Watershed-Based Unsupervised Clustering

Manuele Bicego, Marco Cristani, Andrea Fusiello, and Vittorio Murino

Dipartimento di Informatica, Università di Verona
Ca' Vignal 2, Strada Le Grazie 15, 37134 Verona, Italia
{bicego,cristanm,fusiello,murino}@sci.univr.it

Abstract. In this paper, a novel general purpose clustering algorithm is presented, based on the watershed algorithm. The proposed approach defines a density function on a suitable lattice, whose cell dimension is carefully estimated from the data. The clustering is then performed using the well-known watershed algorithm, paying particular attention to the boundary situations. The main characteristic of this method is the capability to determine automatically the number of clusters from the data, resulting in a completely unsupervised approach. Experimental evaluation on synthetic data shows that the proposed approach is able to accurately estimate the number of the classes and to cluster data effectively.

1 Introduction

Unsupervised classification or clustering [1, 2] is undoubtedly an interesting and challenging research area. It could be defined as the organization of a collection of patterns into groups, based on similarity. It is well known that data clustering is inherently a more difficult task if compared to supervised classification, in which classes are already identified, so that a system can be adequately trained. Clustering has been applied in several contexts, as, for example, data mining, DNA modeling, information retrieval, image segmentation, signal compression and coding, and machine learning. Hundreds of clustering algorithms have been proposed in the literature, mostly divided in two categories: iterative partitional and agglomerative hierarchical techniques. The former attempts to obtain a partition of data that minimizes the within-cluster scatter or the between-scatter matrix. The latter organizes the data in a nested sequence of groups organized in a dendrogram which is cut at the chosen depth level in order to obtain the desired number of clusters.

In this paper, a novel clustering scheme is proposed, based on the watershed segmentation algorithm [3, 4] also called watershed transform. This is an effective and accurate method originally conceived in the Mathematical Morphology (MM) field [5] and widely employed in recent years for intensity image segmentation, and video segmentation [6, 7]. The watershed algorithm has also been used in the clustering context, in order to cluster histograms with the aim of color segmentation [8]. The key idea is to consider the gray level picture as a topographic relief, in which to actuate an immersion process.

A. Rangarajan et al. (Eds.): EMMCVPR 2003, LNCS 2683, pp. 83–94, 2003.

In this paper the use of watershed for general clustering purposes is investigated. From the clustering point of view, watershed presents some appealing characteristics: first, it is accurate, as the obtained image segmentation is typically highly informative. Second, and most important, it is an unsupervised method, as the number of clusters does not have to be determined *a priori*. Other techniques, like the K-means or the agglomerative hierarchical family of methods, require the number of clusters to be fixed *a priori*, or to be detected using index like the Davies-Bouldin criterion [9] or some model selection analysis. Another appealing characteristic of the watershed algorithm is that it could be easily extended to deal with n-dimensional spaces [4].

The watershed algorithm is defined over a discrete topological space, where a function defining the "height" of each point should be given. In the case of images, this function is the color intensity of each pixel, but in the clustering context there is no natural choice, and this function should be carefully defined.

In our approach, this function is derived by dividing the space in a set of cells, each of fixed dimension. The height of each cell represents the density of points in that cell, *i.e.* the number of points belonging to the cell. Clearly, the size of the cell is crucial: if too small it could lead to over-segmentation, a too large size could cause a coarse segmentation. In this paper, this problem is carefully addressed, by devising an automatic way for determining the cell size from data.

Preliminary experimental evaluation on synthetic data shows that the proposed approach is quite accurate in discovering the real structure of the data, detecting automatically the number of clusters and their composition.

The rest of the paper is organized as follows. In Section 2 the fundamentals of the watershed algorithms are summarized, and the whole strategy is detailed in Section 3. Section 4 presents experimental evaluation of the proposed method, and in Section 5 conclusions are drawn and future perspectives are investigated.

2 The Watershed Algorithm

In the field of image processing and more particularly in Mathematical Morphology (MM) [5], gray-scale pictures could be considered as topographic reliefs, in which the numerical value of each pixel of a given image I represents the elevation at that point. In such a context, the image segmentation could be obtained by the watershed transform, a technique originally proposed by Digabel and Lantuejoul [10]. The intuitive idea under this segmentation method is the following: imagine that the image-landscape I is immersed in a lake, with holes pierced in local minima. Basins (also called "catchment basins") will be filled up with water starting at these local minima, and, at points where water coming from different basins meet, dams are built. When the water level has reached the highest peak in the landscape, the process is stopped. As a result, the landscape is partitioned into regions or basins separated by dams, called *watershed lines* or simply *watersheds*. For the sake of clarity, we will use the expression "watershed transform" to denote a labeling of the topographic space, such that all points of a given catchment basin have the same unique label, and a special label, distinct

from all the other labels of the catchment basins, is assigned to all point of the watershed.

Many sequential algorithms have been developed to compute watershed transform (see [11] for a critical review). They can mainly be divided into two classes: the first one is based on the algorithm proposed by Vincent and Soille in [4]; the second one is based on distance functions, and was firstly proposed by Meyer [12]. For our clustering purpose, we prefer the first approach, that is very general: its adaptation to any kind of underlying grid (4-, 6-, 8-connectivity) is straightforward, and it can be easily extended to n-dimensional spaces.

The following subsections present the watershed algorithm, following a definition that is know in literature as *algorithmic definition*.

2.1 Definitions

Let I be the topographic space, 2D for simplicity, whose definition domain is denoted $D_i \subset \mathcal{Z}^2$. I is supposed to take discrete values in a given range $[0, N]$, $N \in \mathcal{N}$. Let $G \subset \mathcal{Z}^2 \times \mathcal{Z}^2$ denote an underlying digital grid, in 8-connectivity for example. We could define the following entities:

Definition 1. *A path P of length l between two points p and q in G is a $(l+1)$-tuple of points $(p_0, p_1, \ldots, p_{l-1}, p_l)$ such that $p_0 = p$, $p_l = q$, and $\forall i \in [1, l], (p_{i-1}, p_i) \in G$. We will define $l(P)$ the length of a given path P, $N_G(p) = \{p' \in \mathbf{Z}^2, (p, p') \in G\}$ the neighbors of a point p, with respect to G.*

Definition 2. *A minimum M of I at altitude h is a connected plateau of points of height h from which it is impossible to reach a lower height point without having to climb:*

$$\begin{aligned} \forall p \in M, \forall q \notin M, \text{ such that } I(q) \leq I(p), \\ \forall P = (p_0, p_1, \ldots, p_l) \text{ such that } p_0 = p \text{ and } p_l = q, \\ \exists i \in [1, l] \text{ such that } I(p_i) > I(p_0). \end{aligned} \tag{1}$$

Definition 3. *The geodesic distance $d_A(x, y)$ between two points x and y in A (set of points simply connected in G) is the minimum length of the paths which join x and y that are totally included in A:*

$$d_A(x, y) = inf\{l(P),\ P \text{ path between } x \text{ and } y \text{ which is totally included in } A\}. \tag{2}$$

Let $B \subset A$ made of several connected components $B_1, B_2, \ldots, B_k$.

Definition 4. *The geodesic influence zone $iz_A(B_i)$ of a connected component B_i of B in A is formed by those points in A whose geodesic distance to B_i is smaller than their geodesic distance to any other component of B:*

$$iz_A(B_i) = \{p \in A,\ \forall j \in [1, k]/\{i\}, d_A(p, B_i) < d_A(p, B_j)\}. \tag{3}$$

The points of A not belonging to any geodesic influence zone form the *skeleton by influence zones (SKIZ)* of B inside A:

$$\mathrm{SKIZ}_A(B) = A/\mathrm{IZ}_A(B) \text{ with } \mathrm{IZ}_A(B) = \bigcup_{i\in[1;k]} iz_A(B_i). \tag{4}$$

2.2 The Watershed Transform

To reproduce the immersion procedure described above, we start from the set $T_{h_{min}}(I) = \{p \in D_I, I(p) \leq h_{min}\}$ of the points first reached by the water. These points constitute the starting set of our recursion. Thus, we set

$$X_{h_{min}} = T_{h_{min}}(I). \tag{5}$$

$X_{h_{min}}$ is composed by the points of I which belong to the minima of lowest altitude. Let us now consider the threshold of I at level $h_{min}+1$, i.e., $T_{h_{min}+1}(I)$. Now, if Y is one of the connected components of $T_{h_{min}+1}(I)$, there are three possible relations of inclusion between Y and $Y \cup X_{h_{min}}$:

1. $Y \cup X_{h_{min}} = \emptyset$: Y is a new minimum of I. Indeed, according to the definitions above, Y is a plateau at level $h_{min} + 1$, since:

$$\forall p \in Y \begin{cases} p \notin X_{h_{min}} \Rightarrow I(p) \geq h_{min} + 1 \\ p \in Y \Rightarrow I(p) \leq h_{min} + 1 \end{cases} \tag{6}$$

   Moreover, all the surrounding points do not belong to $T_{h_{min}+1}(I)$ and have a function value strictly greater than $h_{min} + 1$. The minimum discovered is "pierced", hence, its corresponding catchment basin will be progressively filled up with water.
2. $Y \cup X_{h_{min}} \neq \emptyset$ and is connected: in this case Y corresponds exactly to the pixels belonging to the catchment basin associated with the minimum $Y \cup X_{h_{min}}$ and having a gray level lower than or equal to $h_{min} + 1$:

$$Y = C_{h_{min}+1}(Y \cup X_{h_{min}}). \tag{7}$$

   where $C(M)$ is the catchment basin associated with a minimum M, and $C_h(M)$ is the subset of this catchment basin made of points having an altitude smaller or equal to h:

$$C_h(M) = \{p \in C(M), I(p) \leq h\} = C(M) \cup T_h(I) \tag{8}$$

3. $Y \cup X_{h_{min}} \neq \emptyset$ and is not connected: we therefore notice that Y contains different minima of I. Denote $(Z_1, Z_2, \ldots, Z_k)$ these minima. In this situation, the best possible choice for $C_{h_{min}+1}(Z_i)$ is given by the geodesic influence zone of Z_i inside Y:

$$C_{h_{min}+1}(Z_i) = iz_Y(Z_i). \tag{9}$$

Since all possibilities have been discussed, we take as second set of our recursion:

$$X_{h_{min}+1} = \min_{h_{min}+1} \cup IZ_{T_{h_{min}+1}(I)}(X_{h_{min}}). \tag{10}$$

This relation holds for all levels h, and finally, we obtain the following definition:

Definition 5. *(Catchment basins and watershed by immersion): the set of the catchment basins of the function I is equal to the set $X_{h_{max}}$ obtained after the following recursion:*

$$a) X_{h_{min}} = T_{h_{min}}(I), b) \forall h \in [h_{min}, h_{max}-1], X_{h_{min}+1} = min_{h+1} \cup IZ_{T_{h+1}(I)}(X_h) \tag{11}$$

The watershed of I corresponds to the complement of this set in D_I,i.e. to the set of the points of D_I that do not belong to any catchment basin.

Our watershed algorithm is based on the above definitions, and is thoroughly described in [4]. We consider the subsequent height levels of the topographic space examined, and compute the *geodesic influence zones* on the basis of the labeling of the previous level.

The watershed algorithm is realized in two steps: the first consists in an initial sorting in increasing order of the values of the pixels. In the second step, the flooding phase, the geodesic influence zones are computed by performing a breadth-first scanning of each height level. Suppose the flooding of the catchment basins has been done up to a given level h. Each catchment basin already discovered is supposed to have a unique label. Starting from the pixels that have at least one neighbor already labeled, we compute the geodesic influence zone in order to extend the labeled catchment basins. After this step, only the *minima* at level $h+1$ have not been reached (they are not connected to any of the already labeled catchment basin). Therefore, a second scanning of the pixels at level $h+1$ is necessary to detect and to label the new minima. This procedure stops when the highest pixel has been examined.

3 The Proposed Strategy

In this section, the proposed strategy is detailed. The first goal is to obtain a height function from data, in order to transform the feature space into the topographic space. To this end, the problem space is divided into cells of fixed squared size, and a function is defined over these cells. More formally, given a set of D-dimensional samples $\mathcal{Y} = \mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_N$, where each sample is $\mathbf{y}_i = y_{i,1}, y_{i,2}, ...y_{i,D}$, the discretization process defines a lattice $\mathcal{R}$ on this D-dimensional space. The origin O of this lattice is the minimum over all dimensions, *i.e.*

$$O = [\min_n y_{n,1}, \min_n y_{n,2}, ..., \min_n y_{n,D}] \tag{12}$$

A diagonal transformation is then performed, which stretches the scale of the axes of the data space in order to standardize the range of each feature, such that

$$\forall d \qquad \max_n y'_{n,d} - \min_n y'_{n,d} \equiv k \tag{13}$$

where $\{y'_{n,d}\}$ are the points in the transformed space. The constant k represents the maximum dimension width of the feature space, *i.e.*

$$k = \max_d \left(\max_n y'_{n,d} - \min_n y'_{n,d} \right) \tag{14}$$

In this way we could define the cells as D-dimensional hypercubes of fixed size $\ell_{\mathcal{R}}$. Let us denote the cell in the position $\mathbf{i} = (i_1, ..., i_D)$ as $R(\mathbf{i}) = R(i_1, ..., i_D)$. Obviously, the choice of the parameter $\ell_{\mathcal{R}}$ is critical. Before addressing the problem of calculating $\ell_{\mathcal{R}}$, let us define the function I used for watershed clustering.

Once fixed $\ell_{\mathcal{R}}$, we have a discrete-lattice of $\left(\frac{k}{\ell_{\mathcal{R}}}\right)^D$ cells, describing the feature space. The height function is then defined on this lattice: the value of the function in a cell is the number of points belonging to that cell. In other words, the function value in one cell measures the density of points in that part of the problem space. More formally, the function $I(R(\mathbf{i}))$ is defined as follows:

$$I(R(\mathbf{i})) = \sum_{\mathbf{y}_n \in \mathcal{Y}} \chi_{R(\mathbf{i})}(\mathbf{y}_n) \tag{15}$$

where χ is the characteristic function of the set $R(\mathbf{i})$, defined as

$$\chi_{R(\mathbf{i})}(\mathbf{y}_n) = \begin{cases} 1 & \text{if } \mathbf{y}_n \in R(\mathbf{i}) \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

This function reflects the density properties of the clustering space: assuming that similar points (*i.e.* points that belong to the same cluster) are near in the feature space this function assumes high values in proximity of parts of the space where several similar points are present, while in the boundary (low density parts) assumes low values. By inverting all the values of this function, all the highest values are considered as local minima, from which the recursive process of the watershed transform can adequately start.

Let us now come back to the determination of the cell size $\ell_{\mathcal{R}}$. This represents obviously a crucial choice. If the cell is too small, this could results in a non informative representation, and the watershed algorithm will tend to produce an over-segmentation. On the other side, a too large value could lead to a coarse clustering; if the cell contains points too much far apart, the boundary could not be easily estimated, resulting in a quite rough separation between clusters. In our approach, the dimension of the cell is estimated by making a direct usage of the data. A good compromise between over-segmentation and rough clustering could be obtained by linking the choice of the $\ell_{\mathcal{R}}$ parameter to the median

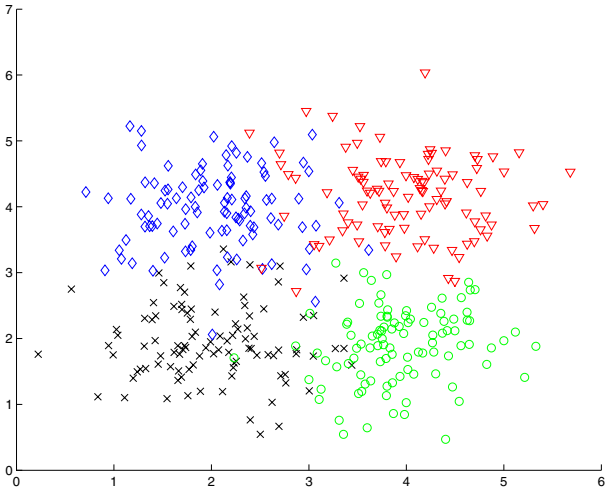


Fig. 1. Synthetic generated data, for a 4 clusters problem with a variance equal to 1

of the pairwise distances between all points. In particular, we compute all distances $d(\mathbf{y}_i, \mathbf{y}_j)$ $(\forall i, j \in 1..N)$, then we extract the median value and fix the $\ell_{\mathcal{R}}$ parameter to

$$\ell_{\mathcal{R}} = \frac{\text{median}(d(\mathbf{y}_i, \mathbf{y}_j))}{m} \tag{17}$$

where m is a constant that has been experimentally fixed to 4, for all evaluated data sets . The use of the median, instead of the mean, allows to gain robustness against outliers. After defining the height function, the clustering is obtained applying to the lattice $\mathcal{R}$ the watershed algorithm described in Section 2. A problem that occurs is represented by the watersheds, *i.e.* the lines that divides the clusters. In our case, each watershed has width equal to one cell: the points in the cell are unlabeled, and have to be assigned to some clusters. To this end we use the following procedure. Starting from the consideration that each line divides only near clusters, we could decide to which of the near groups each point in the watershed belongs. In order to do that, we simply perform a new clustering on the watershed points, using the standard K-means algorithm. This clustering is really fast and quite accurate, since the watershed cells contain only few points, and the number of clusters is known (number of neighbors). After performing the sub-clustering, each mini-cluster is assigned to the nearest maxi-cluster, identified by determining the distance from the centroid of the nearest cells. By the use of this algorithm, the clustering boundaries are refined, and results are more accurate.

4 Experimental Evaluation

In this section, the proposed clustering method is tested, in order to assess its validity in synthetic cases. The following examples have been chosen to get some insight into the behavior of the watershed transform in the context of cluster analysis and to demonstrate the interest of this approach to pattern classification. The proposed approach is compared to the standard K-means algorithm [1, 13]: this approach finds the optimal partition by evaluating, at each iteration, the distance between each item and each cluster descriptor, and by assigning it to the nearest class. At each step, the descriptor of each cluster is re-evaluated by averaging its cluster items. The system stops when no changes are produced in the clustering. In the K-means algorithm, the number of clusters should be decided *a priori*.

We present results obtained on different synthetic problems, varying the difficulty of the task and the number of the clusters. Fixed K the number of clusters, synthetic data are generated according to a K 2D Gaussians $\mathcal{N}(\mu_i, \sigma^2)$, $i = 1 \ldots, K$, sharing the same common variance. The means are randomly placed in the space, drawn from an uniform distribution in the interval $[-5, 5], [-5, 5]$. We vary the variance of the Gaussians in order to drive the difficulty of the problem: the higher the variance, the more overlapped the clusters, implying a more difficult task. For each Gaussian 200 elements have been drawn. An example of

Table 1. Clustering accuracies (means and standard deviations) for the synthetic experiment: (a) 2 clusters; (b) 3 clusters; (c) 4 clusters

(a)

variance σ^2	K-Means Accuracy mean	std	Watershed Accuracy mean	std
0.5	99.50%	2.08%	98.79%	1.95%
1.0	98.10%	3.88%	97.03%	5.20%
1.5	96.50%	4.54%	95.41%	4.53%
2.0	94.43%	4.57%	91.00%	2.56%

(b)

variance σ^2	K-Means Accuracy mean	std	Watershed Accuracy mean	std
0.5	91.90%	13.65%	96.22%	8.23%
1.0	93.86%	8.84%	87.01%	13.18%
1.5	91.53%	8.18%	80.16%	14.82%
2.0	90.05%	5.82%	71.85%	16.40%

(c)

variance σ^2	K-Means Accuracy mean	std	Watershed Accuracy mean	std
0.5	88.39%	13.71%	88.38%	11.02%
1.0	91.01%	9.21%	80.63%	9.06%
1.5	88.92%	5.84%	74.08%	10.32%
2.0	85.33%	4.54%	67.11%	11.19%

Table 2. Average number of clusters estimated by the proposed approach, for different variances and for different number of true clusters

	2 clusters	3 clusters	4 clusters
$\sigma^2 = 0.5$	2.02	3.08	3.72
$\sigma^2 = 1.0$	2.21	3.27	3.75
$\sigma^2 = 1.5$	2.31	3.60	4.26
$\sigma^2 = 2.0$	2.52	3.37	4.43

the generated data is presented in Fig. 1, where a 4 clusters problem is displayed (variance is 1). One can notice that there is a visible overlapping between these distributions, and the problem is quite difficult.

Experiments are repeated 100 times, in order to assess the statistical significance of the results and moreover to minimize the very poor performances of K-means due to wrong initialization. The accuracy of the clustering could be quantitatively assessed, by computing the number of wrongly composed clusters: a clustering error occurs if a pattern is assigned to a cluster in which the majority of the patterns are from another source. The obtained averaged accuracies, together with the standard deviations, are presented in Table 1, for different number of clusters. From this table it is evident that the accuracy of the proposed approach is slightly worse than that of the K-means. With our algorithm, nevertheless, the number of clusters is properly detected in almost all experiments, resulting in a completely unsupervised approach, differently than in the K-means case.

The watershed algorithms is more sensitive to higher variances, since the boundaries could not be easily estimated and the resulting clustering could be

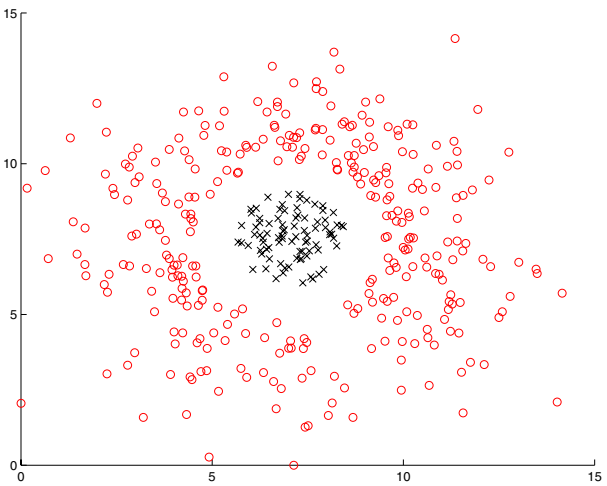
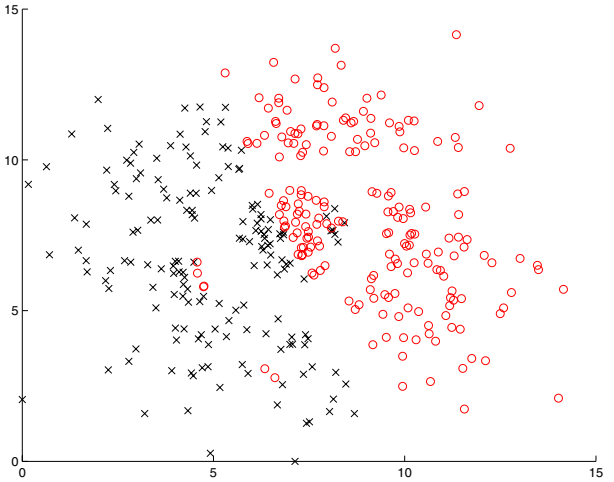
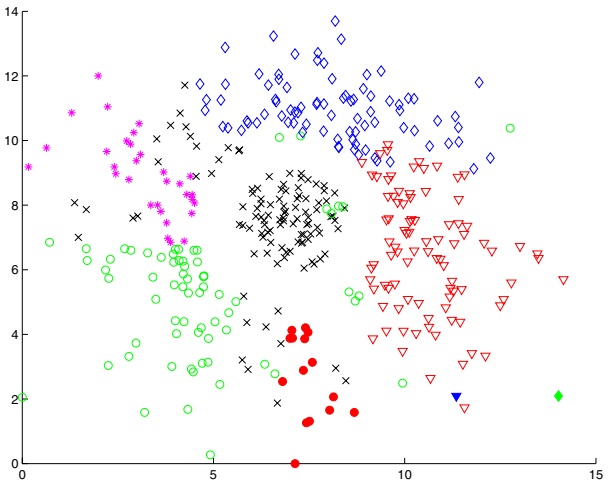


Fig. 2. Data generated from two concentric clusters



(a)



(b)

Fig. 3. Clustering obtained on the concentric clusters problem: (a) K-means; (b) Watershed algorithm

poor. Due to its unsupervised nature, the watershed algorithm performances worsen when increasing the number of true clusters.

Estimated numbers of clusters, determined in each problem, are shown in Table 2, for different Gaussians variance.

One can notice that the watershed algorithm is quite effective in estimating the number of clusters of the problem. Nevertheless, except than in the case of 5 clusters, it tends to over-estimate this number: this represents a well-known

problem of the watershed algorithm, already encountered in the image segmentation literature. In [8], this problem was faced by applying a Gaussian filter in the space: but this process, nevertheless, suppresses also some important minima, so it was not used here. In our approach, a smoothing process is performed during the lattice construction operation, since the size of the cell determines the smoothness of the resulting function.

We tested our approach on another synthetic example, with two concentric clusters, presented in Fig. 2. This clustering experiment is rather difficult. It is well known [1] that the K-means algorithm is not able to deal with this problem: the obtained segmentation is proposed in Fig. 3(a), and is a completely wrong clustering. We applied our watershed algorithm to this problem: the obtained clustering is presented in Fig. 3(b). We can note that the central cluster is correctly identified, but the outside cluster is over segmented. Nevertheless, this represents an improvement with respect to the clustering obtained with the K-means approach, since at least some part of the semantic information is discovered, considering also the fact that this approach is completely unsupervised.

5 Conclusions

In this paper, a novel method for clustering points has been proposed, based on the watershed algorithm. The system automatically derives a discrete lattice from the feature space, and defines a height function. Watershed is then performed in that lattice. Boundary situations are then addressed with a intra-cell analysis, able to remove the watershed lines not needed in the clustering process. The main advantage of this method is its completely unsupervised nature, since it is able to automatically discover the number of clusters of the data. The main problem of this approach is the tendency to produce an over-segmentation, which is intrinsic in the nature of the watershed algorithm. In our opinion, this could be faced by working on the lattice definition, and this will be an issue of a future investigation.

References

[1] Jain, A., Dubes, R.: Algorithms for clustering data. Prentice Hall (1988) 83, 90, 93

[2] Jain, A., Murty, M. N., Flynn, P.: Data clustering: A review. ACM Computing Surveys **31** (1999) 264–323 83

[3] Beucher, S.: Watersheds of functions and picture segmentation. In: IEEE Proc. of Int. Conf. Acoustics, Speech and Signal Processing. (1982) 1928–1931 83

[4] Vincent, L., Soille, P.: Watersheds in digital spaces: an efficient algorithm based on immersion simulations. IEEE Trans. on Pattern Analysis and Machine Intelligence **13** (1991) 583–589 83, 84, 85, 87

[5] Serra, J.: Image Analysis and Mathematical Morphology. Academic Press, London (1982) 83, 84

[6] Patras, I., Hendriks, E., Lagendijk, R.: Video segmentation by map labelling of watershed segments. IEEE Trans. on Pattern Analysis and Machine Intelligence **23** (2001) 326–332 83

[7] Hung, Y. P., Tsai, Y. P., Lai, C. C.: A bayesian approach to video object segmentation via merging 3D watershed volumes. In: IEEE Proc. of Int. Conf. on Pattern Recognition. Volume 1. (2002) 496–499 83
[8] Geraud, T., Strub, P. Y., Darbon, J.: Color image segmentation based on automatic morphological clustering. In: IEEE Proc. of Int. Conf. on Image Processing. Volume 3. (2001) 70–73 83, 93
[9] Davies, D., Bouldin, D.: A cluster separation measure. IEEE Trans. on Pattern Analysis and Machine Intelligence **1** (1979) 224–227 84
[10] Digabel, H., Lantujoul, C.: Quantitative analysis of microstructures in materials sciences. Dr. Riederer-Verlag GmbH, Stuttgart (1978) 84
[11] Roerdink, J. B. T. M., Meijster, A.: The watershed transform: Definitions, algorithms and parallelization strategies. Fundamenta Informaticae **41** (2000) 187–228 85
[12] Meyer, F.: Topographic distance and watershed lines. Signal Processing **38** (1994) 113–125 85
[13] Ball, G., Hall, D.: A clustering technique for summarizing multivariate data. Behavioral Science **12** (1967) 153–155 90