A clustering method based on multidimensional texture analysis

K. Hammouche\textsuperscript{a}, M. Diafa\textsuperscript{a}, J.-G. Postaire\textsuperscript{b, *}

\textsuperscript{a} Université Mouloud Mammeri, Département d’Automatique, Tizi Ouzou, Algeria
\textsuperscript{b} Université des Sciences et Technologies de Lille—LAGIS, UMR CNRS 8146, 59655 Villeneuve d’Ascq Cedex, France

Received 24 May 2004; received in revised form 12 September 2005; accepted 22 November 2005

Abstract

Considering the analogy between image segmentation and cluster analysis, the aim of this paper is to adapt statistical texture measures to describe the spatial distribution of multidimensional observations. The main idea is to consider the cluster cores as domains characterized by their specific textures in the data space. The distribution of the data points is first described as a multidimensional histogram defined on a multidimensional regular array of sampling points. In order to evaluate locally a multidimensional texture, a co-occurrence matrix is introduced, which characterizes the local distribution of the data points in the multidimensional data space. Several local texture features can be computed from this co-occurrence matrix, which accumulates spatial and statistical information on the data distribution in the neighborhoods of the sampling points. Texture features are selected according to their ability to discriminate different distributions of data points. The sampling points where the local underlying texture is evaluated are categorized into different texture classes. The points assigned to these classes tend to form connected components in the data space, which are considered as the cores of the clusters.

Keywords: Cluster analysis; Texture; Co-occurrence matrices; Feature selection

1. Introduction

The aim of cluster analysis is to divide a set of multidimensional observations into subsets according to their similarities and dissimilarities. These observations are generally represented as data points scattered through an \( N \)-dimensional data space, each point corresponding to a vector of observed features measured on the objects to be classified.

In the framework of the statistical approach, many clustering procedures have been proposed, based on the analysis of the underlying probability density function (pdf) \cite{1}.

Independently from cluster analysis, a large amount of research effort has been devoted to image segmentation. To humans, an image is not just an unstructured collection of pixels. We generally agree about the different regions constituting an image due to our visual grouping capabilities.

Among the factors that lead to such perceptual grouping, the most important are similarity, proximity and connectedness. The segmentation process can be considered as a partitioning scheme such that

- every pixel of the image must belong to a region,
- the regions must be composed of contiguous pixels,
- the pixels constituting a region must share a given property of similarity.

These three conditions can be easily adapted to the clustering process. Indeed, each data point must be assigned to a cluster, and the clusters must be composed of neighboring data points since the points assigned to the same cluster must share some properties of similarity.

Considering this analogy between segmentation and clustering, some image segmentation procedures based on the gray-level function analysis have been successfully adapted to multidimensional density function analysis for pattern classification purpose.

In this framework, the underlying pdf is generally estimated on a regular discrete array of sampling points \cite{2}.
The idea of using a pdf estimation for mode seeking is not new [3] and in very simple situations, the modes can be detected by thresholding the pdf at an appropriate level, using a procedure similar to image binarization [4]. A “mode” label is associated with each point where the underlying pdf is above the threshold. Otherwise, the corresponding point is assigned a “valley” label.

However, in practical situations, it is often difficult, or even impossible, to select an appropriate threshold to detect the significant modes. A solution for improving this simple thresholding scheme is to consider the spatial relationships among the sampling points where the underlying pdf is estimated, rather than making a decision at each point independently of the decisions at other points. Probabilistic labeling, or relaxation, is a formalism through which object labels are iteratively updated according to a compatibility measure defined among the neighboring labels. This approach, which has been mainly applied to image processing [5], has been adapted to cluster analysis to reduce local ambiguities in the mode/valley discrimination process [6].

The segmentation of an image can also be considered as a problem of edge detection [7]. Similarly, in the clustering context, a mode boundary can be defined as an area of abrupt local changes in the pdf. It can be detected by means of generalized gradient operators designed to perform a discrete spatial differentiation of the estimated pdf [8]. Although these spatial operators enhance substantially the discontinuities that delineate the modes, a relaxation labeling process, similar to the one used for thresholding, can be necessary for mode boundary extraction [9].

Beside procedures based on the concepts of similarity and discontinuity, mathematical morphology has proven to be a valuable approach for image segmentation. This theory has been adapted to cluster analysis by considering the sets of multidimensional observations as mathematical discrete binary sets [10]. Binary erosions and dilations of these discrete sets eliminate irrelevant details in the shapes of the clusters without geometric distortions [11]. More recently, multivalued morphological operations, such as numerical erosions, dilations and homotopic thinnings have been defined for processing multidimensional pdf using the umbra concept [12]. With these operators, the clusters are delineated by means of the watershed transform [13].

Modeling spatial relationships between pixels by means of Markov random fields has proved to be relevant to the image segmentation problem [14,15]. The Markovian approach has recently been adapted to the mode detection problem in cluster analysis. The hidden field containing the "mode" and the "valley" labels is derived from the observable field representing the data set by means of the estimation-maximization algorithm combined with the maximum a posteriori mode criterion [16,17].

All the above-mentioned clustering methods tend to generalize bi-dimensional procedures initially developed for image processing purpose. In this paper, following the same idea of adapting image processing techniques to cluster analysis, our objective is to consider statistical texture measures to describe the spatial distribution of the data points. Similar to texture segmentation, the approach consists first in selecting a set of features that characterize the local distribution of the data points in the multidimensional data space in terms of textures. These textures, which reflect the spatial arrangement of data points, are then classified on the basis of these features. The data points with similar local textures are aggregated in the data space to define compact connected components of homogeneous textures. Some of these multidimensional domains of uniform texture are finally considered as the cores of the clusters.

This paper is organized as follows. In the next section, we introduce a new local characterization of the data distribution by means of texture descriptors. Section 3 describes the proposed clustering approach based on texture analysis. The selection of the particular texture features suitable for cluster analysis is discussed in Section 4. In the last section, the performance of the clustering algorithm is demonstrated using artificially generated data sets.

2. Multidimensional texture characterization

To illustrate the basic ideas behind the proposed approach, let us consider the eight bi- and three-dimensional uniform distributions of data points of Fig. 1. Due to the ability of humans to identify different visual textures, these distributions appear to be uniform in some sense, and yet different one from each other. These considerations have led the authors to consider the texture as a property of the data point distribution. In this paper, it is assumed that the texture tends to be uniform within the core associated with each cluster, so that these cores can be searched as domains of the data space characterized by a relative homogeneity of suitable texture descriptors.

2.1. Discretization of the data set

In order to adapt texture analysis tools to clustering, it is necessary to introduce a discrete array of sampling points [2]. Let us consider \( Q \) observations

\[
X_q = [x_{q,1}, x_{q,2}, \ldots, x_{q,n}, \ldots, x_{q,N}], \quad q = 1, 2, \ldots, Q,
\]

where \( x_{q,1}, x_{q,2}, \ldots, x_{q,n}, \ldots, x_{q,N} \) are the \( N \) coordinates of the observation \( X_q \) in the data space. The range of variation of each component of the multivariate observations is normalized to the interval \([0,S]\), where \( S \) is an integer, by means of the transformation

\[
x'_{q,n} = S \frac{x_{q,n} - \min_{q=1}^{Q} x_{q,n}}{\max_{q=1}^{Q} x_{q,n} - \min_{q=1}^{Q} x_{q,n}}.
\]

Let \( X'_q = [x'_{q,1}, x'_{q,2}, \ldots, x'_{q,n}, \ldots, x'_{q,N}]^T, \quad q = 1, 2, \ldots, Q, \) be the \( Q \) new observations in the normalized data space. Each axis of this space is partitioned into \( S \) exclusive and adjacent intervals of unit width. This discretization defines
Fig. 1. Random textures synthesized as uniform bi- and three-dimensional distributions of \( Q \) data points: (a) \( Q = 200 \); (b) \( Q = 600 \); (c) \( Q = 1000 \); and (d) \( Q = 1600 \).

Fig. 2. Discretization of a data set: (a) raw data set; and (b) discrete multidimensional histogram with \( S = 25 \).

The result of this sampling procedure is a multidimensional regular array of discrete integers in the range \([0, L]\), where \( L \) is the maximum value of \( \hat{p}(P_r), r = 1, 2, \ldots, S^N \), that is well conditioned for multidimensional texture analysis.

The next step now is to define meaningful features to describe the textural information that will be used to identify the cluster cores in the data space.

2.2. Texture parameters

When considering the examples of Fig. 1, it is clear that structural models based on primitive placement rules cannot satisfactorily describe the texture of the distribution of the data points. Therefore, one is led to consider the textural properties in terms of statistical models and the main difficulty is the selection of a set of relevant features to describe the properties of the spatial distribution of the data. A number of textural parameters have been proposed in the image processing literature, derived from autoregressive models [18], Markov random fields models [19], Gabor filters [20], and wavelet analysis [21].
Table 1
Statistical texture features ($N_c = \sum_{i=0}^{L} \sum_{j=0}^{L} T(i, j)$ is a normalizing parameter)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uniformity-1 (first order)</td>
<td>$f_1 = \frac{1}{N_c} \sum_{i=0}^{L} \sum_{j=0}^{L} T(i, j)$</td>
</tr>
<tr>
<td>Uniformity-2 (second order)</td>
<td>$f_2 = \frac{1}{N_c} \sum_{i=0}^{L} T(i, j)^2$</td>
</tr>
<tr>
<td>Homogeneity</td>
<td>$f_3 = \frac{1}{N_c} \sum_{i=0}^{L} \sum_{j=0}^{L} T(i, j)$ / $N_c$</td>
</tr>
<tr>
<td>Correlation</td>
<td>$f_4 = \frac{1}{N_c} \sum_{i=0}^{L} \sum_{j=0}^{L} ijT(i, j)$</td>
</tr>
<tr>
<td>Energy</td>
<td>$f_5 = \frac{1}{N_c} \sum_{i=0}^{L} \sum_{j=0}^{L} T(i, j)^2$</td>
</tr>
<tr>
<td>Entropy</td>
<td>$f_6 = -\frac{1}{N_c} \sum_{i=0}^{L} \sum_{j=0}^{L} T(i, j) \log(T(i, j)/N_c)$</td>
</tr>
<tr>
<td>Inertia</td>
<td>$f_7 = \frac{1}{N_c} \sum_{i=0}^{L} \sum_{j=0}^{L} (i-j)^2 T(i, j)$</td>
</tr>
<tr>
<td>Means</td>
<td>$f_8 = \frac{1}{N_c} \sum_{i=0}^{L} \sum_{j=0}^{L} T(i, j)$ / $N_c$</td>
</tr>
<tr>
<td>Covariance</td>
<td>$f_9 = \frac{1}{N_c} \sum_{i=0}^{L} \sum_{j=0}^{L} T(i, j)$ / $N_c$</td>
</tr>
<tr>
<td>Cluster shade</td>
<td>$f_{10} = \frac{1}{N_c} \sum_{i=0}^{L} \sum_{j=0}^{L} (i-f_k)(j-f_k) T(i, j)$</td>
</tr>
<tr>
<td>Cluster prominence</td>
<td>$f_{11} = \frac{1}{N_c} \sum_{i=0}^{L} \sum_{j=0}^{L} (i-f_k)^2 + (j-f_k)^2 T(i, j)$</td>
</tr>
<tr>
<td>Absolute value</td>
<td>$f_{12} = \frac{1}{N_c} \sum_{i=0}^{L} \sum_{j=0}^{L}</td>
</tr>
</tbody>
</table>

In the framework of image processing, an element $T(i, j)$ of a co-occurrence matrix is a count of the number of times a pixel $P_r = [x_r, y_r]$ with gray-level $i$ is positioned with respect to a pixel $P_r = [x_r', y_r']$ with gray level $j$ such as

$$P_r' = P_r + \left[ \frac{d \cos \theta}{d \sin \theta} \right],$$

where $d$ is the distance in the direction $\theta$ between the two pixels.

A similar co-occurrence matrix is determined to characterize the local distribution of the data points in a given neighborhood of each sampling point where the histogram value is not null. We use a classical hypercubic neighborhood of side length $(2h + 1)$, where $h$ is an integer. As directionality and periodicity are obviously irrelevant characteristics of the data point distributions, it is not necessary to determine co-occurrence matrices for different sets of discrete values of the distance $d$ and the orientation $\theta$ between the pairs of sampling points taken into account. Hence, only one co-occurrence matrix is determined for each sampling point. The co-occurrences $T(i, j)$ of any given pair $(i, j)$ of histogram values are simply counted for all the couples of adjacent sampling points encountered within this hypercubic neighborhood. Two sampling points are considered as adjacent if they are the centers of two hypercubes that have at least one point in common. As the histogram is quantized on a set of $L + 1$ discrete values, the co-occurrence matrices have $L + 1$ rows and $L + 1$ columns.

Analogously as in Refs. [25,26], several local texture features can be computed from these specific co-occurrence matrices, which accumulate information on the data distribution in the neighborhood of each sampling point (cf. Table 1). These features are expected to characterize such properties as roughness, smoothness, homogeneity, randomness or coarseness rather than textural properties such as directionality or periodicity, since each co-occurrence matrix summarizes the number of occurrences of pairs of histogram values for all possible pairs of adjacent sampling points lying within a $(2h + 1)^N$ given neighborhood, without constraints on their orientations.

3. Cluster core extraction

3.1. Texture classification

Similar to image segmentation, it is expected that sampling points with similar texture properties could be aggre-
Fig. 3. Domains of homogeneous textures associated with the data set of Fig. 2 when different numbers of texture classes are required by the K-means algorithm: (a) two classes of textures; (b) three classes of textures; and (c) four classes of textures: x: cluster cores, +: valleys, o: core boundaries, *: core surrounding.

The aim of this paper is not to compare the performance of unsupervised classifiers, but rather to demonstrate the efficiency of the description of the data structure by means of a set of texture features. To assess the relevance of the proposed texture based clustering procedure, we use the basic K-means algorithm where the desired number of classes of feature vectors has to be specified [27]. The ability of varying the number of expected classes makes it possible to give some insight into the significance of the clusters that can be identified within the data.

Fig. 3 shows the domains of homogeneous textures associated with the discrete data set of Fig. 2(b) when the K-means algorithm requires two to four classes of different textures. The texture discrimination is performed in a two-dimensional feature space defined by two features, namely the homogeneity $f_3$ and the correlation $f_4$ of Table 1, with $S = 25$. When two classes are required, the two domains correspond to the cluster cores and the valleys, respectively (cf. Fig. 3(a)). When the sampling points are assigned to three classes, one of them corresponds to the cores; the second to their boundaries and the last one to the valleys (cf. Fig. 3(b)). In the case of four classes, Fig. 3(c) shows that the cores are surrounded by concentric domains corresponding to different distribution characteristics that are obviously linked to the local data point densities.

We have kept the parameter $S$ and the two texture features unchanged in order to show the influence of the required number of texture classes on the resulting domains of homogeneous textures. A procedure to optimize the value of $S$, to select an appropriate set of texture features and to determine the texture class number will be presented in Section 4.
Table 2
Compactness of the different domains of Fig. 3 when two to four texture classes are required by the K-means algorithm

<table>
<thead>
<tr>
<th>Domain number</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 Classes of textures</td>
<td>0.0026</td>
<td>0.0333</td>
<td>0.0270</td>
<td>0.0285</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\phi = 0.0179$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 Classes of textures</td>
<td>0.0118</td>
<td>0.0083</td>
<td>0.0347</td>
<td>0.0347</td>
<td>0.0099</td>
<td>0.0340</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\phi = 0.0215$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 Classes of textures</td>
<td>0.0052</td>
<td>0.0110</td>
<td>0.0099</td>
<td>0.0330</td>
<td>0.0277</td>
<td>0.0092</td>
<td>0.0082</td>
<td>0.0295</td>
</tr>
<tr>
<td>$\phi = 0.0191$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig. 4. Compact cores resulting from the compactness thresholding of the domains of Fig. 3: (a) two classes of textures; (b) three classes of textures; and (c) four classes of textures.

3.2. Core extraction

Under the assumption that the cluster cores are multi-dimensional domains with homogeneous textures, it is expected that the hypercubes centered on the sampling points assigned to the same class of texture give rise to connected components in the data space. These components can be extracted by means of an aggregation procedure where two hypercubes whose centers belong to the same class of texture are assigned to the same component if they have at least one point in common. Small components resulting from this aggregation procedure may correspond to non-significant domains with only a few data points. Therefore, in all the examples illustrating this paper, any domain containing less than 5% of the total number $Q$ of observations is discarded.

Among the remaining components, those corresponding to the cores of the clusters are expected to be more compact than those corresponding to their boundaries or to the valleys between them. Hence, they can be discriminated from other components by analyzing their compactness defined as

$$C = \frac{\text{[total number of hypercubes]}}{\text{[number of boundary hypercubes]}^\gamma},$$

which is as much high as the component is compact. Table 2 indicates the compactness of the domains resulting from the aggregation of the connected sampling points of Fig. 3.
It appears that the cluster cores are much more compact than the other domains. In all the examples illustrating this paper, the compactness of the cluster cores is always greater than the mean $\phi$ of the lowest and the highest compactness measures of all the reconstructed domains. In these conditions, cluster core detection is straightforward by simple thresholding of the compactness. Fig. 4 shows the cores identified among the domains of homogeneous texture of Fig. 3.

Finally, many grouping procedures can be used to assign the input data points to the clusters attached to the detected cores. One simple solution is to use the data points falling into the cores as prototypes. The remaining data points are finally assigned to the clusters attached to their nearest neighbor among these prototypes [28].

4. Algorithm tuning and feature selection

The performance of the above described algorithm depends mainly on the adjustment of the discretization parameter $S$ and on the relevance of the chosen texture features.

4.1. Discretization tuning

Let us first consider the effect of the resolution of the discretization process. In fact, the adjustment of $S$ depends on the sample size $Q$ on the dimensionality $N$ of the data and on the structure of the distribution of the observations. It can be expected that, when true clusters exist, stable connected subsets of data points with similar texture properties appear for a wide range of values of $S$. Based on this assumption, the adjustment of $S$ can be governed by the concept of cluster stability [29]. Choosing such a parameter in the middle of the largest range where the number of detected clusters remains constant, and different from one, has proved to be a good procedure to optimize a number of clustering algorithms when nothing is a priori known about the structure of the distribution of the observations [30]. Note that the larger the range, the more reliable the tuning procedure.

4.2. Feature selection

In the framework of multidimensional texture analysis, the key problem is the selection of a set of suitable texture features. For choosing relevant features while reducing the dimensionality of the texture classification problem, we propose a performance-dependent feature selection scheme which is directly related to the above mentioned concept of cluster stability. The effectiveness of a subset of features is evaluated by means of the width of the largest range of values of the discretization parameter $S$ leading to the appropriate number of detected cluster cores. As mentioned at the end of Section 4.1, the larger this range, the more reliable the number of detected cores. This criterion is used to select a set of relevant features among the available ones by means of a sequential forward selection technique [31].

To evaluate the relative relevance of $M$ features $f_1, \ldots, f_m, \ldots, f_M$, we consider the feature subspaces $R^1, \ldots, R^m, \ldots, R^M$, taking into consideration an increasing number of texture features, from one to $M$. The algorithm starts with the $M$ possible $R^1$ spaces. The feature which maximizes the range of values of $S$ corresponding to a stable number of detected cores, different from one, is the first selected feature. This feature is combined, in a $R^2$ feature space, with each of the $M - 1$ remaining ones. The corresponding $M - 1$ lengths of the stable ranges for $S$ are then determined and the pair of features that maximizes the length is kept.

When $m$ features out of $M$ have been chosen, the algorithm proceeds in the $R^m+1$ feature space of $m + 1$ dimensions to select the $(m + 1)$th feature that maximizes the length of the range of $S$ when combined with the $m$ previously chosen features. This procedure is iterated until the $M$ features have been ordered by diminishing relevance. The sequence $L(m)$ of length values thus obtained allows to select a subset of relevant features within the full set of $M$ features. These salient features are those that correspond to the starting increasing phase of the length values in the sequence $L(m)$. All the features that follow the first decrease in the sequence $L(m)$ are discarded.

To demonstrate the efficiency of the proposed feature selection technique, we use the bi-dimensional data set of Fig. 2 constituted of three Gaussian clusters of equal weights. The length $L(m)$ of the longest range of values of $S$ where the same number of cluster cores is detected by the clustering procedure is plotted against the number $m$ of selected features (cf. Fig. 5). The feature selected at each step is indicated at the corresponding point of the plot. The series $(f_1, f_5, f_1, f_5, f_6, f_2, f_7), (f_5, f_1, f_2, f_4, f_7, f_3, f_6)$ and $(f_3, f_2, f_6, f_3, f_3, f_1, f_7)$ represent the seven first selected features ordered by decreasing relevance when two, three and four classes of textures are required by the K-means algorithm, respectively. As expected, the number of required classes influences the feature selection. When two classes are required, the selected features are $f_4, f_5$ and $f_1$ since $L(m)$ begins to decrease when $f_3$ is selected (cf. Fig. 5(a)). With three classes of textures, the plot of Fig. 5(b) shows that the two first features $f_5$ and $f_1$ are selected for detecting the three clusters. When four classes of textures are used, it appears that only the first feature $f_5$ is selected for detecting the three clusters (cf. Fig. 5(c)).

4.3. Number of texture classes

The next parameter that remains to be adjusted is the number of texture classes required by the K-means algorithm. This number is not determined automatically by the basic, but well controlled, version of the algorithm used in this work. Fortunately, the concept of cluster stability allows specifying this number by selecting the number of texture classes that leads to the longest range of variation of $S$ where
Fig. 5. Plots of the lengths \( L(m) \) of the largest ranges of \( S \) where the number of detected cores remains constant: (a) two classes of textures; (b) three classes of textures; and (c) four classes of textures.

Table 3

<table>
<thead>
<tr>
<th>Relevant Features</th>
<th>Largest range of ( S )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_4, f_5 ) and ( f_1 )</td>
<td>30</td>
</tr>
<tr>
<td>( f_6, f_3, f_1, f_2 ) and ( f_4 )</td>
<td>27</td>
</tr>
<tr>
<td>( f_6 )</td>
<td>22</td>
</tr>
<tr>
<td>( f_6, f_3 ) and ( f_2 )</td>
<td>18</td>
</tr>
</tbody>
</table>

4.4. Hypercubic neighborhood size

The neighborhoods used to determine the local values of the texture features have been defined as hypercubes constituted of \( (2h + 1)^N \) unit cells centered at the sampling points (cf. Section 2.2). We have analyzed the effect of the parameter \( h \) on the behavior of the algorithm. For each neighborhood size varying from \( h = 1 \) to \( h = 4 \), we have selected the relevant texture features as explained in Section 4.2 to classify the bi-dimensional data of Fig. 2(a), asking always for two texture classes. Table 3 indicates the largest ranges of the discretization parameter \( S \) where the numbers of detected clusters remain constant for each neighborhood size. It appears that the largest of these ranges is obtained for \( h = 1 \). Furthermore, beside being the best choice in terms of reliability of the results, the choice of the minimal neighborhood size (\( h = 1 \)) reduces the computation time while improving the sensitivity of the procedure to local texture properties.

5. Experimental results

The following examples have been chosen to provide some insight into the behavior of the proposed texture based clustering procedure and to demonstrate the interest of this approach for pattern classification.

5.1. Example 1

The data set presented in Fig. 6(a) is the basis for the first example, which is composed of 800 bi-dimensional observations drawn from the 2 overlapping normal distributions of unequal weights specified in Table 4.

In order to tune the algorithm, the number of required texture classes is varied from two to four. The largest range where the number of detected clusters remains constant appears for three classes. It corresponds to a partition of the data set into two clusters (cf. Fig. 6(b)). Fig. 6(c) shows the discrete data set obtained for \( S = 30 \), which is the middle of this range. The two cores detected as domains of homogeneous, but different, textures are displayed in Fig. 6(d). \( f_4 \) is the unique texture feature selected by the procedure described in Section 4 that leads to ask for three texture classes when using the K-means algorithm.

The result of the classification is shown in Fig. 6(e). Table 4 summarizes the statistics of the two detected clusters. The performance of the classifier is measured by the classification error-rate, estimated as the ratio of the number of misclassified observations to the total number of observations. The error-rate obtained with the proposed algorithm is equal to 2.125%. In this example, the classes do not overlap too much and the actual error-rate is very close to the theoretical minimum error-rate achieved by use of the Bayes decision rule associated with the true statistics of the data set, which is equal to 2%. The difference between these two error-rates corresponds to only one observation misclassified out of over 800.

5.2. Example 2

The major difficulties in cluster analysis are with non-spherical clusters, bridges between clusters and non-linearly
Fig. 6. Cluster analysis for the bi-dimensional data set of Example 1: (a) raw data set (for statistical parameters, see Table 4); (b) effect of the parameter \( S \) on the number of detected cores; (c) discrete multidimensional histogram; (d) cluster cores obtained for \( S = 30 \); and (e) result of clustering.

Table 4
Statistical parameters of the two bivariate distributions from which are drawn the observations of Example 1 and statistical parameters of the detected clusters

<table>
<thead>
<tr>
<th>Generated data</th>
<th>Results of clustering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of data points</td>
<td>Mean vector</td>
</tr>
<tr>
<td>Population 1</td>
<td>500</td>
</tr>
<tr>
<td>Population 2</td>
<td>300</td>
</tr>
</tbody>
</table>

1 separable clusters. The bivariate data set of Fig. 7(a) has been generated keeping these well-known difficulties in mind. It is composed of two populations of 1000 data points each drawn as

\[
x_1 = A_1 \cos \Theta + B_1,
\]

\[
x_2 = A_2 \cos \Theta + B_2,
\]

where \( \Theta \) is a normal random variable with mean \( m \) and standard deviation \( s \), and where \( B_1 \) and \( B_2 \) are normal random variables with means \( \mu_1 \) and variances \( \sigma \) (cf. Table 5).

For this example, the largest range of \( S \) where the two clusters have been identified is \([20–47]\) (cf. Fig. 7(b)). Fig. 7(c) shows the discrete data set obtained for \( S = 34 \), i.e. the middle of that range. Four texture features, namely \( f_5 \), \( f_2 \), \( f_6 \), etc.
and \( f_3 \), have been selected to obtain the two cores shown in Fig. 7(d). The set of four-dimensional feature vectors have been partitioned into two classes, giving rise to two cores that are used to classify the input data. The result is shown in Fig. 7(e). The error-rate associated with the texture clustering procedure is 4.7%, whereas it reaches 12.5% with the ISODATA algorithm [32]. This example shows that when central points cannot represent the clusters globally, the texture based approach, which takes into account the local properties of the distribution of the input data, performs much better than the ISODATA algorithm.

### 5.3. Example 3

We now present a multidimensional case, which demonstrates the ability of the procedure to identify interlaced clusters for data of higher dimensionality. The data shown in Fig. 8(a) consist of two clusters generated as circular torus formed by the rotation of a plane circular Gaussian distribution about an axis in the plane of that distribution. These two torus are interlaced as the rings of a chain. The result achieved by the clustering procedure with \( S = 34 \), which is the middle of the [17–50] largest range where the number
of detected clusters remains constant, is presented in Fig. 8(b). The texture classification process relies on the features \( f_3 \) and \( f_4 \) with two texture classes. Although the discrimination between the two classes of textures is based on the basic K-means algorithm, and although the assignment of the data points standing out of the two detected cores is performed by means of the simple nearest neighbor rule, the error-rate associated with the texture clustering procedure is 0.1%, whereas it reaches 12.17% with the ISODATA algorithm. This result demonstrates the effectiveness of the approach in a non-trivial situation.

### 5.4. Example 4

The last example is presented in order to help quantify the computational effort required by the proposed algorithm. We use data sets constituted of two well-separated Gaussian distributions of observations with means \( M_1 = [2, 2, 2, \ldots, 2]^T \) and \( M_2 = [-2, -2, -2, \ldots, -2]^T \) and with unit covariance matrices: \( \Sigma_1 = \Sigma_2 = I_N \). For \( N \)-dimensional data, \( M_1 \) and \( M_2 \) are \( N \)-dimensional vectors, while \( \Sigma_1 \) and \( \Sigma_2 \) are \( N \times N \) unit covariance matrices.

The running times of the proposed algorithm have been evaluated for data of various dimensionality and for various sample sizes. Since the main purpose of these simulations is to demonstrate the effect of the dimensionality \( N \) and the number \( Q \) of observations on the computation load, the different parameters of the algorithm are not optimized as proposed in Section 4. All runs were made with \( S = 10 \), using the seven first texture features of Table 1, and requiring three texture classes. This strategy allows running the clustering algorithm under comparable conditions for different dimensionalities of the data and different sample sizes.

**Fig. 9** shows the computation times of the procedure for three sample sizes (\( Q = 100, 500 \) and 1000) and for \( N \) varying from 2 to 6. These simulations were performed on a HP Pentium 4/1.7 GHz computer with 128 Mocets memory. Although the running times are computer dependent, they give an idea of the computation time required by the algorithm in a severe case, when seven texture features are taken into account.

These computation times are mainly dependent on the number of sampling points, the data dimensionality \( N \) and the quantization level \( L \). The distribution of the data points is first approximated by the discrete multidimensional histogram with \( S^N \) cells. Thanks to the fast algorithm proposed in Ref. [2], the number of elementary operations required by this procedure is \( N Q \).

Then, for each sampling point, the co-occurrence matrix is generated by considering all the couples of adjacent sampling points encountered within a hypercubic neighborhood of size length \( (2h + 1) \). As there are \( (3^N - 1) \) adjacent sampling points for each of the \( (2h + 1)^N \) sampling points falling in the hypercubic neighborhood, \( (3^N - 1)(2h + 1)^N \) couples of sampling points are considered to compute the co-occurrence values at each sampling point of the discrete multidimensional histogram. Hence, the determination of all the co-occurrence matrices requires \( S^N (3^N - 1)(2h + 1)^N \) operations. When \( h = 1 \), as in all the examples presented in...
this paper, the number of elementary operations is approximately equal to $(9SN)^N$.

Since each co-occurrence matrix must be looped through once to calculate the texture parameters in $L^2$ operations, a total of $(9SN)^N L^2$ operations are required to calculate the texture features for all the sampling points.

The sampling points where the local underlying texture is evaluated are assigned to different texture classes using the K-means algorithm that requires $SN^N Kt$ operations, where $K$ is the number of texture classes and $t$ is the number of iterations necessary for the algorithm to converge.

The connected components are extracted by means of an aggregation procedure where two hypercubes whose centers belong to the same class of texture are assigned to the same component if they have at least one point in common. As $(3^N - 1)$ adjacent neighbors of each sampling point are considered, $(3SN)^N$ operations are required by the connected components extraction procedure.

The core extraction procedure requires the determination of the compactness of all the detected connected components, involving $(3SN)^N$ elementary operations.

 Altogether $(3SN)^N (3^N L^2 + Kt + 2) + NQ$ operations are required by the whole procedure. As $(Kt + 2)$ is generally negligible with respect to $(3^N L^2)$, the total number of operations is roughly equal to $(9SN)^N L^2 + NQ$.

In order to reduce the above evaluated complexity, we have determined the texture parameters only for each non-empty hypercube. When the dimension $N$ increases, the number $R$ of these non-empty hypercubes decreases and tends to an upper limit equal to $Q$ which is generally substantially smaller than $SN$.

The computational load can be further reduced by two solutions proposed in the texture analysis literature. The first one consists in reducing the number $L$ of quantization levels [33], the second one in updating the texture features determined in a hypercubic neighborhood from those obtained in adjacent neighborhoods [34]. We have chosen another well-established solution that consists in storing only the non-zero elements of the co-occurrence matrices [35]. This last solution is well-adapted for large quantization levels $L$, i.e. when the co-occurrence matrix becomes large and sparse. We have used the link list concept [36] to avoid the storage of the pairs of values of the co-occurrence matrices that have zero probability. Each node of that linked list is a structure containing one of the pairs of co-occurring values effectively encountered in the hypercubic neighborhood, its probability of co-occurrence for neighboring sampling points and a link to the next node in the list. To include a new pair in a linked list, a search is performed by finding the node having the same pair of values. If this node is found, then its probability is incremented. Otherwise, a new node is added at the end of the list. The length $G$ of the linked list is equal to the number of distinct pairs of values found in the considered hypercubic neighborhood. It is often significantly shorter than $L^2$, so that the computation requirement of the whole clustering scheme is reduced to $R(3SN^G) + NQ$.

6. Conclusion

A general-purpose clustering procedure has been proposed, based on multidimensional texture analysis. A set of multivariate observations is first preprocessed in order to obtain a multidimensional histogram representing the spatial distribution of the input data as an array of sampling points with integer values. A texture feature vector is then attached to each sampling point lying in a non-empty area of the data space, so that the class detection scheme is based on the classification of these feature vectors rather than on a direct clustering scheme in the initial data space. In this feature space, a simple clustering algorithm performs well for identifying the subsets of sampling points with homogeneous textures that constitute the cluster cores. The detected cores, which are well-separated domains of homogeneous textures in the data space, can virtually be of any shape and size. For classification purposes, the data points falling into these domains are considered as the prototypes of the different clusters. The remaining data points are assigned to their respective clusters by means of the nearest neighbor classification rule.

Experiments show how to select the relevant statistical texture features and how to adjust the resolution of the discretization process. When the texture analysis procedure is compared with classical classification schemes using simulated spherical clusters, it performs comparably well. However, the new procedure is much more efficient in difficult clustering situations such as non spherical or non-linearly separable clusters.

After a series of adaptations of classical image processing tools to cluster analysis such as thresholding, edge detection, relaxation, Markov field models and mathematical morphology, this paper shows that texture analysis has also potential applications in the field of pattern classification.

References


—JACK-GÉRARD POSTAIRE received the Engineering degree from the “Ecole Centrale de Lille”, France, in 1970 and the “Docteur es Sciences” degree in Automatic Control from University Mouloud Mammeri of Tizi Ouzou, Algeria, in 1994. He is now the head of the Automatic Control Department of the Electrical Engineering and Computer Science Faculty of the University Mouloud Mammeri of Tizi Ouzou, Algeria, and the head of the “Robotics and Vision” research laboratory in the same university. His research interest includes image processing and pattern recognition.

About the Author—MOUSSA DIAF was born in 1953 in Algeria. He has received the Engineering degree from the Polytechnique School of Algiers, Algeria, in 1978, the Ph.D. degree in Automatic Control from the University of Sciences and Technologies of Lille, France, in 1983 and the “Docteur es Sciences” degree in Automatic Control from University Mouloud Mammeri of Tizi Ouzou, Algeria, in 1994. He is a permanent teacher in the Automatic Control Department of the Electrical Engineering and Computer Science Faculty of the University Mouloud Mammeri of Tizi Ouzou, Algeria, and the head of the “Robotics and Vision” research laboratory in the same university. His research interest includes image processing and pattern recognition.

About the Author—KAMAL HAMMOUCHE was born in 1965 in Tizi Ouzou, Algeria. He has received the Engineering degree in Electronics in 1989 and the Master of Philosophy degree in Industrial Automatic Control in 1996 from the University Mouloud Mammeri of Tizi Ouzou, Algeria, respectively. He is a permanent teacher in the Automatic Control Department of the Electrical Engineering and Computer Science Faculty of the University Mouloud Mammeri of Tizi Ouzou, Algeria. His research interests are in the area of image processing and pattern recognition.