that it depends on the initial estimations of the parameters.

Recently, Veenman *et al.* [2003] proposed a cellular coevolutionary algorithm for image segmentation. The algorithm places agents in a two-dimensional grid representing an image. The agents move pixels between each other to improve the homogeneity of the regions. Neighboring agents form alliances if the union of their regions is homogenous. This approach does not require the user to specify the number of clusters in advance. However, it requires the user to specify a parameter (discussed in section 3.1.4, equation (3.22)) that has a profound effect on the performance of the algorithm.

3.3 Color Image Quantization

Color image quantization is the process of reducing the number of colors presented in a digital color image [Braquelaire and Brun 1997]. Color image quantization can be formally defined as follows [Velho *et al.* 1997]:

Given a set of $N_{S'}$ colors $\boldsymbol{S}' \subset \Re^{N_d}$. The color quantization is a map $f_q : \boldsymbol{S}' \rightarrow \boldsymbol{S}''$ where $\boldsymbol{S}''$ is a set of $N_{S''}$ colors such that $\boldsymbol{S}'' \subset \boldsymbol{S}'$ and $N_{S''} < N_{S'}$. The objective is to minimize the quantization error resulting from replacing a color $\boldsymbol{c} \in \boldsymbol{S}'$ with its quantized value $f_q(\boldsymbol{c}) \in \boldsymbol{S}''$.

Color image quantization is an important problem in the fields of image processing and computer graphics [Velho *et al*. 1997]:

- It can be used in lossy compression techniques [Velho *et al*. 1997];
- It is suitable for mobile and hand-held devices where memory is usually small [Rui *et al*. 2002];
- It is suitable for low-cost color display and printing devices where only small number of colors can be displayed or printed simultaneously [Scheunders, A Genetic 1997].
- Most graphics hardware use color lookup tables with a limited number of colors [Freisleben and Schrader 1997].

Color image quantization consists of two major steps:

- Creating a colormap (or palette) where a small set of colors (typically 8-256 [Scheunders, A Genetic 1997]) is chosen from the (2^{24}) possible combinations of red, green and blue (RGB).
- Mapping each color pixel in the color image to one of the colors in the colormap.

Therefore, the main objective of color image quantization is to map the set of colors in the original color image to a much smaller set of colors in the quantized image [Xiang and Joy 1994]. Furthermore, this mapping, as already mentioned, should minimize the difference between the original and the quantized images [Freisleben and Schrader 1997]. The color quantization problem is known to be NP-complete [Wu and Zhang 1991]. This means that it is not feasible to find the global optimal solution because this will require a prohibitive amount of time. To address this problem, several approximation techniques have been used. One popular approximation method is the use of a standard local search strategy such as K-means. K-means has already been applied to the color image quantization problem [Shafer and Kanade 1987; Celenk 1990]. However, as previously mentioned, K-means is a greedy algorithm which depends on the initial conditions, which may cause the algorithm to converge to suboptimal solutions. This drawback is magnified by the fact that the distribution of local optima is expected to be broad in the color image quantization problem due to the three dimensional color space. In addition, this local optimality is expected to affect the visual image quality. The local optimality issue can be addressed by using stochastic optimization schemes.

Several heuristic techniques have been proposed in the literature. These techniques can be categorized into two main categories: pre-clustering and post-clustering. The next subsections discuss each of these categories.

3.3.1 Pre-clustering approaches

Pre-clustering approaches divide the color into disjoint regions of similar colors. A representative color is then determined from each region. These representatives form the colormap. There are many fast algorithms in this category which are commonly used.

The median cut algorithm (MCA) [Heckbert 1982] is often used in image applications because of its simplicity [Freisleben and Schrader 1997]. MCA divides the color space repeatedly along the median into rectangular boxes until the desired number of colors is obtained.

The variance-based algorithm (VBA) [Wan 1990] also divides the color space into rectangular boxes. However, in VBA the box with the largest mean squared error between the colors in the box and their mean is split.

The octree quantization algorithm [Gervautz and Purgathofer 1990] repeatedly subdivides a cube into eight smaller cubes in a tree structure of degree eight. Then adjacent cubes with the least number of pixels are merged. This is repeated until the required number of colors is obtained [Dekker 1994]. Octree produces results similar to MCA, but with higher speed and smaller memory requirements [Freisleben and Schrader 1997].

Xiang and Joy [1994] proposed an agglomerative clustering method which starts with each image color as a separate cluster. Small clusters are then repeatedly clustered into larger clusters in a hierarchical way until the required number of colors is obtained. The abandoning of the fixed hierarchical division of the color space is a significant improvement over the octree approach [Xiang and Joy 1994].

A similar approach called *Color Image Quantization by Pairwise Clustering* was proposed by Velho *et al*. [1997]. In this approach, a relatively large set of colors is chosen. An image histogram is then created. Two clusters that minimize the quantization error are then selected and merged together. This process is repeated until the required number of colors is obtained. According to Velho *et al*. [1997], this approach performed better than MCA, VBA, octree, K-means and other popular quantization algorithms when applied to the two colored images used in their experiments.

Xiang [1997] proposed a color image quantization algorithm that minimizes the maximum distance between color pixels in each cluster (i.e. the intra-cluster distance). The algorithm starts by assigning all the pixels into one cluster. A pixel is then randomly chosen as the *head* of the cluster. A pixel that is the most distant from its cluster head is chosen as the head of a new cluster. Then, pixels nearer to the head of the new cluster move towards the new head forming the new cluster. This procedure is repeated until the desired number of clusters is obtained. The set of cluster heads forms the colormap.

A hybrid competitive learning (HCL) approach combining competitive learning and splitting of the color space was proposed by Scheunders [A Comparison 1997]. HCL starts by randomly choosing a pixel as a cluster centroid. Competitive learning is then applied resulting in assigning all the image pixels to one cluster surrounding the centroid. A splitting process is then conducted by creating another copy of the centroid; competitive learning is then applied on both centroids. This process is repeated until the desired number of clusters is obtained. According to Scheunders [A Comparison 1997], HCL is fast, completely independent of initial conditions and can obtain near global optimal results. When applied to commonly used images, HCL outperformed MCA, VBA and K-means, and performed comparably with competitive learning [Scheunders, A Comparison 1997; Scheunders, A Genetic 1997].

Braquelaire and Brun [1997] compared the various pre-clustering heuristics and suggested some optimizations of the algorithms and data structures used. Furthermore, they proposed a new color space called H_1 H_2 H_3 and argued that it improves the quantization heuristics. Finally, they proposed a new method which divides each cluster along the axis H_1, H_2 or H_3 of greatest variance. According to Braquelaire and Brun [1997], the proposed approach generates images with comparable quality to that obtained from better but slower methods in this category.

Recently, Cheng and Yang [2001] proposed a color image quantization algorithm based on color space dimensionality reduction. The algorithm repeatedly sub-divides the color histogram into smaller classes. The colors of each class are projected into a line. This line is defined by the mean color vector and the most distant color from the mean color. For each class, the vector generated from the projection of the colors into the line is then used to cluster the colors into two representative palette colors. This process is repeated until the desired number of representative colors is obtained. All color vectors in each class are then represented by their class mean. Finally, all these representative colors form the colormap. According to Cheng and Yang [2001], this algorithm performed better than MCA, and performed comparably to SOM when applied on commonly used images.

3.3.2 Post-clustering approaches

The main disadvantage of the pre-clustering approaches is that they only work with color spaces of simple geometric characteristics. On the other hand, post-clustering approaches can work with arbitrary shaped clusters. Post-clustering approaches perform clustering of the color space [Cheng and Yang 2001]. A post-clustering algorithm starts with an initial colormap. It then iteratively modifies the colormap to improve the approximation. The major disadvantage of post-clustering algorithms is the fact that it is time consuming [Freisleben and Schrader 1997].

The K-means algorithm is one of the most popular post-clustering algorithms. It starts with an initial set of colors (i.e. initial colormap). Then, each color pixel is assigned to the closest color in the colormap. The colors in the colormap are then recomputed as the centroids of the resulting clusters. This process is repeated until convergence. The K-means algorithm has been proven to converge to a local optimum

[Freisleben and Schrader 1997]. As previously mentioned, a major disadvantage of K-means is its dependency on initial conditions.

FCM [Balasubramanian and J. Allebach 1990] and Learning Vector Quantization [Kotropoulos *et al.* 1992] have also been used in the color image quantization. Scheunders and De Backer [1997] proposed a joint approach using both competitive learning and a dithering process to overcome the problem of contouring effects when using small colormaps.

Fiume and Quellette [1989] proposed an approach which uses simulated annealing for color image segmentation. Pre-clustering approaches were used to initialize the colormap.

SOMs (discussed in Section 3.1.6) were used by Dekker [1994] to quantize color images. The approach selects an initial colormap, and then modifies the colors in the colormap by moving them in the direction of the image color pixels. However, to reduce the execution time, only samples of the colors in the image are used. According to Dekker [1994], the algorithm performs better than MCA and octree.

Rui *et al*. [2002] presented an initialization and training method for SOM that reduces the computational load of SOM and at the same time generates reasonably good results.

A hybrid approach combining evolutionary algorithms with K-means has been proposed by Freisleben and Schrader [1997]. A population of individuals, each representing a colormap, are arbitrary initialized. Then, after each generation, the K-means algorithm (using a few iterations) is applied on each individual in the population. The standard error function of the Euclidean distance is chosen to be the fitness function of each individual. Based on the experiments conducted by Freisleben and Schrader [1997], this hybrid approach outperformed both MCA and octree algorithms.

Genetic C-means algorithm (GCMA) uses a similar idea where a hybrid approach combining a genetic algorithm with K-means was proposed by Scheunders [A Genetic 1997]. The fitness function of each individual in the population is set to be the mean square error (MSE), defined as

$$MSE = \frac{\sum_{k=1}^{K} \sum_{\forall z_p \in C_k} (\boldsymbol{z}_p - \boldsymbol{m}_k)^2}{N_p} \tag{3.27}$$

As in Freisleben and Schrader [1997], each chromosome represents a colormap. GCMA starts with a population of arbitrary initialized chromosomes. K-means is then applied to all the chromosomes to reduce the search space. A single-point crossover is then applied. This is followed by the application of mutation which randomly decides if a value of one is added to (or subtracted from) the gene's value (i.e. mutating the gene's value with ±1). All the chromosomes are then pairwise compared and the chromosome with the lowest MSE replaces the other chromosome. This process is repeated until a stopping criterion is satisfied. A faster version of this approach can be obtained by applying K-means to the best chromosome in each generation. For the remaining chromosomes, an approximation of K-means is used where a single iteration of K-means is applied on a randomly chosen subset of pixels. This process is repeated a user-specified number of times using different subsets. GCMA outperformed MCA, VBA, K-means, competitive learning and HCL when applied on commonly used images [Scheunders, A Comparison 1997; Scheunders, A Genetic 1997]. However, GCMA is computationally expensive.

Recently, a new approach using model based clustering trees was proposed by Murtagh *et al*. [2001]. The algorithm requires selecting a 3D color space (e.g. RGB) and specifying the order of color bands. For the first color band, the number of clusters is determined using BIC (discussed in section 3.1.5, equation (3.23)). The EM algorithm is used to estimate the model parameters and each pixel is then assigned to its most likely cluster. The second color band is then used to split each of the clusters generated from the previous step. The generated clusters are further subdivided using the third color band.

3.4 Spectral Unmixing

In remote sensing, classification is the main tool for extracting information about the surface cover type. Conventional classification methods assign each pixel to one class (or species). This class can represent water, vegetation, soil, etc. The classification methods generate a map showing the species with highest concentration. This map is known as the *thematic map*. A thematic map is useful when the pixels in the image represent pure species (i.e. each pixel represents the spectral signature of one species). Hence, thematic maps are suitable for imagery data with a small ground sampling distance (GSD) such as LANDSAT Thematic Mapper (GSD = 30 m). However, thematic maps are not as useful for large GSD imagery such as NOAA'a AVHRR (GSD = 1.1 km) because in this type of imagery pixels are usually not pure. Therefore, pixels need to be assigned to several classes along with their respective concentrations in that pixel's footprint. Spectral unmixing (or *mixture modeling*) is used to assign these classes and concentrations. Spectral unmixing generates a set of maps showing the proportions of all species present in each pixel footprint. These maps are called the *abundance images*. Hence, each abundance image shows the concentration of one species in a scene. Therefore, spectral unmixing provides a more complete and accurate classification than a thematic map generated by conventional classification methods.

Spectral unmixing can be used for the compression of multispectral imagery. Using spectral unmixing, the user can prioritize the species of interest in the compression process. This is done by first applying the spectral unmixing on the original images to generate the abundance images. The abundance images representing the species of interest are then prioritized by coding them with a relatively high bit rate. Other abundance images are coded using a relatively low bit rate. At the decoder, the species-prioritized reconstructed multispectral imagery is generated via a re-mixing process on the decoded abundance images [Saghri *et al.* 2002]. This approach is feasible if the spectral unmixing algorithm results in a small (negligible) residual error.

3.4.1 Linear Pixel Unmixing (or Linear Mixture Modeling)

Spectral unmixing is generally performed using a linear mixture modeling approach. In linear mixture modeling the spectral signature of each pixel vector is assumed to be a linear combination of a limited set of fundamental spectral components known as *end-members*. Hence, spectral unmixing can be formally defined as follows:

$$\boldsymbol{z}_p = \boldsymbol{EM}.\boldsymbol{f} + \boldsymbol{e} = f_1\boldsymbol{em}_1 + f_2\boldsymbol{em}_2 + \cdots + f_i\boldsymbol{em}_i + \cdots + f_{N_e}\boldsymbol{em}_{N_e} + \boldsymbol{e} \qquad (3.28)$$

where the symbols are defined as follows:

$\boldsymbol{z}_p$	a pixel signature of N_b components
$\boldsymbol{EM}$	$N_b \times N_e$ matrix of end-members $\boldsymbol{em}_{1,\cdots,N_e}$
f_i	fractional component of end-member i (i.e. proportion of footprint covered by species i)
$\boldsymbol{f}$	vector of fractional components $(f_1, f_2, \cdots, f_i, \cdots, f_{N_e})^{\mathrm{T}}$
$\boldsymbol{em}_i$	end-member i of N_b components
$\boldsymbol{e}$	residual error vector of N_b components
N_b	number of spectral bands
N_e	number of components, $N_e \le N_b$

Provided that the number of end-members is less than or equal to the true spectral dimensionality of the scene, the solution via classical least-squares estimation is,

$$\boldsymbol{f} = (\boldsymbol{EM}^{\mathrm{T}}\boldsymbol{EM})^{-1}\boldsymbol{EM}^{\mathrm{T}}\boldsymbol{z}_p \tag{3.29}$$

Therefore, there are two requirements for linear spectral unmixing:

- the spectral signature of the end-members needs to be known, and
- the number of end-members has to be less than or equal to the true spectral dimensionality of the scene (i.e. the dimension of the feature space). This is known as the *condition of identifiability*.

The condition of identifiability restricts the application of the linear spectral unmixing when applied to multispectral imagery, because

- the end-members may not correspond to physically identifiable species on the ground, and
- the number of distinct species in the scene may be more than the true spectral dimensionality of the scene. For example, for Landsat TM with seven spectral bands (N_b =7), the true spectral dimension is at most five (N_e=5) based on principal component analysis.

3.4.2 Selection of the End-Members

There are many methods for end-member selection proposed in the literature [Settle and Drake 1993; Antoniades *et al.* 1995; Hlavka and Spanner 1995; Bateson and Curtiss 1996; Maselli 1998; Parra *et al.* 2000; Saghri *et al.* 2000]. In the following, several representative techniques are presented.

Mathematical techniques such as Gram-Schmidt orthogonalization and principal component analysis can be used to obtain orthogonal end-members which can be used to linearly unmix each pixel vector of the scene. There are several advantages for the mathematical techniques, namely

- they result in minimum residual error, and
- there is no human interaction time.

However, mathematical techniques suffer from the following drawbacks:

- They may generate end-members with negative components.
- They may not correspond to physical species in the scene.

Manual techniques can also be used to obtain end-members which can be used to linearly unmix each pixel vector of the scene. In manual techniques, the user will select the end-members directly from the scene, or from a library of end-members. The advantages of manual techniques are the disadvantages of mathematical techniques and *vice versa*.

Spectral screening is another way to obtain end-members. In this approach, a set of unique pixels are selected from the scene. The selection is based on a user-specified spectral angle threshold. The approach works as follows:

- The first pixel in the image is assumed to be unique and is added to the set of unique pixels.
- The pixels in the image are then sequentially scanned and each pixel whose spectral angle with respect to all the unique pixels in the set exceeds the user-specified spectral angle threshold, is added to the set of unique pixels.

Clearly, this technique suffers from two major drawbacks:

- the generated set of unique pixels depends on the order in which the pixels are scanned, and
- the generated set also depends on the spectral angle threshold.

To overcome the condition of identifiability, Maselli [1998] proposed a method of dynamic selection of an optimum end-member subset. In this technique, an optimum subset of all available end-members is selected for spectral unmixing of each pixel vector in the scene. Thus, although every pixel vector will not have a fractional component for each end-member, the ensemble of all pixel vectors in the scene will collectively have fractional contributions for each end-member.

For each pixel vector, a unique subset of the available end-members is selected which minimizes the residual error after decomposition of that pixel vector. To determine the N_e optimum end-members for pixel vector $\boldsymbol{z}_p$, the pixel vector is projected onto all available normalized end-members. The most efficient projection, which corresponds to the highest dot product value $c_{\max}$, indicates the first selected end-member $\boldsymbol{em}_{\max}$. It can be shown that this procedure is equivalent to finding the end-member with the smallest spectral angle with respect to $\boldsymbol{z}_p$ [Saghri *et al.* 2000]. The residual pixel signature, $\boldsymbol{r}_{\boldsymbol{z}_p} = \boldsymbol{z}_p - c_{\max}.\boldsymbol{em}_{\max}$ is then used to identify the second end-member by repeating the projection onto all remaining end-members. The process continues up to the identification of a prefixed maximum N_e number of end-members from the total of N_m available end-members.

More recently, Saghri *et al.* [2000] proposed a method to obtain end-members from the scene with relatively small residual errors. In this method, the set of end-members are chosen from a thematic map resulting from a modified ISODATA. The modified ISODATA uses the spectral angle measure instead of the Euclidean distance measure to reduce the effect of shadows and sun angle effects. The end-members are then set as the centroids of the compact and well-populated clusters. Maselli's approach discussed above is then used to find the optimum end-member subset from the set of available end-members for each pixel in the scene. Linear spectral unmixing is then applied to generate the abundance images.

According to Saghri *et al.* [2000], the proposed approach has several advantages:

- the resulting end-members correspond to physically identifiable (and likely pure) species on the ground,

- the residual error is relatively small, and
- minimal human interaction time is required.

However, this approach has the drawback that it uses ISODATA which depends on initial conditions.

3.5 Conclusions

This chapter presented an overview of a set of problems from the field of pattern recognition and image processing. The clustering problem was defined and discussed, followed by image segmentation and color image quantization. Finally, spectral unmixing was overviewed. From the discussion presented in this chapter it can be observed that all these problems are difficult to solve and they need efficient optimization methods to solve them. In this thesis, the PSO is used to address these difficult problems. In the next chapter, a PSO-based clustering algorithm is proposed and compared with other *state-of-the-art* clustering algorithms.