DOI: 10.15439/2015F222
ACSIS, Vol. 5

Applying fuzzy clustering method to color image segmentation

Omer Sakarya
University of Gdańsk
Institute of Informatics
osakarya@inf.ug.edu.pl

***Abstract*—The goal of this paper was to apply fuzzy clustering algorithm known as Fuzzy C-Means to color image segmentation, which is an important problem in pattern recognition and computer vision. For computational experiments, serial and parallel versions were implemented. Both were tested using various parameters and random number generator seeds. Various distance measures were used: Euclidean, Manhattan metrics and two versions of Gower coefficient similarity measure. The F and Q segmentation evaluation measures and output images were used to assess the result of color segmentation. Serial and parallel run times were compared.**

I. Introduction

COLOR image segmentation is a method of assigning pixels of given image to segments which share similar color. Pixels from a segment should be similar colorwise and pixels from different segments should be distinct. The problem of color image segmentation is one of the most difficult problems in computer vision. There exist many algorithms for this particular problem, however none of them work well for all kinds of images. Photos of real world are very different in colors, shapes and noise. Usually before choosing an algorithm for color image segmentation, domain knowledge is used to assess the type of algorithm needed for particular set of photos. The goal of color image segmentation research is to find an universal algorithm that would not require domain knowledge prior use and would provide good results for all kinds of photos. Color image segmentation is an important part of various computer vision problems, including pattern recognition. It is a step performed before pattern recognition, so if the color segmentation is poor, the pattern recognition step may fail. The aim of this paper was to apply fuzzy clustering algorithm known as Fuzzy C-Means to color image segmentation with intention of developing a general method for various types of images.

The paper is organized as follows. Section II introduces color image segmentation problem. In section III distance measures and fuzzy clustering method are described. In section IV the implementation details are presented. Section V presents a possible way of source code parallelization for speed-up. Section VI overviews the computational experiments and achieved results. The last section contains conclusions and plans for further research.

II. Color image segmentation

Formally image segmentation can be defined as follows [1]: If $P()$ is a homogeneity predicate defined on groups of connected pixels, then segmentation is a partition of the set F into connected subsets or regions $(S_1, S_2, \ldots, S_n)$ such that:

$$\bigcup_{i=1}^{n} S_i = F \wedge \forall_{i \neq j} S_i \cap S_j = \phi \wedge \forall_i P(S_i) = true \wedge \forall_{i \neq j} P(S_i \cup S_j) = false \tag{1}$$

The two most important problems in color image segmentation is choosing the proper algorithm for given type of images and choosing the right colorspace. There are various color representations used in color image segmentation, however none of them is perfect for all kinds of images [1]. In computational experiments the RGB color space was used (see section IV).

III. Fuzzy clustering

In data clustering, elements from data set are divided into clusters where elements in the same cluster are similar and elements from different clusters are not. There are many similarity and distance measures $dist(x, y)$ that can be used in combination with a clustering algorithm. Example distance measures are Euclidean (2) and Manhattan (3) metrics:

$$d_e(x, y) = \sqrt{(y_1 - x_1)^2 + \ldots + (y_n - x_n)^2} \tag{2}$$

$$d_m(x, y) = \sum_{k=1}^{n} |x_k - y_k| \tag{3}$$

Fuzzy clustering is one of the possible approaches to clustering. As opposed to hard clustering where data element x belongs exclusively to one cluster, in fuzzy clustering element x belongs to every cluster to some degree.

Fuzzy C-Means (FCM) is one of the fuzzy clustering algorithms that can be used for color image segmentation. It is an iterative algorithm that can make use of various similarity and distance measures. The FCM algorithm assigns membership values to each data element, which are inversely related to the relative distance of an element to the centroids. In FCM, the closeness of each data x_k to the center of a cluster v_i (centroid) is defined as the membership (u_{ik}) of x_k to the i-th cluster of data set minimizing the following objective function [2]:

$$J_m(U, V) = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^{m} dist(x_k, v_i)^2 \tag{4}$$

where $X = \{x_1, \ldots, x_N\}$ a given set of unlabeled N data; $V = \{v_1, \ldots, v_c\}$ are the cluster centers and $m = [1, \infty]$

is the weighting exponent which determines the fuzziness of the resulting clusters, $U = [u_{ik}]$ matrix c x n, where u_{ik} is membership of x_k to the i-th cluster $\sum_{i=1}^{c} u_{ik} = 1, \forall k = 1, 2, \ldots, n$. The cluster centers and the memberships are computed as:

$$v_i = \frac{\sum_{k=1}^{n} u_{ik}^{m} x_k}{\sum_{k=1}^{n} u_{ik}^{m}} \tag{5}$$

$$u_{ik} = 1 / \sum_{j=1}^{c} \left(\frac{dist(x_k, v_i)}{dist(x_k, v_j)} \right)^{2/(m-1)} \tag{6}$$

Algorithm 1 presents the Fuzzy C-Means method. The $initU$ function randomly initializes the membership matrix U and the parameter $maxItNum$ specifies the maximum number of iterations of the algorithm. U' stores the previous values of U and is used in line 8 to check if the solution found so far has converged according to some ε and the algorithm should stop, $dist()$ is one of the distance measures. As a result of the algorithm membership matrix U and centroids V are computed. In (6) it is possible that the denominator will be equal to 0, such situation should be considered in computer implementation of the algorithm to avoid errors.

Algorithm 1 Fuzzy C-Means

```
Require: $X$ - data set, $c$ - number of clusters, $dist$ - distance
    measure, $maxItNum$ - maximum number of iterations
Ensure: $U$ - membership matrix, $V$ - centroids
 1: $U \Leftarrow initU$
 2: $i \Leftarrow 0$
 3: while $i < maxItNum$ do
 4:     compute new centroids $V$ using (5)
 5:     $U' \Leftarrow U$
 6:     compute new membership matrix $U$ using (6) and $dist$
 7:     $i \Leftarrow i + 1$
 8:     if $max|U - U'| < \varepsilon$ then
 9:         break
10:     end if
11: end while
```

IV. Applying FCM to color image segmentation

Images in a computer are stored in various formats, however many of them use the RGB color space. This means that each pixel is represented by three numbers being the red, green and blue components. Such triples can be treated as vectors, which means that it is is possible to use the FCM algorithm to cluster such data. The advantage of using fuzzy clustering for color image segmentation is that as a result we obtain a membership matrix which may be used to generate more than one segmented image. This can be achieved by choosing for each pixel which one of the membership values we want to use as the one defining the final color of a pixel in segmented image.

To perform the computational experiments for this paper the FCM algorithm was implemented in the C programming language on GNU/Linux operating system. The portable pixmap image format was used for simplicity, because it only contains a small header and the following values represent RGB triples. Binary method of randomly initializing the membership matrix U was used, where each column has value 1 in one of the rows, and the rest contains 0. The output membership matrix U and centroid values V were used to generate the segmented image file. In U, the membership information defines to which segments to what degree given pixel belongs. The centroids V contain colors of the segments. For each column of U the maximum membership value was selected, then the index of this value was used to assign color from the set of centroids V to a pixel. Algorithm 2 presents the segmentation method.

Algorithm 2 Segmentation

```
Require: $pixels$ - array of RGB triples representing pixels
    of original image, $N$ - number of pixels, $c$ - number of
    clusters, $U$ - membership matrix, $V$ - centroids
Ensure: $pixels$ - segmented image
1: for $k \Leftarrow 1; k \leq N; k \Leftarrow k + 1$ do
2:     $m \Leftarrow 0$
3:     for $i \Leftarrow 1; i \leq c; i \Leftarrow i + 1$ do
4:         if $u_{ik} > u_{mk}$ then
5:             $m \Leftarrow i$
6:         end if
7:     end for
8:     $pixels_k \Leftarrow V_m$
9: end for
```

V. Parallelizing source code using OpenMP

Serial source code was parallelized using OpenMP. Independent 'for loop' iterations were identified and OpenMP pragmas were used, which resulted in speed-up caused by parallel computation. This solution is automatically scalable, which means that when the same program is executed on a CPU with more computing cores, the program will use all of them automatically and execute faster. The experiments for this paper were performed on a laptop with Intel Core 2 Duo processor.

Although the main loop in FCM algorithm is not independent, it was possible to parallelize centroids vector V and membership matrix U computation in each iteration. Each centroid and membership can be computed independently from others, there is no data race condition. In the implementation, for research purposes, also the computation of J_m objective function and square error criterion were parallelized. It is important to note that there was no need to modify the algorithm, only the source code was parallelized using few OpenMP pragmas.

VI. Results overview

Two versions of the program were used - serial and parallel. The difference between them was that the parallel version used OpenMP pragmas. Both versions were compiled from the same source code, where the serial version had disabled pragmas. The run time was measured for both programs executed with the same parameters, however it is the wall-clock time of the whole program, which not only computes the FCM function, but also reads, writes files and computes J_m, square error criterion and more. The programs would execute even faster if the additional operations and computations were removed. The time measurement is just a very general information of how OpenMP pragmas influenced the run time of a parallel program in comparison to serial program.

The experiment was performed using different parameters on the same image (photo), it was 402 pixels wide and 600 pixels high. Parameters that could be specified for the program were: photo file, number of clusters, maximum iterations number, distance measure and random number generator seed. The available distance measures were Euclidean, Manhattan metrics and two versions of Gower coefficient [3] - regular (7,9) and modified (8,9). The modified version that takes specifics of RGB vector data into account was prepared for experiments in this paper.

$$S_i(x_i, y_i) = \begin{cases} 1, & \text{if } x_i = y_i \\ 0, & \text{if } x_i \neq y_i \end{cases} \tag{7}$$

$$S_i(x_i, y_i) = \begin{cases} 1, & \text{if } x_i \in [y_i - C, y_i + C] \\ 0, & \text{if } x_i \notin [y_i - C, y_i + C] \end{cases} \tag{8}$$

$$S(x, y) = \sum_{i=1}^{n} \frac{\omega_i S_i(x_i, y_i)}{n} \tag{9}$$

Since Gower coefficient is a similarity measure, it was converted to distance measure as follows: $dist(x, y) = 1 - S(x, y)$. C is some constant and ω is weight applied to similarity of the i-th element of data vector. For experiments, the value of ω was set to 1, which means that each color component of pixel RGB vector was equally important and various values of C were tested. No satisfactory results were achieved with regular and modified Gower coefficient. The final segmented image had only one, sometimes few segments with colors not similar to colors found in original photo (see table VII). Specifying random number generator seed allowed to test both programs with the same pseudo random numbers. For each parameter setting and different seed the programs were executed 10 times.

As suggested in [4] two evaluation measures F and Q were used (10,11) to assess the quality of result color image segmentation. Both measures do not require any parameter or threshold value and can be used for automatic evaluation, however it is important to remember that they should not be treated as definitive evaluation of final segmented image.

$$F(I) = \frac{1}{1000 \times N} \sqrt{r} \sum_{i=1}^{r} \frac{e_i^2}{\sqrt{A_i}} \tag{10}$$

$$Q(I) = \frac{1}{1000 \times N} \sqrt{r} \times \sum_{i=1}^{r} \left[\frac{e_i^2}{1 + \log A_i} + \left(\frac{r(A_i)}{A_i} \right)^2 \right] \tag{11}$$

I is the segmented image, N the image size (number of pixels), r the number of regions of the segmented image, while A_i and e_i are, respectively, the area and the average color error of the i-th region; e_i is defined as the sum of the Euclidean distances between RGB color vectors of the pixels of region i and the color vector attributed to region i in the segmented image. $r(A_i)$ represents the number of regions having an area equal to A_i. The smaller the F and Q, the better the segmentation result should be. Equation (10) is composed of three terms: the first is a normalization factor that takes into account the size of the image, the second, $\sqrt{r}$, penalizes segmentations that form too many regions, the last term, the sum, penalizes segmentations having non-homogeneous regions. Since the average color error e_i of the region is significantly higher for large regions than for small ones, e_i has been scaled by the factor $\sqrt{A_i}$. In equation (11) the body of the sum is composed of two terms: the first is high only for non-homogeneous regions (typically, large ones), while the second term is high only for regions whose area A_i is equal to the area of many other regions in the segmented image (typically, small ones) [4].

Number of result nonzero clusters was recorded, usually it was smaller than required through parameter c number of clusters. The average, minimum and maximum values of F, Q, serial and parallel run times were computed. During the experiments, the value of m was set to 2. Computational experiment results are presented in tables I-VI and the images are in tables VII and VIII.

VII. Conclusions

While performing computational experiments the following observations were made.

- The final number of clusters was always smaller than required c
- Using OpenMP caused significant speed-up
- Euclidean and Manhattan metrics were used successfully
- In performed experiments, both regular and modified Gower coefficient similarity measures could not be used efficiently for color image segmentation

Table VIII illustrates the best and worst result color image segmentations according to Q evaluation measure.

Plans for further research include applying various clustering algorithms to color image segmentation for example kernel methods and using different distance measures and color representations. The FCM algorithm may be sensitive to initial membership matrix U, so different initialization experiments could be performed. The source code, program output logs, input and output photos are available on-line [5].

TABLE I
EXPERIMENT 1 RESULTS: C=8, MAXITNUM=200, DIST=EUCLIDEAN, SEED=RANDOM; R - NUMBER OF OUTPUT NON-ZERO CLUSTERS

id	seed	it. num.	r	F	Q	s. time [s]	p. time [s]
1	1429546833	16	4	969.95	21937.85	10.323	5.486
2	1429547378	35	5	661.64	12932.12	22.304	11.546
3	1429547770	18	4	850.30	18145.39	11.587	6.089
4	1429548126	20	4	818.16	16749.37	12.842	6.758
5	1429548449	44	4	819.41	16973.20	27.952	14.444
6	1429549265	13	3	1292.71	29458.85	8.428	4.528
7	1429549571	25	3	1821.21	48692.53	15.982	8.351
8	1429549847	19	4	818.12	16748.04	12.217	6.407
9	1429550122	23	4	966.34	21815.84	15.388	8.059
10	1429550350	18	4	850.30	18145.39	11.585	6.116

TABLE II
EXPERIMENT 2 RESULTS: C=16, MAXITNUM=200, DIST=EUCLIDEAN, SEED=RANDOM; R - NUMBER OF OUTPUT NON-ZERO CLUSTERS

id	seed	it. num.	r	F	Q	s. time [s]	p. time [s]
1	1429554268	28	6	523.80	9184.89	64.51	32.75
2	1429554740	60	9	385.67	6427.85	137.63	69.58
3	1429555239	43	7	440.25	7504.62	98.89	50.06
4	1429555659	68	6	573.76	9821.39	155.97	78.81
5	1429556554	24	7	481.22	7418.76	55.26	28.05
6	1429556961	40	6	551.52	9711.68	91.78	46.49
7	1429557450	63	9	347.47	5299.01	146.93	74.23
8	1429557970	52	7	422.69	7111.55	122.66	65.5
9	1429558433	48	7	446.11	7535.54	110.04	55.72
10	1429558905	39	8	372.50	5664.38	89.48	49.73

TABLE III
EXPERIMENT 3 RESULTS: C=8, MAXITNUM=200, DIST=MANHATTAN, SEED=RANDOM; R - NUMBER OF OUTPUT NON-ZERO CLUSTERS

id	seed	it. num.	r	F	Q	s. time [s]	p. time [s]
1	1430606237	16	4	815.49	16742.46	8.57	4.62
2	1430606543	14	3	1268.10	28595.22	7.51	4.06
3	1430606803	12	3	1268.74	28628.85	6.48	3.53
4	1430607058	17	4	946.32	20943.45	9.12	4.87
5	1430607264	16	3	1205.41	26793.42	8.55	4.58
6	1430607686	20	5	635.21	11408.41	10.64	5.66
7	1430607926	26	6	616.88	11120.28	13.74	7.24
8	1430608179	25	3	1531.17	35771.00	13.22	7.29
9	1430608583	14	3	1268.10	28595.22	7.5	4.05
10	1430608741	20	4	827.35	16917.65	10.63	5.66

TABLE IV
EXPERIMENT 1 RESULTS: C=8, MAXITNUM=200, DIST=EUCLIDEAN, SEED=RANDOM; AVERAGE, MINIMUM AND MAXIMUM; R - NUMBER OF OUTPUT NON-ZERO CLUSTERS

	average	minimum	maximum
it. num.	23.1	13	44
r	3.9	3	5
F	986.82	661.64	1821.21
Q	22159.86	12932.12	48692.53
s. time [s]	14.860	8.428	27.952
p. time [s]	7.778	4.528	14.444

TABLE V
EXPERIMENT 2 RESULTS: C=16, MAXITNUM=200, DIST=EUCLIDEAN, SEED=RANDOM; AVERAGE, MINIMUM AND MAXIMUM; R - NUMBER OF OUTPUT NON-ZERO CLUSTERS

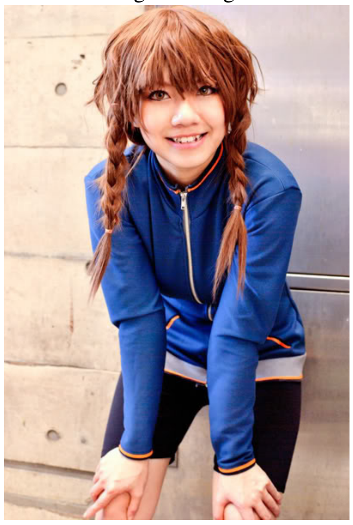
	average	minimum	maximum
it. num.	46.5	24	68
r	7.2	6	9
F	454.50	347.47	573.76
Q	7567.97	5299.01	9821.39
s. time [s]	107.315	55.26	155.97
p. time [s]	55.092	28.05	78.81

TABLE VI
EXPERIMENT 3 RESULTS: C=8, MAXITNUM=200, DIST=MANHATTAN, SEED=RANDOM; AVERAGE, MINIMUM AND MAXIMUM; R - NUMBER OF OUTPUT NON-ZERO CLUSTERS

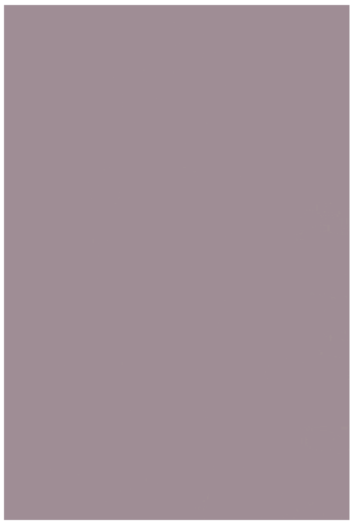
	average	minimum	maximum
it. num.	18	12	26
r	3.8	3	6
F	1038.28	616.88	1531.17
Q	22551.60	11120.28	35771.00
s. time [s]	9.59	6.48	13.74
p. time [s]	5.156	3.53	7.29

TABLE VII
SEGMENTATION RESULTS - GOWER COEFFICIENT (SCALE = 0.33)

original image



Gower coefficient



Modified Gower coefficient, $C = 32$

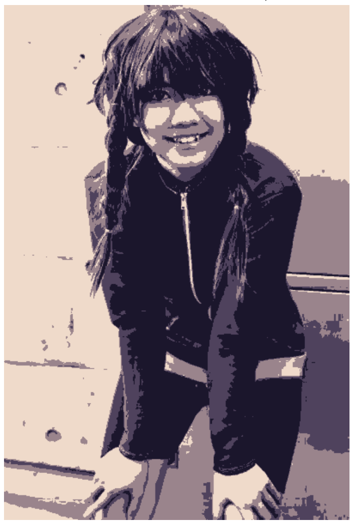
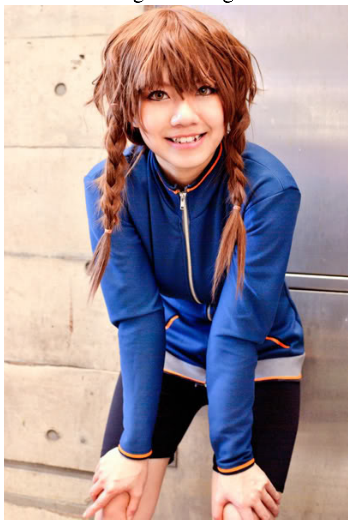
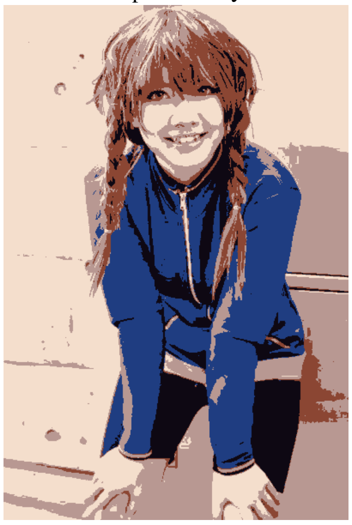
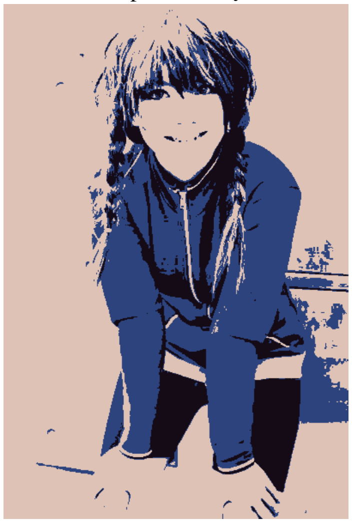
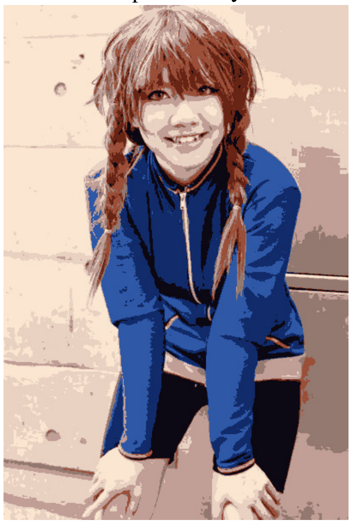
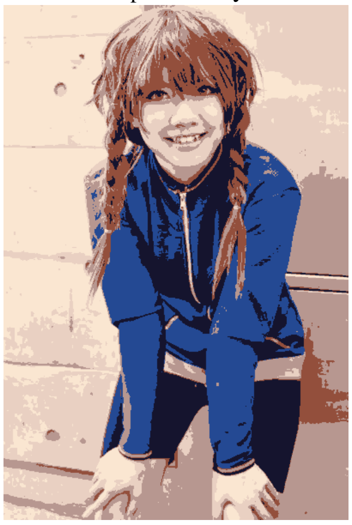
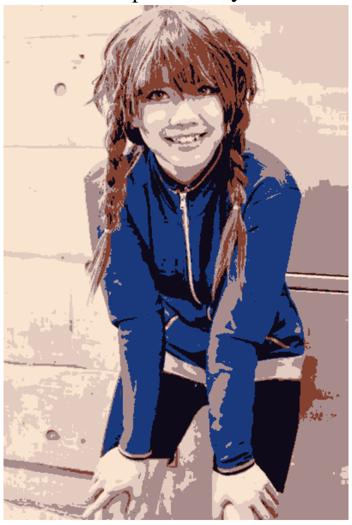
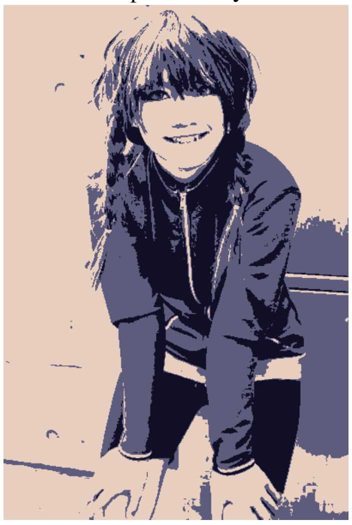


TABLE VIII
IMAGES (SCALE = 0.33)

original image	exp. 1 min Q	exp. 1 max Q
		
original image	exp. 2 min Q	exp. 2 max Q
		
original image	exp. 3 min Q	exp. 3 max Q
		

References

[1] Cheng H.D., Jiang X.H., Sun Y., Wang Jingli. 2001. Color image segmentation: advances and prospects. Pattern Recognition, Volume 34, Issue 12:2259-2281, http://dx.doi.org/10.1016/S0031-3203(00)00149-7

[2] Correa C., Constantino V., Barreiro P., Diago M. P., Tardaguila J. 2011. A Comparison of Fuzzy Clustering Algorithms Applied to Feature Extraction on Vineyard. Inteligencia artificial revista iberoamericana de inteligencia artificial, Volume 1, Issue 1:778

[3] dos Santos T.R.L, Zarate L.E. 2015. Categorical data clustering: What similarity measure to recommend? Expert Systems with Applications, Volume 42, Issue 3:1247-1260, http://dx.doi.org/10.1016/j.eswa.2014.09.012

[4] Borsotti M., Campadelli P., Schettini R. 1998. Quantitative evaluation of color image segmentation results. Pattern Recognition Letters, Volume 19, Issue 18:741-747, http://dx.doi.org/10.1016/S0167-8655(98)00052-X

[5] http://inf.ug.edu.pl/˜osakarya