Image Categorization Based on Computationally Economic LAB Colour Features

Adrian Ciobanu, Mihaela Costin, and Tudor Barbu

Institute of Computer Science of the Romanian Academy
{adrian.ciobanu,mihaela.costin,
tudor.barbu}@iit.academiaromana-is.ro

Abstract. An easy to compute and small colour feature vector is introduced in this paper, as a tool to be used in the process of retrieval or classification of similarly coloured digital images from very large databases. A particular set of „ab" planes from the LAB colour system is used, along with a specific configuration of colour regions within them. The colour feature vector is low dimensional (only 96 components), computationally economic and performs very well on a carefully selected database of rose images, publicly available.

Keywords: image retrieval, feature vector, colour similarity, colour classification, agglomerative hierarchical algorithm.

1 Introduction

In this paper we introduce a computationally economic colour feature vector extracted from digital images that are previously converted in the CIELAB colour space format (called simply LAB from here on). This kind of feature vector can be easily used for retrieving images from large databases based on their colour content, as well as for classifying/indexing these images.

The LAB colour space can be seen as a stack of „***ab***" planes, each one containing all the possible colours for a given luminance ***L*** [2]. The perpendicular ***L*** axis is going through the centre of these "***ab***" planes from low luminance values to high luminance values. The conversion of a digital image from the RGB format (with parameter values ranging from 0 to 255 for each red, green and blue colour channel) into the LAB format, results in real numbers for the luminance ***L***, and ***a*** and ***b*** parameters, but these values can be conveniently transformed again into the range from 0 to 255, as it is automatically done, for instance, in Matlab.

We can see the uniform distribution of colours in the "***ab***" planes of a LAB color space in Fig. 1, with colours varying from green to red along the "***a***" axis and from blue to yellow along the "***b***" axis. A particular color is defined by a certain point ($\boldsymbol{a_i}$, $\boldsymbol{b_i}$) in the "***ab***" plane of the corresponding luminance $\boldsymbol{L_i}$. Moving gradually away from this point in the same plane results in going through uniformly changed similar colours. The LAB color space was specially designed with the purpose of the human eye to perceive this gradual change of colour as a uniform one.

V.E. Balas et al. (Eds.): Soft Computing Applications, AISC 195, pp. 585–593.
springerlink.com

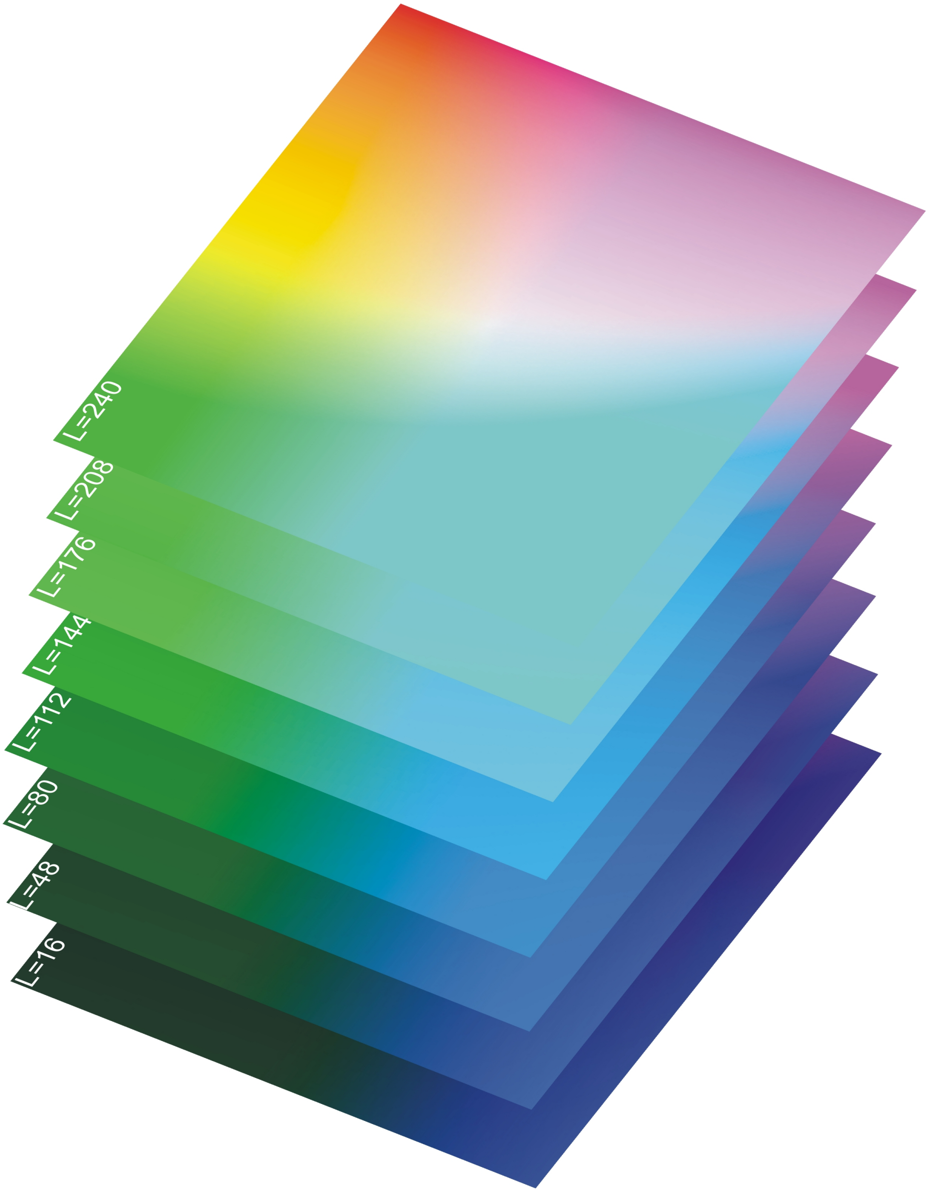

Fig. 1. The distribution of colours in the LAB colour space

2 Simple LAB Colour Feature Vector

It is usual to reduce the number of possible colours before extracting a feature vector. This is better to have as few components as possible if it is devised for large databases of images. Within the LAB colour space, we can obtain this reduction just by cutting the number of the "***ab***" planes from a total of 256 to just 8. The following central $\boldsymbol{L}$ values were taken into consideration by us: 16, 48, 80, 112, 144, 176, 208 and 240. The reduction was implemented simply by browsing all the pixels in the image and replacing the values of $\boldsymbol{L}$ with the nearest value in this set. The image is not significantly changed after this procedure from the point of view of colour content, while there is some loss in texture.

In our previous works [3][4][5] we used to create a 256×256 bins histogram for each of the 8 "***ab***" planes. However, we noticed that the bins of such histograms are mostly empty, as only a few of the $(\boldsymbol{a},\boldsymbol{b})$ possible pairs correspond to pixels in images. Moreover, it was hard to extract only a small number of features from these histograms. So, in this paper we are taking another approach that simply divides each "***ab***" plane in 12 particular regions and constructs a histogram with only 12 bins for each plane (see Fig. 2). This is resulting in only 96 bins for the LAB histograms of an image, taking in consideration all the 8 considered "***ab***" planes.

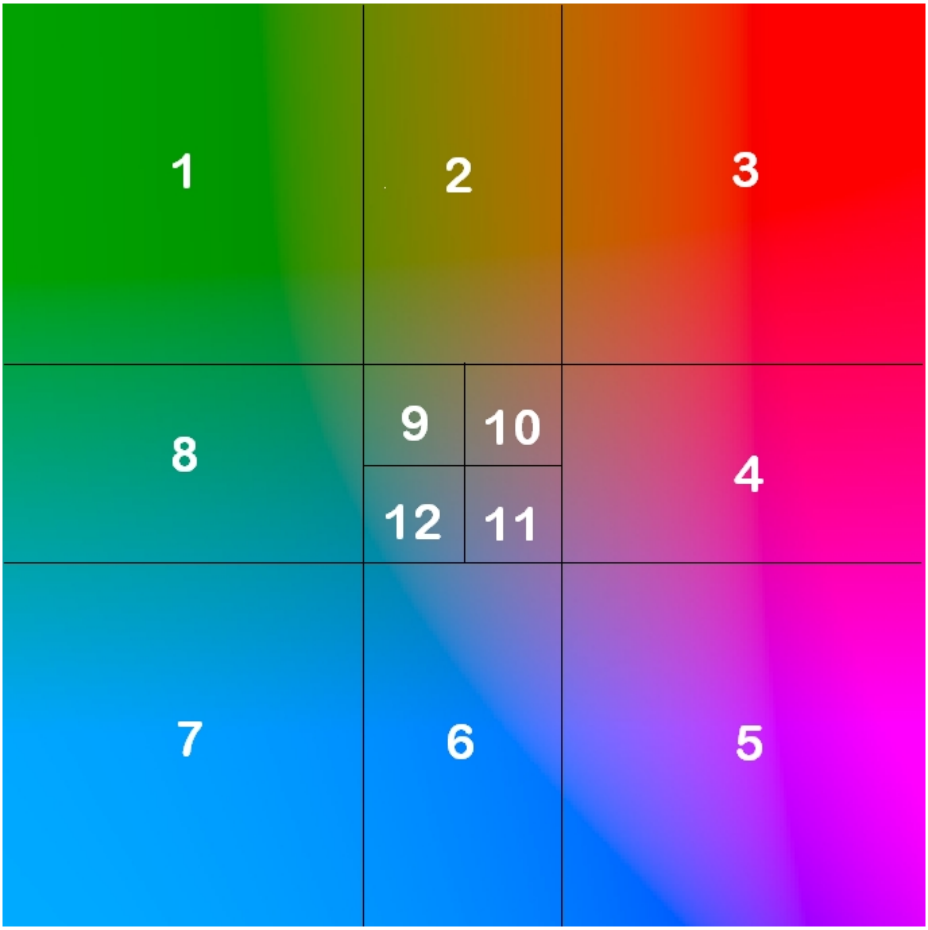

Fig. 2. The choice for splitting an „ab" plane into 12 regions

This particular choice of the set of regions resulted from our extensive tests concerning possible colour feature vectors within the LAB colour system. Regions 1, 3, 5 and 7 are somehow uniform in colour. However, in everyday pictures taken outside, a few numbers of colours correspond to these regions. Most of them are found in areas of the regions nearing the center of an “ab” plane. More colours in digital images fall in the 2, 4, 6 and 8 regions, as here we have colours that combine red and green (for instance yellow at high luminance values), red and violet, violet and blue, and blue and green. We found still more colours in the centre of the “***ab***” planes, as here is the region for black (at low luminance values), white (at high luminance values) and all sort of gray and brownish tones that are very much present in everyday pictures. In our experiments we found a lot of hits in the regions 9, 10, 11 and 12 and this is the reason why we have devised such small regions in the center of the “***ab***” planes.

To compute the histogram based on the 8 “***ab***” planes and the 12 regions for each plane, we simply browse all the pixels of an LAB converted image and for each pixel we add one unit to the corresponding bin. First we find the ***L*** value and select the corresponding “***ab***” plane and then we find the corresponding colour region in that plane by testing the ***a*** and ***b*** values of each pixel. So, extracting the feature vector is just a problem of counting, without any computation that could induce long processing times for high resolution images. The feature vector with 96 components is then constructed by putting the 12 values for each plane in the order indicated in Fig. 2 and then by concatenating the values for each plane in the order of their specific ***L*** value, from low values (16) to high values (240). For an image ***I***, the colour feature vector can be expressed as:

$$F(I) = \left(n_1, n_2, \ldots, n_{12}, n_{13}, \ldots, n_{24}, n_{25}, \ldots, n_{96}\right) \tag{1}$$

where $\boldsymbol{n_i}$, $\boldsymbol{i} = 1,\ldots,96$, are the number of pixels counted as corresponding to each colour region in the above described configuration. In order to compare the feature vectors computed for two images $\boldsymbol{I^1}$ and $\boldsymbol{I^2}$, we used as a distance the sum of the absolute differences of these feature vectors:

$$d(I^1, I^2) = d(F^1, F^2) = \sum_{i=1}^{96} \left| n_i^1 - n_i^2 \right| \tag{2}$$

A small value of the distance ***d*** means images I^1 and $\boldsymbol{I^2}$ are similarly close to each other from their colour composition point of view.

3 Experimental Results

In order to assess the performances of our colour feature vector, we performed extensive trials on the known reduced Corel database that consists of 1,000 colour images, grouped in 10 classes of 100 images of the same dimension: 384×256 pixels or 256×384 pixels, with 24 bits allocated for each RGB coded pixel. The images are conveniently numbered from 0 to 999. While each class of images has a specific theme, from the colour point of view they are not so homogenous. With this simple colour feature vector we have obtained better results in retrieving similar images than those obtained with our previous colour feature vectors. However, running tests on the whole Corel database is not as significant as running tests on some images where the colour content matter the most. And for this purpose we selected the class of images depicting roses, numbered from 600 to 699. Normally each image presents only one rose in the center of the image on some dark background. There are also some images with several roses and the backgrounds may differ from one image to another. So, we make the colour of roses even more important by taking into account only the center of each image, an area of 144×96 pixels, or 96×144 pixels, depending on the initial orientation of the image.

With this arrangement of 100 centers of the original images as input image files, we extracted the feature vectors and computed the distance between all the possible pairs of images and then, for each image, we have retrieved the five closer images. Figure 3 shows several examples of retrieved images and we can see that the computed distance between images, based on our simple colour feature vector, works very well in finding similarly coloured images.

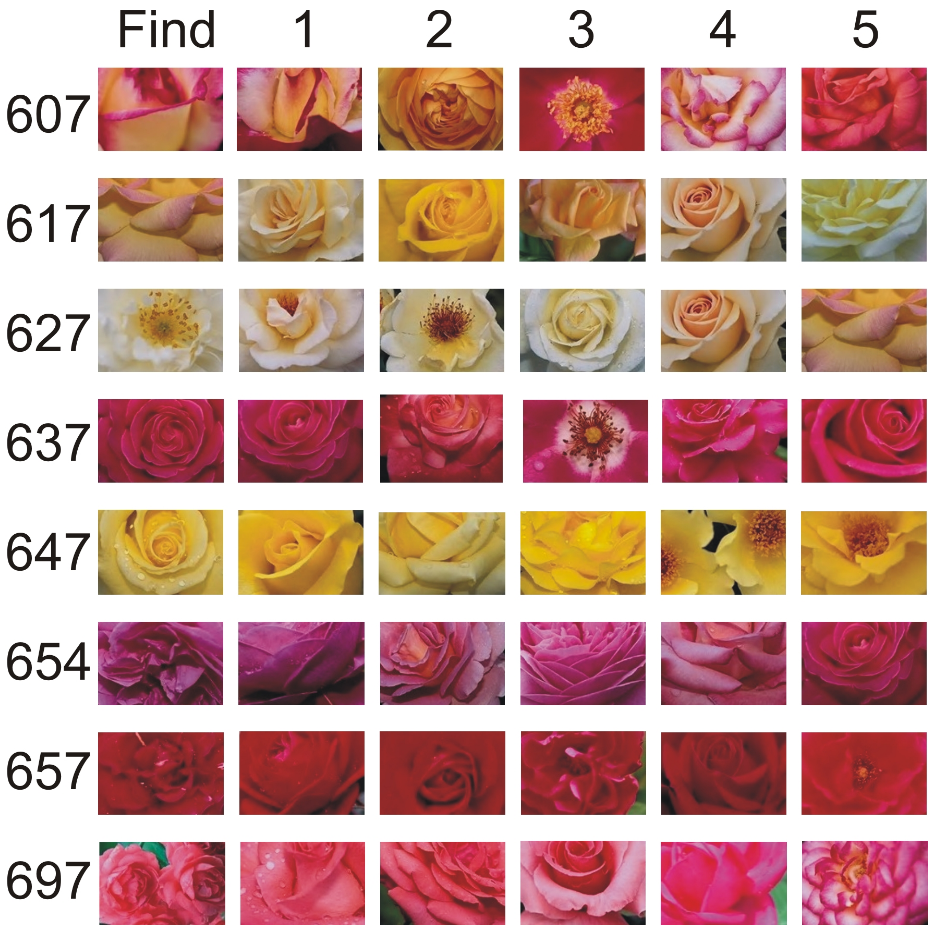

Fig. 3. Examples of images retrieval

Moreover, we wanted to see how this colour similarity distance performs in the case of a classification task. We used an agglomerative hierarchical algorithm [6] to classify the 100 images ($\boldsymbol{I^1}, \boldsymbol{I^2}, \ldots, \boldsymbol{I^{100}}$) of roses based on our colour feature vector and its associated sum of absolute differences distance. Image $\boldsymbol{I^1}$ corresponded to image file 600 in the Corel database, image $\boldsymbol{I^2}$ to image file 601 and so on up to $\boldsymbol{I^{100}}$ that corresponded to image file 699. The following steps were taken:

1. Initially, we had $\boldsymbol{m}$ = 100 clusters, which were represented by the colour feature vector of each rose image

$$C_1 = \{ F(I^1) \}, \ldots, C_{100} = \{ F(I^{100}) \} \tag{3}$$

2. The overall minimum distance $\boldsymbol{d_{\min}}$ between the 100 initial clusters is then found by inspecting the table with all the distances between the rose images. Then the two initial clusters C_i and C_j ($i < j$) that correspond to that minimum distance are merged into cluster C_i and the cluster C_j is emptied, so that we remain with only $\boldsymbol{m}$ = 99 clusters.

$$dist(C_i, C_j) = d_{min} \Rightarrow C_i = C_i \cup C_j, C_j = \phi \tag{4}$$

3. Now, we recomputed the distances between the remaining clusters with the following formula:

$$dist(C_i, C_j) = \frac{\sum_{x \in C_i} \sum_{y \in C_j} d(x, y)}{card(C_i) \cdot card(C_j)} \tag{5}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are colour feature vectors for the images in clusters $\boldsymbol{C_i}$ and $\boldsymbol{C_j}$, $\boldsymbol{d}$ is computed according to (2), and $\boldsymbol{card(C_i)}$ represents the cardinality of the cluster $\boldsymbol{C_i}$, i.e. the number of colour feature vectors in that cluster at a given iteration. The minimum distance is again computed with:

$$d_{min} = \min_{i \neq j \in [1, m]} dist(C_i, C_j) \tag{6}$$

and the move (4) is again applied, reducing the number of remaining clusters to $\boldsymbol{m}$ - 1.

The step 3 can be repeated several times, for instance until a specific number of clusters is reached. However, we were very much interested how the clusters are developing and we followed the algorithm step by step. Figure 4 shows the situation when 34 clusters were reached.

11	600	636	628	605	614	649	615	633	666	619	631	0	0	0
6	601	610	657	667	616	678	0	0	0	0	0	0	0	0
14	602	621	691	653	603	630	608	626	639	643	634	669	673	696
3	604	627	642	0	0	0	0	0	0	0	0	0	0	0
1	606	0	0	0	0	0	0	0	0	0	0	0	0	0
2	607	655	0	0	0	0	0	0	0	0	0	0	0	0
7	609	623	647	670	658	644	641	0	0	0	0	0	0	0
2	611	617	0	0	0	0	0	0	0	0	0	0	0	0
1	612	0	0	0	0	0	0	0	0	0	0	0	0	0
6	613	645	674	665	693	695	0	0	0	0	0	0	0	0
6	618	624	629	659	620	650	0	0	0	0	0	0	0	0
5	622	660	664	668	661	0	0	0	0	0	0	0	0	0
2	625	685	0	0	0	0	0	0	0	0	0	0	0	0
2	632	638	0	0	0	0	0	0	0	0	0	0	0	0
1	635	0	0	0	0	0	0	0	0	0	0	0	0	0
2	637	663	0	0	0	0	0	0	0	0	0	0	0	0
2	640	654	0	0	0	0	0	0	0	0	0	0	0	0
3	646	648	697	0	0	0	0	0	0	0	0	0	0	0
2	651	689	0	0	0	0	0	0	0	0	0	0	0	0
3	652	677	690	0	0	0	0	0	0	0	0	0	0	0
1	656	0	0	0	0	0	0	0	0	0	0	0	0	0
1	662	0	0	0	0	0	0	0	0	0	0	0	0	0
2	671	684	0	0	0	0	0	0	0	0	0	0	0	0
2	672	681	0	0	0	0	0	0	0	0	0	0	0	0
1	675	0	0	0	0	0	0	0	0	0	0	0	0	0
2	676	680	0	0	0	0	0	0	0	0	0	0	0	0
1	679	0	0	0	0	0	0	0	0	0	0	0	0	0
2	682	687	0	0	0	0	0	0	0	0	0	0	0	0
2	683	694	0	0	0	0	0	0	0	0	0	0	0	0
1	686	0	0	0	0	0	0	0	0	0	0	0	0	0
1	688	0	0	0	0	0	0	0	0	0	0	0	0	0
1	692	0	0	0	0	0	0	0	0	0	0	0	0	0
1	698	0	0	0	0	0	0	0	0	0	0	0	0	0
1	699	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 4. The formation of clusters when only 34 were reached (first column gives the number of images in one cluster and then the file names of the images contained in each cluster are listed)

We stopped the iterations when 17 clusters remained because we considered the result sufficiently good to demonstrate the capabilities of classification for our simple colour feature vector (see Figure 5)

11	600	636	628	605	614	649	615	633	666	619	631	0	0	0
14	601	610	657	667	616	678	688	618	624	629	659	620	650	662
14	602	621	691	653	603	630	608	626	639	643	634	669	673	696
6	604	627	642	625	685	656	0	0	0	0	0	0	0	0
10	606	646	648	697	613	645	674	665	693	695	0	0	0	0
4	607	655	612	635	0	0	0	0	0	0	0	0	0	0
7	609	623	647	670	658	644	641	0	0	0	0	0	0	0
4	611	617	692	698	0	0	0	0	0	0	0	0	0	0
12	622	660	664	668	661	640	654	637	663	652	677	690	0	0
2	632	638	0	0	0	0	0	0	0	0	0	0	0	0
4	651	689	683	694	0	0	0	0	0	0	0	0	0	0
2	671	684	0	0	0	0	0	0	0	0	0	0	0	0
2	672	681	0	0	0	0	0	0	0	0	0	0	0	0
2	675	679	0	0	0	0	0	0	0	0	0	0	0	0
2	676	680	0	0	0	0	0	0	0	0	0	0	0	0
3	682	687	699	0	0	0	0	0	0	0	0	0	0	0
1	686	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 5. The formation of clusters when we stopped the iterations and 17 clusters were formed

The obtained classification is presented in Figure 6, where we concatenated the image files that correspond to file numbers in Figure 5. Here we show the entire rose images, although overlapped, but one must keep in mind that the features were taken only for their central part, as mentioned above.

Fig. 6. Example of rose categorization

We can see that we obtained 5 large classes, two of them with 14 images (nuances of red roses), and three of them with 12, 11 (nuances of pink roses) and, respectively 10 images (nuances of orange roses). There are two other relevant clusters with 7 and 6 images for white and, respectively, yellow roses. And there are some small clusters, which would probably attach to one of the big clusters if we would not have stopped the iterations. There is only one image that didn't cluster at all, and it is very different from the others, which is a good result, too. Notice that the clusterization is done only by colour, different petal types of roses getting together in the same class because of their similar colour. Probably, a supplementary stage of texture classification would give good results in identifying the same species of roses.

4 Conclusion

We introduced in this paper a simple colour feature vector based on processing images in the LAB colour format. After converting images from the usual RGB format in the CIELAB format, the features are obtained just by counting the number of pixels in the images belonging to some specific ranges of colour, so the computation burden is low. Also, the relatively reduced number of components, only 96, make this colour feature vector feasible to be used with very large databases of images. We presented some interesting experimental results showing the good capabilities of this colour feature vector when used for the tasks of retrieving similarly coloured images or even classifying them on the base of their colour content.

Further research will be done for preprocessing images in the LAB format and make a selection of the most important colour pixels to be counted, in order to obtain an even more characteristic feature vector.

References

[1] Jain, A.K.: Fundamentals of Digital Image Processing. Prentice-Hall International, Inc. (1989)

[2] Fairchild, M.D.: Color and Image Appearance Models. Color Appearance Models, pp. 340–385. John Wiley and Sons (2005)

[3] Ciobanu, A., Costin, M., Barbu, T.: Extraction of main colors from a digital color image. In: Proceedings of the International Multidisciplinary Scientific Geo-Conference SGEM 2010, Albena, Bulgaria, June 20-25, vol. 1, pp. 1069–1076 (2010)

[4] Ciobanu, A., Costin, M.: Improved Color Analysis and Feature Extraction Method for Digital Images. In: 6th European Conference on Intelligent Systems and Technologies, ECIT 2010, Iasi, Romania (October 2010)

[5] Barbu, T., Ciobanu, A., Costin, M.: Unsupervised Color-Based Image Recognition using a LAB Feature Extraction Technique, Buletinul Institutului Politehnic Iaşi, Universitatea Tehnică "Gheorghe Asachi", Tom LVII (LXI), Fasc. 3, Secția Automatică şi Calculatoare (2011)

[6] Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R., Wu, A.Y.: An Efficient K-Means Clustering Algorithm: Analysis and Implementation. IEEE Trans. on Pattern Analysis and Machine Intelligence 24(7), 881–892 (2002)

[7] Sonka, M., Hlavac, V., Boyle, R.: Image Processing, Analysis, and Machine Vision. International Student Edition, Thomson (2008)

[8] Schettini, R., Ciocca, G., Zuffi, S.: Indexing and Retrieval in Colour Image Databases. In: Lindsay, W., RonnierLuo, M. (eds.) Indexing and Retrieval in Colour Image Databases. ch. 10, pp. 183–213. John Wiley & Sons, England (2002)