GEOGRAPHIC IMAGE MINING

Alexandru BOICEA[1], Florin RADULESCU[1], Ciprian-Octavian TRUICA[1], Irina VASILE[1]

DOI: 10.21163/GT_2016.112.01

ABSTRACT:

Image segmentation is a branch of the image processing domain which involves partitioning the image into multiple segments referred to as sets of pixels. The segmentation purpose is to change the representation of an image into a form that is analyzed easier and less processed in order to extract information based on the colors in the image. This paper shows how we can determine statistics on the relief by processing the images that represent geographic maps through Data Mining algorithms. The important aspects of this type of mining are also described in this study. A dedicated application is used for data processing.

Key-words: *Image processing, Image segmentation, Data mining, Clustering methods, Image color analysis*

1. INTRODUCTION

Image segmentation is usually used to extract different forms, objects, clusters, sets of pixels, shapes, and so on from an image. It is also a branch of image processing that deals with extracting important information from images based on colors (Zhang, 1996). Processing involves assigning an identifier to each set of pixels, in order to be used in the processing of statistics relating to an image or set of images (Haralick & Shapiro, 1985). This paper presents image processing techniques for relief maps. This type of processing has been less exploited over time, unlike the similarity between images, object detection, or face identification.

Two criteria must be addressed when developing a statistical system for relief maps:

- Graphics oriented system, which provides methods of performing graphics that process the data from multiple sources, from which statistics are extracted;
- A system that uses the best algorithm for processing the images used as input data, taken from more sources.

Some of traditional database processing techniques are being replaced with data mining techniques dedicated to various applications (Mocanu et al, 2014, Arrais de Freitas et al, 2016). Image mining is based on standard data mining methods (Thamilselvan & Sathiaseelan, 2015). The textual descriptions are not used for image mining and processing. The K-means algorithm is used for image segmentation and clustering (Dhanachandra, Manglem & Chanu, 2015).

Some problems can be solved by simple K-means clustering, while other situations may require more complex algorithms with larger memory or time requirements (Naik & Shah, 2014). Fuzzy C-means (FCM) and Genetic Algorithms were characteristics of the acquired image, resolution characteristics of the acquired image, resolution limitations and the imperfections induced by the process of image acquisition (Awad et al., 2009). Another image segmentation technique is based on the non-parametric clustering procedure in the

[1] *University Politehnica of Bucharest, Faculty of Automatic Control and Computers, 060042,Bucharest, Romania, alexandru.boicea@cs.pub.ro, florin.radulescu@cs.pub.ro, ciprian.truica@cs.pub.ro, irina.vasile@cti.pub.ro*

discretized color space using the discrete probability density function (Krstinić, Skelin & Slapnicar 2011). Using images from the Internet, additional information is often available, hence retrieval the new image processing methods based on the contents of images are necessary (Lee & Nang, 2011).

Geographic image processing can have many utilities, eg if processes images from flooded areas we can generate real-time warnings about the dangers posed to local resident or tourists in the area (Magyari-Saska, 2014) and finding access routes for medical emergencies(Nicoara & Haidu, 2014). Another utility is monitoring deforestation and afforestation in mountain areas (Costea & Haidu, 2010).

2. IMPLEMENTATION METHOD

Implementation and testing are done with a software application that, besides processing relief maps, allows storing and retrieving statistical results after image processing.

The application provides the following services:

- Uploading images representing relief map, taken from the user's personal files or downloaded via the geospatial mapping platform: Microsoft Bing Maps.
- Image segmentation by applying the clustering algorithm K-means (Hartigan & Wong, 1979) with the number of clusters chosen by the user;
- Inserting processed information in a data structure for later use in constructing graphics;
- Generating statistics and graphics;
- Exporting results into MS Word compatible files.

The software application is based on image mining, more exactly image segmentation using K-means clustering algorithm. It demonstrates a good applicability of this algorithm for obtaining statistical information for the relief elements on a relief map. The software provides a mechanism to correct the imprecise results (discovered clusters). It is well documented that image mining processing is an area where the user intervention is needed for precise results, because for now we are not working with textual information (characters or numerical). In image Mining we are working with images that are structured like a pixel matrix. The application is developed in C# using the .NET Framework 3.5. The graphical user interface is developed using Windows Application Forms, and Microsoft Visual Studio 2010 is used as integrated development environment (IDE).

The information resulted from maps processing is stored in an XML file. The in-memory programming interface LINQ-to-XML is used for parsing the XML file. An integrated specialized external control (Earth control) is used to query the Microsoft Bing Maps service. Microsoft.Office.Interop.Word namespace is used to export the data in a Word file and System.Windows.Forms. DataVisualization. Charting namespace is used to generate graphics.

3. K-MEANS ALGORITHM

K-means algorithm is a clustering algorithm that groups N points from a finite space into K clusters based on the distance between these points and the cluster centers. The use of this algorithm tries to find clusters in the image, where a cluster contains pixels with similar colors.

In K-means, each data point x_m belonging to real *n*-dimensional space R^n, is assigned to its nearest centroid y_p (a cluster weight center), for minimizing the mean squared distance from each data point to its nearest centroid (Kanungo et al, 2002).

The use of the Squared Euclidian Distance places progressively greater weight on objects that are further apart:

$$d^2(x_m, y_p) = \sum_{i=1}^{n} (x_m^{(i)}, y_p^{(i)})^2 \tag{1}$$

The Squared Euclidean Distance is not a distance function as it does not satisfy the triangle inequality, however it is frequently used in optimization problems in which distances only have to be compared. Therefore, the Euclidean distance is used as the distortion measure (Chang, Lai & Jeng, 2011). From the scientific and mathematical point of view, distance is defined as a quantitative degree of how far apart two objects are (Cka, 2007). Another method for generating the cluster center without incrementing the execution time is to reduce the mean square error of the final cluster (Purohit & Joshi, 2013).

The implementation of the algorithm consists of the following steps:

- Choose K centroids, either randomly or based on some heuristics, in the image-vector space (our approach is based on the random generation) (Patil & Jondhale, 2010; Khan & Ahmad, 2013).
- Store the values of RGB (Red, Green, Blue) for each of the chosen random point in a data structure (matrix).
- Scroll the image (RGB array points) and calculate for each point the Euclidean distance between that point and each centroid. The point is assigned to the cluster with the nearest centroid.
- Re-compute the centroids for the next iteration as the mean arithmetic RGB value levels of the image points in cluster.
- If RGB values of each new centroid and the initial centroids are similar, then the algorithm ends (calculated value is compared with a predefined threshold value which is equal to 0.03), otherwise the algorithm is repeated for the centroids discovered in the current step.

The K-Means class contains the following fields:

- Centroids
- Clusters, specify number of clusters
- Distance, the metric function used to discover the clusters in the algorithm implementation (initialized through the class constructor)
- Compute process that runs the algorithm

Fig. 1 shows the class diagram:

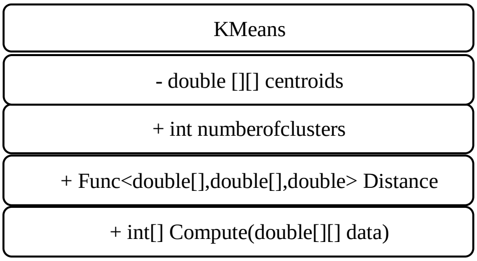

KMeans
- double [][] centroids
+ int numberofclusters
+ Func<double[],double[],double> Distance
+ int[] Compute(double[][] data)

Fig. 1 KMeans class diagram

```
public int[ ] Compute(double[ ][ ] imageData)
{    int rowsNumber = imageData.Length;
     int columnsNumber = imageData[0].Length;
// Choose a number of distinct indexes that are in range0:n-1
     int[ ] indexes = Utils.Random(rowsNumber, k);
// Set the centroids
     actualCentroids =imageData.Submatrix(indexes;
     int[ ] count = new int[k];
     int[ ] tickets = new int[rowsNumber];
     double[ ][ ] computeCentroids;
       while(true)
        {
 //Reset the centroids and the counters
       computeCentroids = new double[k][ ];
       for (int i=0; i < k; i++)
          {
            computeCentroids[i] =
                  new  double[columnsNumber];
            count[i] = 0;
          }
//The pixels from the imagedata will be //accumulated into the
nearest clusters loop for //each point
         for (int j=0; j<imageData.Length; j++)
          {
 // Store the initial information
          double[] point = imageData[j];
// Compute the nearest cluster centroid
            int cluster_value = tickets[j] =
               Utils.ComputeNearest(imageData[j]);
          count[cluster_value]++;
          double[ ] centroid =
               computeCentroids[cluster_value];
          for (int m=0; m<centroid.Length; m++)
           centroid[m] += point[m];
            }
//In the above section it is computed the new //centroid value for
every cluster
         for (int i = 0; i < k; i++)
            {
           double[ ] averageValue =
               computeCentroids[i];
           double clusterNumber = count[i];
           for (int j=0; j<columnsNumber; j++)
             averageValue[j] /= clusterNumber;
             }
 //Stop condition is related to the threshold
    if (centroids.IsEqual(computeCentroids,threshold))
            break;
 // Next step
            centroids = computeCentroids; }
```

The algorithm is implemented by the Compute function of class K-means and returns a data structure that represents an array of points and correspondence with the corresponding clusters. In the following is a fragment of the function implementing the algorithm:

Fig. 2 shows the progressive transformation of clusters by applying the algorithm K-Means. The algorithm converges after five iterations from the left to the right (Mirkes, 2016).

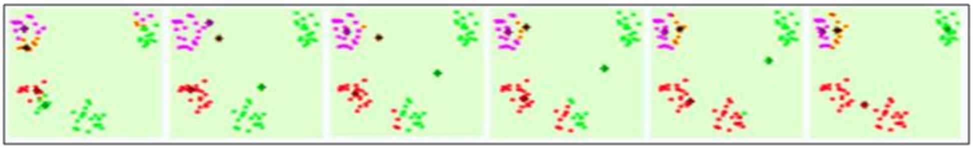


Fig. 2 Progressive transformation of clusters

4. APPLICATION DESCRIPTION

The application is structured into five interconnected modules:

- Image Selection Module. Images that are processed can be selected from the user's home directory or can be captured with Microsoft Bing Maps platform.
- Image Segmentation Module. It is the module that processes the images selected by the user, using K-means algorithm.
- Data Storage Module. With this module, the processed data is stored within the ImageStore in order to be used at making graphs.
- Graphics Module. With this module, graphics are being generated based on the data from ImageStore.
- Export Results Module. This module is used to export processing results in a Word file.

Fig. 3 shows the application architecture

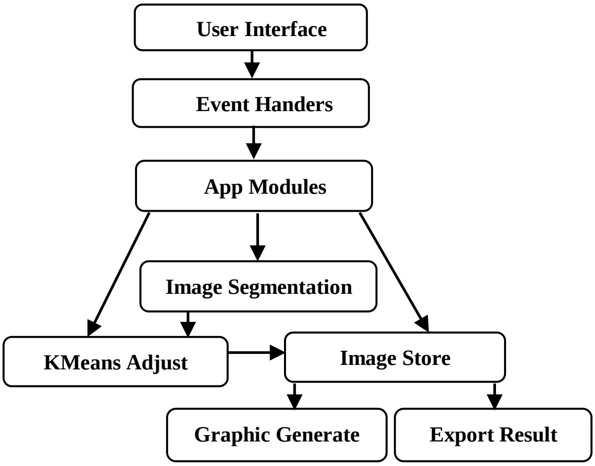


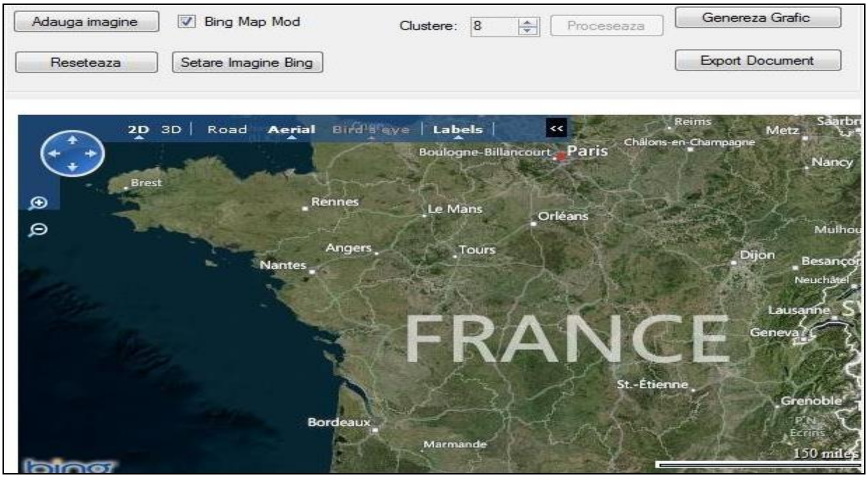
Fig. 3 Application architecture

4.1 Image Selection Module

The application has been tested and analyzed for several types of images, considered as input. This module provides two ways of working:

- Capturing images with an embedded controller, using Microsoft Bing Maps mapping service.
- Images can be selected from the user’s directory system;

Fig. 4 shows an image selection screen using the Bing Maps platform, which is a view service of geographical areas around the world. Bing Maps is integrated with Microsoft’s comprehensive developer tools such as Visual Studio and other products and services such as Windows, Azure and Office.

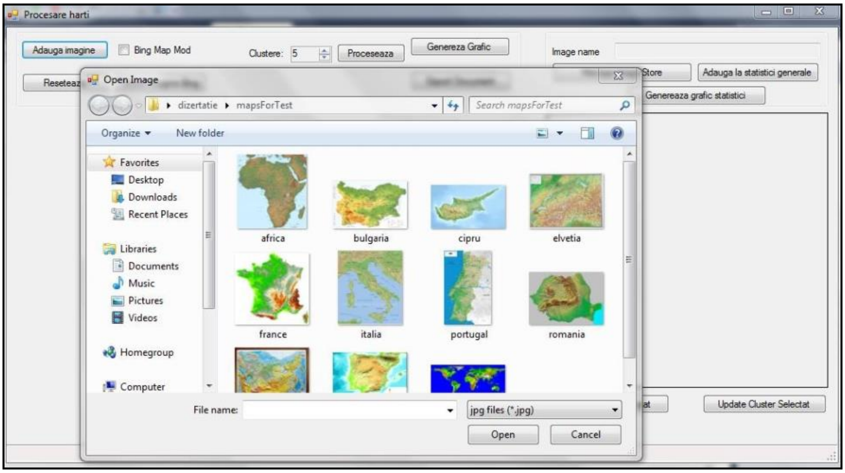


☑ Bing Map Mod

Clustere: 8

Fig. 4 Selecting images from Bing Maps platform

Fig. 5 shows the selection screen of images from the user's folders. Once the user has selected a source image (image captured with Microsoft Bing Maps or a local one), the K-means algorithm is applied to process the image set in the PictureBox control.



□ Bing Map Mod

Clustere: 5

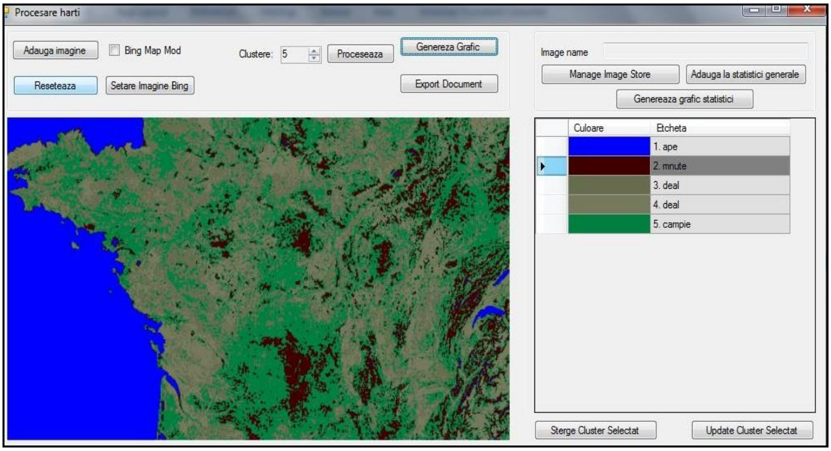
File name:

jpg files (*.jpg)

Fig. 5 Selection screen of images from the user's folders

The algorithm is applied to an array of points (pixels which form the image). The specific RGB values are associated to each point. The vector point is obtained using a method of the BitMap class on the image loaded in PictureBox control. The number of clusters K is chosen by the user and represents the number of collection points that are determined by the algorithm. The success of K-means image segmentation depends very much on the number of clusters. For a correct estimation, the short edges are considered minimal details that are normally imposed of the texture of objects, while long edges are used to estimate the number of clusters into image (Chang, Lai & Jeng 2011).

The results of clustering process are seen in **Fig. 6**.



Culoare	Etcheta
🟦	1. ape
🟫	2. mnute
▨	3. deal
▨	4. deal
🟩	5. campie

Fig. 6 Results of the clustering process

After identifying the pixels that make up a cluster, will be assigned a label and color of each cluster. The label may be modified by the user, in order to associate one description for each color in the cluster. Moreover, the user can change the colors set after image processing, if they have not been clearly determined. Because the subject of the processing is represented by an image, not text, the image processing allows any user to refine the results obtained. The user has also the possibility to delete a cluster, if it is redundant, because it has a small number of pixels in the image.

4.2 Data Storage Module

Information about the processed images will be saved in the database in order to generate statistics. This process is also known as image annotation and it is useful for information retrieval (Mocanu, 2010; Paul & Beegom, 2010; Russell et al, 2008). ImageStore is the database application where information is saved after images processing. Since storing images in binary form in a database is a less efficient approach, the method of storing the processed images in the user’s directory files is used. ImageStore is the repository of images that have been processed and, for performance reasons, this is a XML file.

ImageStore contains the following components:

- The path name of the original image saved to a folder on the user's computer;
- The number of clusters used in K-means clustering algorithm;
- Specific elements of a cluster.

XML file has the following structure:

- Element: image
- Attribute: path, name
- Children: clusters
  - Element: cluster
  - Attribute: label
  - Children: color(Element), scale(Element)
    - Attribute: r, g, b (RGB model color)

Below is a sequence of XML code for ImageStore. Writing and reading the XML file is done using LINQ-to-XML.

4.3 Graphics Module

This module generates graphics for a single image (image currently displayed in the PictureBox) or multiple images stored in ImageStore. The graph has the following sources of data for X and Y axes:

- X: Labels for clusters;
- Y: The percentage of pixels in the image belonging to the cluster.

```
private void buttonAddGeneralStatistics_Click(object sender,
EventArgs e)
{  this.Enabled = false;
string rootPath =   ConfigurationSetting.AppSettings[“rootPath”];
if (initialImage == null || colorsScale == null)
return;
XElement first = XElement.Load(rootPath +     “image_store.xml”);
string initialImagePath = rootPath + @”images\”  +
Guid.NewGuid().ToString() + “.jpeg”;
initialImage.Save(initialImagePath);
XElement image = new XElement(“image”,
  new  XAttribute(“path”, initialImagePath),
  new XAttribute(“name”,      textBoxImageName.Text));
XElement clusters = new XElement (“clusters”);
      foreach (KeyValuePair<Color, int> pairscale in  colorsScale)
  {   clusters.Add(new XElement(“cluster”, new
XAttribute(“label”,getLabelColor(pairscale.Key)),
  new XElement(“color”,
  new XAttribute(“r”, pairscale.Key.R),
  new XAttribute(“g”, pairscale.Key.G),
  new XAttribute(“b”, pairscale.Key.B),
  new XElement(“scale”, pairscale.Value)));
  }
```

In **Fig. 7**, the graphic obtained through image processing is displayed. Every relief form has a color correspondence:

1. Water (ocean, sea, river); 2. Mountain; 3. Hill; 4. Plateau; 5. Plain;

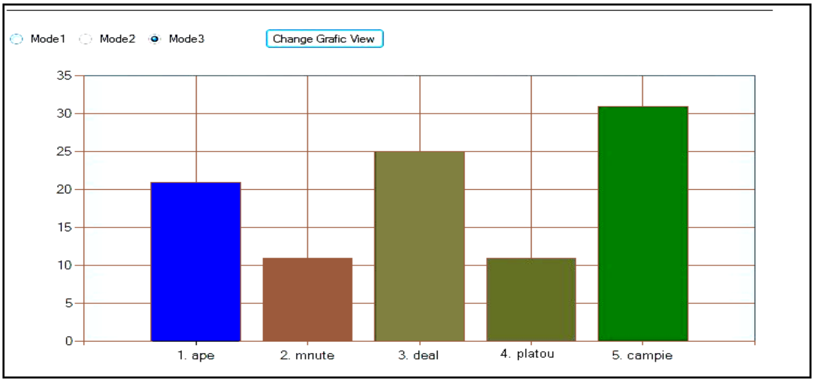


Fig. 7 The graphic afferent processed image

4.4 Export Result Module

The user can export the information into a Word document. To export the results into a Word file, Microsoft.Office.Interop. Word namespace is used.

5. EXPERIMENTAL RESULTS

We present an example for testing the algorithm starting from the initial image in **Fig. 8.a**, taken from a user's data directory. The image is segmented into 6 clusters of different colors, gray is the background of the image and each of the other colors is a form of relief.

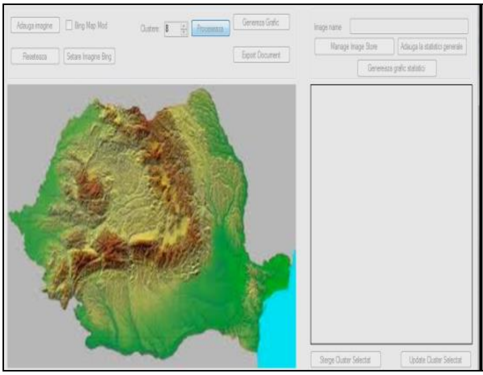
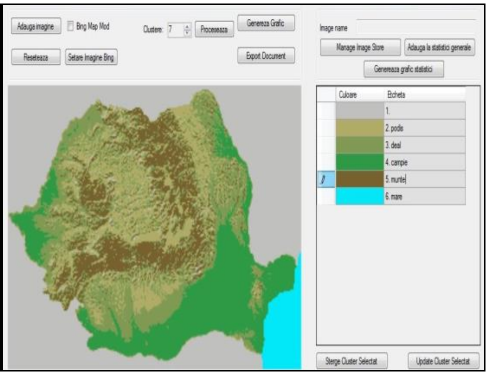


Fig. 8 a. Image before applying K-means algorithm



Culoare	Eticheta
	1.
	2. podis
	3. deal
	4. campie
	5. munte
	6. mare

Fig. 8 b. Image after applying K-means algorithm

The graphic from the image is generated based on K-means cluster results (color frequency and label color associated by the user, using the GUI interface). The interface for selecting colors associated of the clusters is shown in **Fig. 8.b**.

The graphic from **Fig. 9** shows the share of each relief form, calculated by applying clustering algorithm. Relief forms and color correspondences are:

1. Image background; 2. Plateau; 3. Hill; 4. Plain; 5. Mountain; 6. Black Sea.

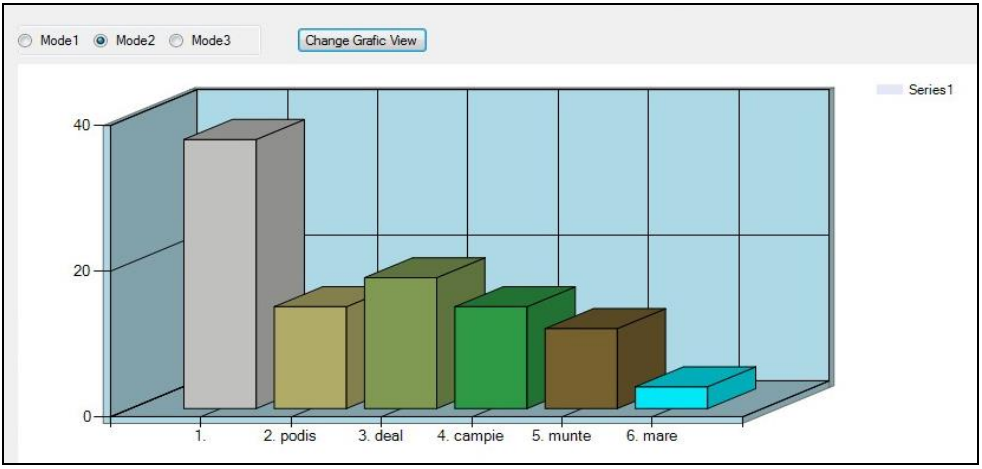
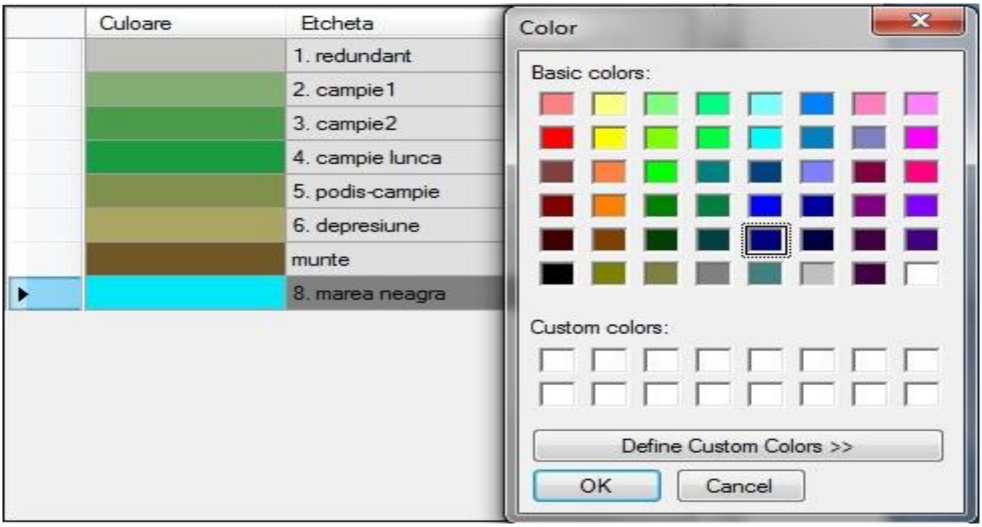


Fig. 9 Statistics by cluster types

The colors can be changed using the interface shown in **Fig. 10**.

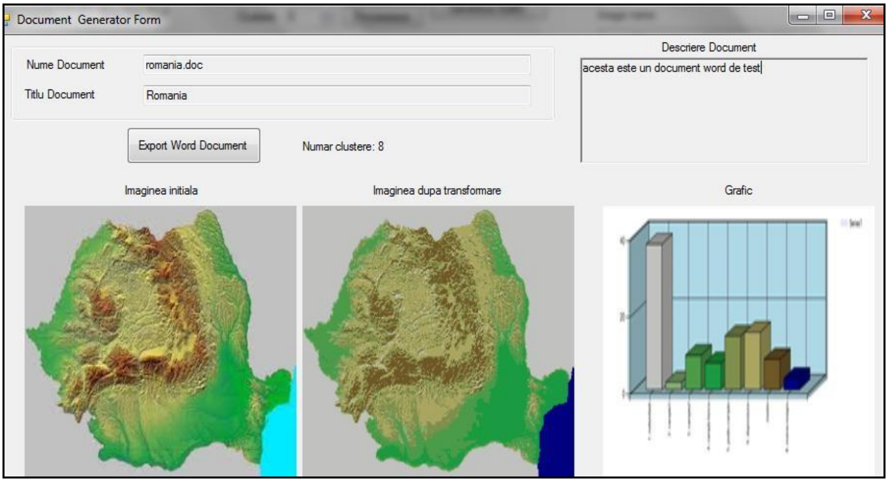


	Culoare	Etcheta
	⬜	1. redundant
	🟩	2. campie1
	🟩	3. campie2
	🟩	4. campie lunca
	🟩	5. podis-campie
	🟨	6. depresiune
	🟫	munte
▶	🟦	8. marea neagra

Fig. 10 Color selection interface for clusters

Generating the word document is done with the interface shown in **Fig.11**.

Through ImageStore, other kind of statistic can be built using the data generated from more than one map.



Document Generator Form

Nume Document: romania.doc

Titlu Document: Romania

Descriere Document: acesta este un document word de test

Numar clustere: 8

Fig. 11 Generating results in a Word file

Fig. **12** shows a collection of images that represents maps with forms of relief for Romania, Portugal and Italy.

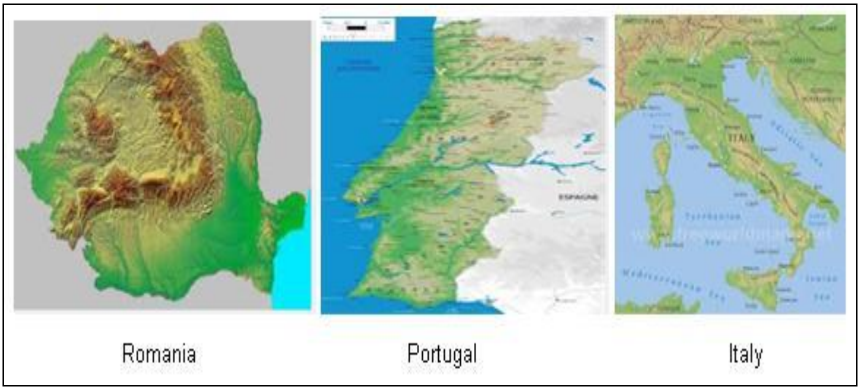
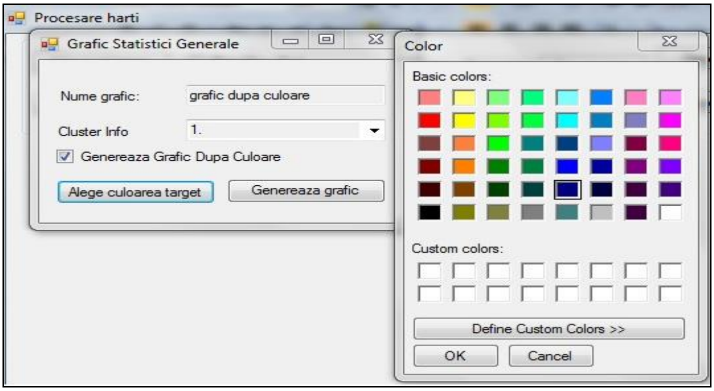


Fig. 12 Relief map collection

Fig. 13 presents the interface for building statistics from more than one map. In this case, the statistics can be generated based on the cluster color or cluster label (Y axis), for more countries (X axis).



Nume grafic: grafic dupa culoare

Cluster Info 1.

☑ Genereaza Grafic Dupa Culoare

Fig. 13 Interface for generating the map collection statistics

Fig. **14.a** shows the statistics based on cluster color for sea/ocean and **Fig. 14.b** shows the statistics based on cluster label for plain relief. The graphs are generated using the data obtained by maps processing and stored in ImageStore.

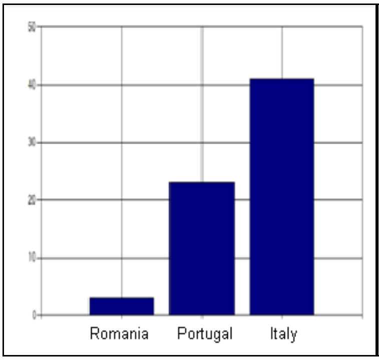


Fig. 14 a. Statistics by cluster color

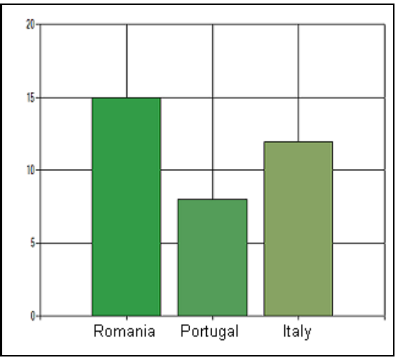


Fig. 14 b. Statistics by cluster label

Before calculating the statistics, images must be processed in order to have the same number of clusters associated with the same colors, labels and meanings.

6. CONCLUSIONS

Image processing is a complex technique with many applications, e.g. object recognition, data mining, image segmentation, etc. Nowadays, image processing is an area of high importance in various sectors. In the systems based on images, the content of an image can be perceived in terms of different features such as color, texture or shape.

In this paper an application for image segmentation into clusters by applying data mining algorithms is presented.

The application implements the K-means clustering algorithm for map relief image processing and segmentation based on the colors used for each form of relief. The application is tested on image from the geographic domain, but it can also be used for processing images from other domains such as climatology, medicine, biology, etc.

The tests are performed for a series of images which represent maps with different relief forms (images with a variable number of colors). The K-means algorithm is an iterative mining technique that is used to partition an image into K clusters, based on pixel color, intensity, texture, location or a weighted combination of these characteristics.

K-means clustering algorithm is applied over the pixels of the input image and, as a result, the clustered image and the specific statistics are obtained. The distance between the points is based on the pixel color and parameter K. The K parameter can be selected manually, randomly or by a heuristic.

The algorithm allows image segmentation, but the results depend greatly on the number of the initial colors and on number of clusters who was chosen. Choosing a very small number of clusters leads to obtaining inconclusive results. On the other hand, if too many clusters are chosen, an overly large segmentation is obtained and so the results are more difficult to interpret.

By applying this technique to image processing, the area of utility can be extended, for example, on the similarity of images based on shades of color.

REFERENCES

Arrais de Freitas, N. C. , Reboucas Filho, P. P., Gurgel de Moura, C. D., & Pereira dos Santos Silva, M. (2016) AgentGeo: Multi-Agent System of Satellite Images Mining, *IEEE Latin America Transactions*, 14(3), 1343-1351, doi: 10.1109/TLA.2016.7459619;

Awad, M., Chehdi, K., & Nasri, A. (2009) Multi-component image segmentation using a hybrid dynamic genetic algorithm and fuzzy C-means, *IET Image Processing journal*, 3(2), 52-62, doi: 10.1049/iet-ipr.2007.0213;

Chang, C. T., Lai, J. Z. C., & Jeng, M. D. (2011) A fuzzy k-means clustering algorithm using cluster center displacement, *Journal of Information Science and Engineering*, 27(3), 995-1009;

Cka, S. K. (2007) Comprehensive Survey on Distance/Similarity Measures between Probability Density Functions, *International Journal of Mathematical models and methods in applied sciences*, 1, 300-307;

Costea, G., & Haidu, I. (2010) Detection of Recent Spatial Changes Regarding Landuse in Small Basins From the Apuseni Natural Park, *Geographia Technica*, 12, 11-17;

Dhanachandra, N., Manglem, K., & Chanu Y. J. (2015*)*, Image segmentation using k-means clustering algorithm and subtractive clustering algorithm, *Procedia Computer Science*, 54, 764-771, doi:10.1016/j.procs.2015.06.090

Haralick, R. M., & Shapiro, L. G. (1985) Image segmentation techniques, *Computer Vision, Graphics, and Image Processing*, 29(1), 100-132, doi: 10.1016/S0734-189X(85)90153-7

Hartigan, J. A., & Wong, M. A. (1979) Algorithm as 136: A k-means clustering algorithm, *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1), 100–108

Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R., & Wu, A. Y. (2002) An efficient k-means clustering algorithm: analysis and implementation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7), 881-892, doi: 10.1109/TPAMI.2002.1017616

Khan, S. S., & Ahmad, A. (2013) Cluster centre initialization algorithm for k-means cluster, *Expert Systems with Applications*, 40(18), 7444-7456, doi: 10.1016/j.eswa.2013.07.002

Krstinić, D., Skelin, A. K., & Slapnicar, I. (2011), Fast two-step histogram-based image segmentation, *IET Image Processing*, 5(1), 63-72, doi: 10.1049/iet-ipr.2009.0107

Lee, J., & Nang, J. (2011) Content-based image retrieval method using the relative location of multiple ROIs, *Advances in Electrical and Computer Engineering*, 11(3), 85-90, doi:10.4316/AECE.2011.03014

Magyari-Saska, Z. (2014) Quantifying Threats Along Tourist Trails: An Initial Approach, *Geographia Technica*, 9, Issue 1, 78-86

Mirkes, E. M. (2016) K-means and K-medoids applet. University of Leicester. Retrieved 15 September 2016. [Online] Available at:

http://www.math.le.ac.uk/people/ag153/homepage/KmeansKmedoids/Kmeans_Kmedoids.html

Mocanu, S. , Din, R. , Saru, D., & Popa, C. (2014) Using graphics processing units for accelerated information retrieval, *Studies in Informatics and Control*, 23(3), 249-256

Mocanu, I. (2010) From content-based image retrieval by Shape to Image Annotation, *Advances in Electrical and Computer Engineering*, 10(4), 49-56, doi:10.4316/AECE.2010.04008

Naik, D., & Shah, P. (2014) A review on image segmentation clustering algorithms, *International Journal of Computer Science and Information Technologies*, 5(3), 3289-3293

Nicoara, P. S., & Haidu, I. (2014) A Gis Based Network Analysis For The Identification Of Shortest Route Access To Emergency Medical Facilities, *Geographia Technica*, 9, Issue 2, 60-67

Patil, R. V., & Jondhale, K. C. (2010) Edge based technique to estimate number of clusters in k-means color image segmentation, *IEEE International Conference on Computer Science and Information Technology (ICCSIT)*, 117-121, doi: 10.1109/ICCSIT.2010.5563647

Paul, E., & Beegom, A. (2015) Mining images for image annotation using SURF detection technique, *International Conference on Control Communication & Computing India (ICCC)*, 724-728, doi: 10.1109/ICCC.2015.7432989

Purohit, P., & Joshi, R. (2013) A new efficient approach towards k-means clustering algorithm, *International Journal of Computer Applications*, 65(11), 7-10

Russell, B. C., Torralba, A., Murphy, K. P., & Freeman, W. T. (2008) LabelMe: A Database and Web-Based Tool for Image Annotation, *International Journal of Computer Vision*, 77(1), 157-173, doi: 10.1007/s11263-007-0090-8

Thamilselvan, P., & Sathiaseelan, J. G. R. (2015) Image classification using hybrid data mining algorithms - a review, *International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*, 1-6, doi: 10.1109/ICIIECS.2015.7192922

Zhang, Y. J. (1996) A survey on evaluation methods for image segmentation, *Pattern Recognition*, 29(8), 1335-1346, doi: 10.1016/0031-3203(95)00169-7