Pobrane z czasopisma Annales AI- Informatica **http://ai.annales.umcs.pl**
Data: 26/11/2026 11:45:42

Annales UMCS Informatica AI 8(1) (2008) 111-121
10.2478/v10065-008-0011-7

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI

http://www.annales.umcs.lublin.pl/

Watershed merging method for color images

Jakub Smołka*, Maria Skublewska-Paszkowska

Institute of Computer Science, Faculty of Electrical Engineering and Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract

Watershed transformation can be applied to color as well as to gray-scale images. A problem arises when dealing with color images. It is caused by the fact that pixels in such images are vectors that describe all color components whereas the watershed transformation requires a scalar height function as its input. There are multiple gradient magnitude definitions for color images that allow for the needed conversion. As in the case of gray-scale images, the image after watershed transformation is heavily over-segmented. One can blur the image before calculating the gradient magnitude, threshold the gradient image or merge the resulting watersheds. Unfortunately, the result is still over-segmented.

A solution presented in this paper complements those mentioned above. It uses hierarchical cluster analysis methods for joining similar classes of the over-segmented image into a given number of clusters. After the image has been preprocessed and segmented, the over-segmentation is reduced by means of the cluster analysis. The attribute values for each watershed in each color component are calculated and clustering is performed. The resulting similarity hierarchy allows for the simple selection of the number of clusters in the final segmentation.

Several clustering methods, including complete linkage and Ward's methods with different sets of components, have been tested. Selected results are presented.

1. Introduction to the watershed transformation

Beucher and Lantuejoul [1] introduced watershed transformation - an image segmentation method that mimics pouring water onto a landscape. It divides an image into watersheds (or catchment-basins) whose edges are continuous. The method requires an image with scalar valued pixels as its input. The input image is treated as a height function. Higher values indicate the presence of edges, while local minima are where the catchment-basins originate (pouring water collects in valleys). Gray-scale images can directly be used as the watershed's transformation input, but this usually is not the case because in most cases high

* Corresponding author: *e-mail address*: jakub.smolka@pollub.pl

Pobrane z czasopisma Annales AI- Informatica **http://ai.annales.umcs.pl**
Data: 26/11/2026 11:45:42

values do not indicate edges. Such images need to be transformed using some sort of edge detection filter. A gradient magnitude filter is well suited for this task [2,3]. Calculating the gradient for grayscale image is quite straightforward. Color images, on the other hand, cannot be used directly by the watershed transformation. This is because their pixels are vector valued. Many different approaches to calculating the gradient magnitude of a color image exist. Regardless of the type of the image being segmented, the result of the watershed transformation is usually over-segmented. This paper describes how one can deal with the over-segmentation problem in the case of color images.

2. Color image gradient

As mentioned above, color images, since their pixels are vectors, need to be converted into a height function. Color sRGB images can be described and interpreted as a multivalued function [4]

$$\Phi(x_1,x_2)\colon \mathbb{R}^2 \to \mathbb{R}^3, \tag{1}$$

whose components are

$$\Phi_j(x_1,x_2)\colon \mathbb{R}^2 \to \mathbb{R}^3 \colon j = 1..3 \tag{2}$$



a) b) c)

Fig. 1. a) Original sRGB image (budynek_1), b) PCA gradient, c) sum-of-squares gradient

Two types of filters have been used for the purpose of this paper. The first type is a heuristic method that finds the gradient magnitude as the square root of the sum of the individual vector component derivative squared [5]. For a two dimensional image with three components, the formula takes the following form:

$$M = \sqrt{\sum_{i=0}^{2}\sum_{j=0}^{3}\left(\frac{\partial \Phi_j}{\partial x_i}\right)}. \tag{3}$$

This method will be refered to as “sum-of-squares” further in this paper.

The second type of filter used, is based on principal component analysis. It was proposed by Sapiro and Ringach [4]. It calculates the gradient magnitude as the difference between the two largest eigenvalues in the principal component

Pobrane z czasopisma Annales AI- Informatica **http://ai.annales.umcs.pl**
Data: 26/11/2026 11:45:42

analysis of the partial derivatives of the color components [5]. For a two dimensional image the gradient magnitude formula takes the following form [6];

$$M = \lambda_+ - \lambda_- \tag{4}$$

with eigenvalues given by

$$\lambda_\pm = \frac{g_{11} + g_{22} \pm \sqrt{g_{11} - g_{22}^2 + 4g_{12}^2}}{2}, \tag{5}$$

where

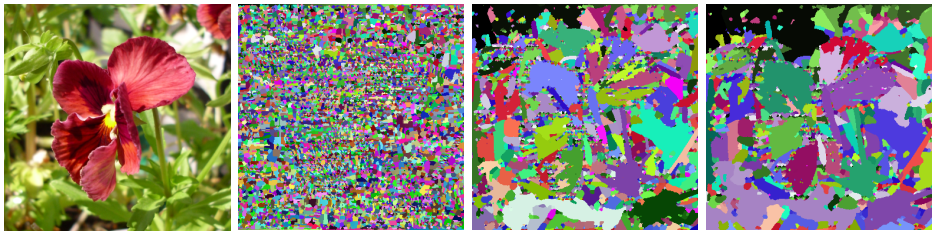
$$g_{ij} = \frac{\partial \Phi}{\partial x_i} \cdot \frac{\partial \Phi}{\partial x_j}. \tag{6}$$

The λ_+ and λ_- eigenvalues can be interpreted, correspondingly, as the maximal and minimal rates of change [4]. This method will be referred to as "PCA gradient" further in this paper.

The main practical difference between these two ways of obtaining the gradient magnitude is that the PCA gradient brings out significant edges while diminishing those less important; the sum-of-squares gradient, on the other hand, is better when minor details are important. Usually the latter causes stronger over-segmentation.

There are many more methods for calculating color image gradient magnitude, including: luminance gradient, hue circular gradient, saturation weighing-based color gradient, supremum-based color gradient, chromatic gradient and perceptual gradient. These methods have been described by Angulo and Serra [7].

3. Methods for over-segmentation reduction



a) b) c) d)

Fig. 2. Results of applying different pre- and post-processing parameters a) Original sRGB image (kwiatek_3), b) 0% threshold and 0% merging level result in 10084 watersheds being created; c) 5% threshold, 0% merging level, 3489 watersheds; d) 5% threshold, 10% merging level, 1965 watersheds

If the watershed transformation is performed on a gradient image (regardless of the type of the applied gradient filter) without any other processing, its result will be heavily over-segmented (see Fig. 2b). In most cases such a large number

Pobrane z czasopisma Annales AI- Informatica **http://ai.annales.umcs.pl**
Data: 26/11/2026 11:45:42

of watersheds is useless and significantly increases the computational cost of merging with cluster analysis. Most of the detected edges are insignificant. There are two simple methods (which can be used together) that allow for a significant reduction in the watershed number without a loss of segmentation quality. The first of these methods is preprocessing the gradient image with thresholding. It is performed on the gradient image before the watershed transformation. Its goal is to eliminate low gradient magnitude values. These are caused by noise, texture or weak edges. In this paper the applied threshold is expressed as the percentage of the maximum watershed depth in the gradient image [6]. The second method is a post-processing step which consists in merging neighbouring watersheds. One watershed floods its neighbour if the neighbour's depth is below a certain level called merging level [6]. Like the threshold it is expressed as the percentage of the maximum watershed depth.

Figure 2 shows the effects of using the described methods. Fig. 2a includes the original image and Fig. 2b the result of applying an edge-preserving smoothing filter [8], a sum-of-squares gradient and a watershed transformation. The watersheds have been colored using a hashing scheme in order to make them more visible. Fig. 2c presents the effect of thresholding the gradient image with a 5% threshold and Fig. 2d depicts the effect of thresholding and merging watersheds up to a 10% level. As one can see, thresholding reduced the number of watersheds by a factor of 2.9, and thresholding combined with merging reduced this number by a factor of 5.1. It should be also noted that increasing the threshold and merging level may lead to watersheds belonging to completely different objects being merged (under-segmentation). That is why the described two methods are usually insufficient.

4. Watershed attributes used for merging

For the purpose of this paper, the results of using three different attributes were obtained. In the initial research on the usability of hierarchical cluster analysis for watershed merging in grayscale images [9], four attributes were used but only three have turned out to be useful. The watershed's size did not help create good segmentations. Other attributes will be described in more detail.

Since RGB images have three color components, each attribute is calculated separately for each of them. Consequently, each watershed is described with three times as many values as in the case of a gray-scale image. Of course, it is possible to calculate an attribute only for selected color components. This possibility will be investigated, especially using other color spaces than sRGB.

The first kind of watershed attribute is its *mean value*:

$$\mu_i = \sum_{j=1}^{t} \frac{X_{ij}}{t}. \tag{7}$$

Pobrane z czasopisma Annales AI- Informatica **http://ai.annales.umcs.pl**
Data: 26/11/2026 11:45:42

An example is shown in figure 3b. Visualizations of this attribute closely resemble the original (fig. 3a) image when it is strongly over-segmented.

The other two attributes are similar. They are *variance*:

$$\sigma_i^2 = \sum_{j=1}^{t} \frac{\left(X_{ij} - \bar{X}_i\right)^2}{\left(t-1\right)} \tag{8}$$

and standard deviation: σ_i.

These attributes are, to some extent, sensitive to texture within the watershed. Even regions that are practically identical when their mean value is compared (figure 3b) may be visibly different when their variance or standard deviation is taken into account (figures 3c and 3d).



a) b) c) d)

Fig. 3. Different types of attributes used: a) original sRGB image (kwiatek_3), b) RGB averages, c) RGB standard deviation, d) RGB variance

5. Cluster analysis used for watershed merging

Since basic methods for removing watersheds cannot usually remove over-segmentation, the use of cluster analysis for merging watersheds in color images is proposed. This approach was quite successful with gray scale images [9,10]. Cluster analysis is an iterative process where, in each iteration, the two most similar clusters are found and merged. The merges are based on a similarity/ dissimilarity matrix which is updated in each iteration. Three clustering methods were used for this preliminary comparison: *complete linkage* (CLINK), *unweighted pair-group method using arithmetic averages* (UPGMA) and *Ward's minimum variance* method. They were chosen because they performed the best in the comparison described in [10]. Single linkage (SLINK) has been left out because it did not give satisfactory results [9,10].

The CLINK and UPGMA methods are very similar [11]. They differ only in the way the similarity matrix is modified, that is, how the distances between the newly created cluster (newly merged cluster) and the remaining clusters are determined. With the CLINK method, the distance between two clusters is the same as that between the two most dissimilar objects (i.e. watersheds) they consist of [11]. The UPGMA method averages the distances between all possible pairs of objects (each pair must consist of objects belonging to different clusters

Pobrane z czasopisma Annales AI- Informatica **http://ai.annales.umcs.pl**
Data: 26/11/2026 11:45:42

before the merge) [11]. More formally, the CLINK method calculates new distances using the following equation

$$d_{C_1C_2} = \max_{\substack{m\in C_1 \\ n\in C_2}} d_{nm}, \tag{9}$$

while UPGMA uses

$$d_{C_1C_2} = \frac{1}{n_{C_1} \cdot n_{C_2}} \sum_{\substack{m\in C_1 \\ n\in C_2}} d_{nm}, \tag{10}$$

where: C_1, C_2 – clusters, m – object that belongs to cluster C_1, n – object that belongs to cluster C_2, d_{nm} – distance between objects m and n,– dissimilarity measure of clusters C_1 and C_2 (one of them is a cluster that just has been merged), – number of objects in cluster C_1 – number of objects in cluster C_2.

Ward's method is sometimes called the minimum variance method. It differs from CLINK and UPGMA not only in the way it finds clusters to merge but also in the sense that it does not need any additional distance measuring coefficient. In this method the fusion of two clusters is based on the size of an error sum of squares criterion [12]. The algorithm does not search for the most similar clusters (with the help of similarity matrix); instead, it attempts to find an optimal merger, such that it causes a minimal increase in the total within-cluster error sum of squares, E, given by

$$E = \sum_{m=1}^{c} E_m, \tag{11}$$

$$E_m = \sum_{l=1}^{t_m} \sum_{i=1}^{n} \left(X_{ilm} - \bar{X}_{im} \right)^2, \tag{12}$$

where: E – total error sum of squares, E_m – error sum of squares of *m-th* cluster, c – number of clusters, n – number of attributes, t_m – number of objects in *m-th* cluster, - the average value of *i-th* attribute in *m-th* cluster, X_{ilm} – value of the *i-th* attribute of the *l-th* object's in the *m-th* cluster.

As a result, in each iteration the algorithm has to check all possible mergers.

As mentioned above, the CLINK and UPGMA methods need a similarity/dissimilarity matrix as their input. Such a matrix can be obtained by using the *Euclidean distance coefficient* [11] given by

$$e_{jk} = \sqrt{\sum_{i=1}^{n} \left(X_{ij} - \bar{X}_{ik} \right)^2}, \tag{13}$$

where: *j, k* – numbers of objects, n – number of attributes, X_i – value of *i-th* attribute of *j-th* object, X_{ik} – value of *i-th* attribute of *k-th* object.

It is a dissimilarity measure which represents the distance between two points in the n-dimensional space.

Pobrane z czasopisma Annales AI- Informatica **http://ai.annales.umcs.pl**
Data: 26/11/2026 11:45:42

The complete segmentation process based on watershed transformation proceeds as follows: (1) the image is filtered using an edge preserving smoothing filter [8], (2) either the sum-of-squares or the PCA gradient is applied to the filtered image, (3) the obtained gradient magnitude image is thresholded, (4) the watershed transformation is applied, (5) the neighbouring watersheds are merged based on their depth, (6) the number of resulting watersheds is determined, (7) the attribute values are calculated for each watershed, (8) the similarity/dissimilarity matrix is determined using the distance coefficient (in the case of CLINK and UPGMA methods), (9) the algorithm finds the two most similar clusters and merges them; additionally, it updates the similarity hierarchy which is represented by a tree (10) the similarity/dissimilarity matrix is updated (in the case of CLINK and UPGMA methods), (11) if there is more than one cluster left, the algorithm goes back to step (9), (12) based on the similarity hierarchy (tree) and the requested number of classes, the final segmentation is generated. The final step is not time-consuming; hence, the class count can be changed interactively.

The following pseudo-code provides a more formal description of the proposed method:

```
I_f:=edgePreservingSmooth(I);          //reduce noise
I_g:=computeGradientMagnitude(I_f);    //convert image into a height function
I_t:=thresholding(I_g,t);              //eliminate small gradient values
                                       //(reduce over-segmentation)
W:=watershedTransformation(I_t);       //create watersheds
W_m:=mergeWatersheds(W,l);             //reduce over-segmentation
c:=getNumberOfWatersheds(W_m);
A:=computeAttributes(W_m,c);
S:=computeDissimilarityMatrix(A,c);    //or similarity matrix
while (c>1)
   //saves results to first, second and distance
   findMostSimilarWatersheds(S,first,second,distance);
   //updates the similarity hierarchy (combines two clusters)
   addToTree(first,second,distance,tree);
   //calculates similarity measures for the new cluster
   updateSimilarityMatrix(S,first,second);
   c:=c-1;
end while;
Segmentation:=cutSimilarityTree(tree,classes);
```

where: I – original image, I_f – smoothed image, I_g – gradient magnitude image, I_t – thresholded gradient image, t – applied threshold, W – watershed image, l – watershed merging level, W_m – image with merged watersheds, c – current

Pobrane z czasopisma Annales AI- Informatica **http://ai.annales.umcs.pl**
Data: 26/11/2026 11:45:42

number of clusters (initially number of watersheds), *A* – array holding watershed attribute values, *S* – dissimilarity (or similarity) matrix, *first/ second* – numbers of clusters to be combined, *distance* – dissimilarity (or similarity) measure of clusters to be combined, *tree* – tree holding the similarity hierarchy, *classes* – requested number of classes.

6. Results

Prior to the segmentation, all test images were preprocessed. First, the edge preserving filter was applied (the curvature anisotropic diffusion filter [5] with the following parameters: conductance=0,3, Δt=0,12 and 5 iterations). Second, the gradient image was calculated, and third, the image was thresholded. In the presented results, the threshold value is given as the percentage of the maximal value present in the gradient image. Depending on the image, a different type of gradient filter was used. It was either a PCA or a sum-of-squares gradient [5]. The reason is that, generally, images produced by the PCA filter, when compared to the results of using the sum-of-squares, contain fewer visible edges (lower values). This makes selecting the proper threshold for preprocessing more difficult. By proper threshold the authors mean: "such threshold value that does not cause significant edges to be removed from the result of the watershed transformation." Usually, in spite of thresholding, the result is still over-segmented. Preprocessed images were transformed using a watershed segmentation filter. In order to reduce the number of resulting watersheds, the watersheds were merged based on their depth [5]. A basin floods its neighbour if the neighbour's depth does not exceed a given threshold called a merging level. This causes small basins to be merged with larger ones. As follows from the results presented below, the merging level is given by the percentage of the maximum watershed depth.

So far eighteen different color images have been used for testing the described approach to watershed merging. All the images were acquired using an sRGB camera and were processed in that color space. The images depict different types of objects. For the purpose of this paper one picture was selected for each object type. The following results were obtained for five selected images. As one can see from the figure description, the attribute set that allowed for obtaining the best quality segmentations is the watershed's average combined with the watershed's variance. This means that, for each watershed and each color channel, its average and variance were calculated and result in 6 attributes describing each watershed. Using only the watershed's average (3 attributes) usually gives good results too; however, adding variance prevents the clustering algorithm from leaving "stray" classes consisting of only a few watersheds. Such classes appear if the number of classes requested for the final segmentation is

Pobrane z czasopisma Annales AI- Informatica **http://ai.annales.umcs.pl**
Data: 26/11/2026 11:45:42

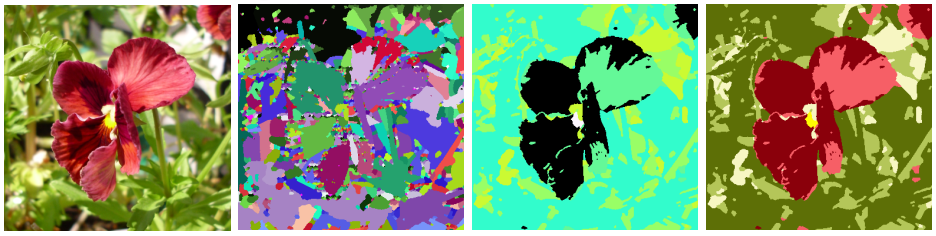
small (5 for example). The attribute set that was most unsuccessful was the watershed's average combined with its standard deviation. During testing usually the UPGMA and Ward's clustering methods gave good results (figures 5-8). The CLINK method can also be used successfully as shown in figure 4. For methods that require a similarity measure, the Euclidean distance was used.

The number of classes in the presented segmentations was chosen arbitrarily. The goal was to create a segmented image where all significant objects are still visible while selecting the smallest number of classes. When the proper clustering parameters are used, increasing the number of classes in the segmented image visibly increases the number of details present. Improper parameters cause small insignificant classes to appear.



a) b) c) d)

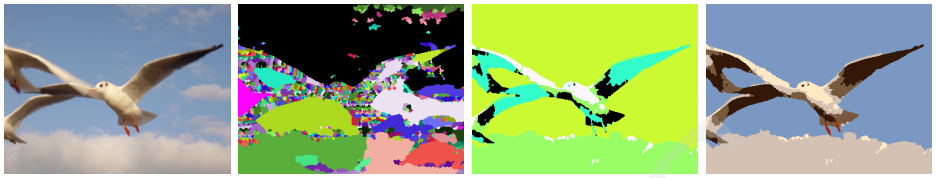
Fig. 4. a) Original sRGB image (budynek_1), b) over-segmented image (PCA-gradient, threshold 0%, merging level 9%) with 1873 watersheds, c) final segmentation (colored using a hashing scheme), d) final segmentation (colored using class' averages) (5 classes, RGB averages and RGB variances, Euclidean distance, CLINK method)



a) b) c) d)

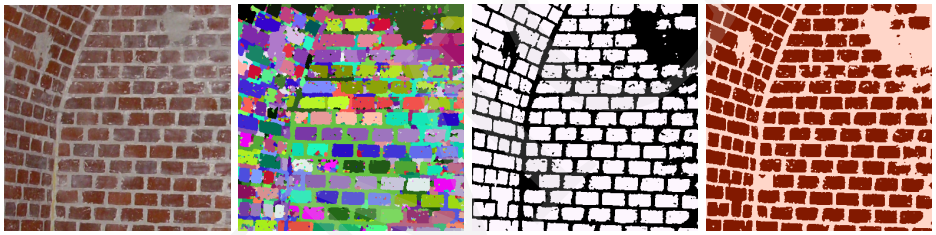
Fig. 5. a) Original sRGB image (kwiatek_3), b) over-segmented image (sum-of-squares-gradient, threshold 5%, merging level 10%) with 1965 watersheds, c) final segmentation (colored using a hashing scheme), d) final segmentation (colored using class' averages) (6 classes, RGB averages, RGB variances, Euclidean distance, UPGMA method)

Pobrane z czasopisma Annales AI- Informatica **http://ai.annales.umcs.pl**
Data: 26/11/2026 11:45:42



a) b) c) d)

Fig. 6. a) Original sRGB image (mewy_3), b) over-segmented image (PCA-gradient, threshold 0%, merging level 1%) with 1512 watersheds, c) final segmentation (colored using a hashing scheme), d) final segmentation (colored using class' averages) (6 classes, RGB averages, RGB variances, Euclidean distance, UPGMA method)



a) b) c) d)

Fig. 7. a) Original sRGB image (mur_1), b) over-segmented image (PCA-gradient, theshold 0%, merging level 8%) with 2555 watersheds, c) final segmentation (colored using a hashing scheme), d) final segmentation (colored using class' averages) (2 classes, RGB averages, RGB variances, Ward's method)



a) b) c) d)

Fig. 8. a) Original sRGB image (sufit_2), b) over-segmented image (sum-of-squares-gradient, threshold 5%, merging level 10%) with 2144 watersheds, c) final segmentation (colored using a hashing scheme), d) final segmentation (colored using class' averages) (4 classes, RGB averages, RGB variances, Euclidean distance, UPGMA method)

All test pictures presented in this paper were taken by Jakub Smołka and are free of charge available from:
http://pluton.pol.lublin.pl/~jsmolka/test_images/2008_ibiza/.

Pobrane z czasopisma Annales AI- Informatica **http://ai.annales.umcs.pl**
Data: 26/11/2026 11:45:42

Conclusions

Clustering methods can be used for eliminating over-segmentation not only in black and white medical images as shown in [9,10] but also in color images depicting different types of objects. This approach is useful when over-segmentation occurs in the entire image (figs. 4 and 5), in a certain class (Fig. 7 – mortar between the bricks, or Fig. 6 – the seagull) or in a certain region of an image (Fig. 8 – heavily over-segmented upper right corner of the image due to camera sensor noise). Preliminary tests, whose results are shown in this paper, lead to the conclusion that using even the simplest set of attributes – the RGB averages and a UPGMA or a Ward's method – allows for eliminating over-segmentation. The main advantage of this approach is that is does not disregard the available information about color whereas, in the case of plain watershed segmentation, this information is no longer taken into account once the gradient image is calculated.

References

[1] Beucher S., Lantuejoul C., *Use of watersheds in contour detection*. International Workshop on Image Processing: Real-time Edge and Motion Detection/Estimation, (1979).
[2] Gonzalez R. C., Woods R. E., *Digital Image Processing*. Addison-Wesley Publishing Company, (1993).
[3] Russ J. C., *The Image Processing Handbook*. CRC Press, (2002).
[4] Sapiro G., Ringach D. L., *Anisotropic Diffusion of Multivalued Images with Applications to Color Filtering*. IEEE Transactions on Image Processing, 11 (1996).
[5] Ibanez L., Schroeder W., Ng L., Cates J., *The ITK Software Guide*. Kitware Inc., (2003).
[6] ITK 3.4.0 library source code available at: http://www.itk.org
[7] Angulo J., Serra J., *Color Segmentation by Ordered Mergings*. IEEE Proceedings – International Conference on Image Processing, 2(3) (2003) 125.
[8] Pietro Perona, Jitendra Malik, *Scale-Space and Edge Detection Using Anisotropic Diffusion.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 12 (1990) 629.
[9] Smołka J., *Hierarchical Cluster Analysis Methods Applied to Image Segmentation by Watershed Merging*. Annales UMCS Informatica AI, 6 (2007) 73.
[10] Smołka J., Skublewska-Paszkowska M., *Comparison of Hierarchical Cluster Analysis Methods Applied to Image Segmentation by Watershed Merging.* Advances in Soft Computing 45 – Computer Recognition Systems 2, Springer-Verlag, (2007).
[11] Romesburg Ch., *Cluster Analysis for Researchers.* Lulu Press, (2004).
[12] Everitt B. S., Landau S., Leese M., *Cluster Analysis.* Arnold a member of the Hodder Headline Group, (2001).