Serialized *k*-Means for Adaptative Color Image Segmentation

Application to Document Images and Others

Yann Leydier[1,2], Frank Le Bourgeois[2], and Hubert Emptoz[2]

[1] Archimed Lyon, 4 quai des Étroits, 69005 Lyon, France
y.leydier@archimed.fr
[2] LIRIS, 20 av. Albert Einstein, 69621 Villeurbanne, France
{frank.lebourgeois|hubert.emptoz}@liris.cnrs.fr

Abstract. This paper introduces an adaptative segmentation system that was designed for color document image analysis. The method is based on the serialization of a *k*-means algorithm that is applied sequentially by using a sliding window over the image. During the window's displacement, the algorithm reuses information from the clusters computed in the previous window and automatically adjusts them in order to adapt the classifier to any new local variation of the colors. To improve the results, we propose to define several different clusters in the color feature space for each logical class. We also reintroduce the user into the initialization step to define the number of classes and the different samples for each class. This method has been tested successfully on ancient color manuscripts, video images and multiple natural and non-natural images having heavy defects and showing illumination variation and transparency. The proposed algorithm is generic enough to be applied on a large variety of images for different purposes such as color image segmentation as well as binarization.

1 Introduction

Every year, great amounts of manuscripts are digitized in the world to preserve cultural legacy. Today, some of these images are available on the Internet in online digital libraries. Most of them are manually indexed at great cost, thus the automation of this task has become a necessity. Several indexing systems have been already proposed for document information retrieval using image analysis tools. Most of them are accurate when applied on printed document archives but very few propose a solution to index ancient manuscripts. The difficulty to analyze automatically digitized manuscripts is related to the complexity of the segmentation. This is due to the bad quality of the documents images, and the complexity of their contents. Ancient manuscripts have a very rich content [1] and require a multicolor segmentation to extract colored text, illuminated marks, etc...

Generic algorithms for color image segmentation are difficult to apply on document images for many reasons. Firstly, documents are generally digitized using high resolution, which provide large digital images that slow down classical segmentation algorithms. It makes it difficult to achieve a global segmentation on the entire image

S. Marinai and A. Dengel (Eds.): DAS 2004, LNCS 3163, pp. 252–263, 2004.

without memory overloads and excessive computing time. On the other hand, images of digitized documents have some particular defects that make the use of classical segmentation algorithm difficult. Figure 1 shows the defects we usually find in color images of ancient manuscripts like stains, holes, humidity marks, degradation of the ink, characters strokes, color variation of the paper, recto/verso transparency...

In most of the image processing tools described in the literature, some restauration might be done ahead of the segmentation. The proposed algorithm can be used to restaurate and/or segment images.

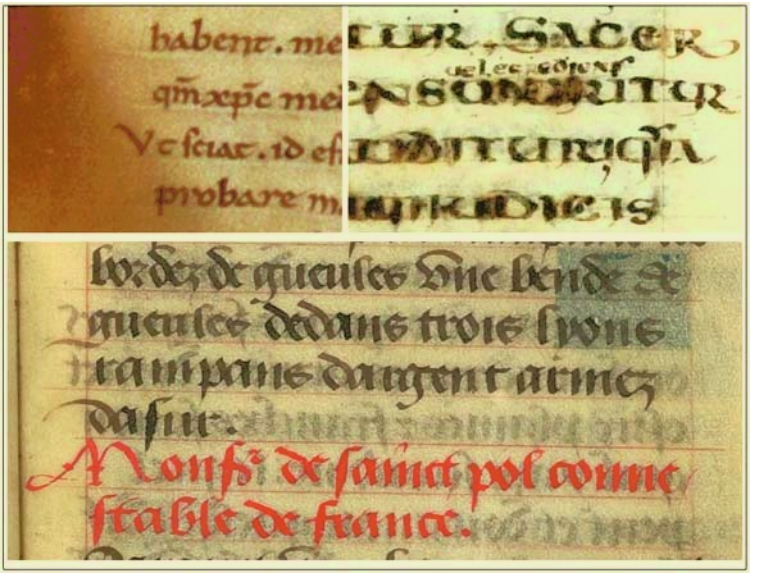

Fig. 1. Samples of color images of ancient manuscripts.

2 Proposition

2.1 Serialization of the *k*-Means Algorithm

Document image processing is generally better achieved by adaptative segmentation methods. It has been previously shown that adaptative thresholding algorithms, such as Niblack's or Sauvola's [9], perform better than non-adaptative ones like Otsu's or Fisher's. It is explained by the specific defects found in document images, such as stains, illumination variation and color fading which need to be processed by a highly adaptative algorithm. It is generally achieved by considering the local information in the neighborhood centered on each pixel using a sliding window.

The principle of our method is to apply an unsupervised classifier, like the *k*-means algorithm, on a sliding window so that the segmentation is neighbor-dependent and allows the classifier to slightly adapt the clusters to any new local color information. We

chose the *k*-means algorithm because it is the simplest and most efficient unsupervised classifier that can be easily modified and controlled, but other classifiers can be used.

In our algorithm, each point P of the image is labelled by a *k*-means classifier that is initialized with the clusters' centers of the previous classification and trained on a window w_P centered on P. The dependence between successive classifications is justified by the fact that most of the information is similar in overlapping windows. The serialization of the *k*-means has three properties:

- An important reduction of the amount of iterations to stabilize the centers.
- A higher adaptativity of the segmentation as the center of each cluster can move swiftly.
- Every cluster does *not* have to be represented in each window.

We choose to serialize the *k*-means along the x-axis because most of our images, which show open books, have a larger width than the height. Further tests show that the adaptation along the y-axis does not provide better results and may lead to an extra computation time. At the beginning of each scanline, the centers of the color clusters are initialized with the original centers privided by the user (see section 2.3) then, along the scanline, we ensure that the centers do not swap during the adaptation (see section 2.4).

2.2 Choice of the Features

Our serialized *k*-means is independent from the choice of the features and the dimension of the feature space. Experiments show that we need the maximal amount of information available to separate, for example, some verso characters that are visible by transparency from faded recto characters that have approximately the same luminosity and hue.

At first we applied the classification on the RGB channels as our images are color scans and some parts of the text are colored [6].

On some images, using the YUV[1] coordinates system – which is a linear combination of RGB – can slightly improve the segmentation accuracy.

The HSL colorspace (Hue, Saturation, Luminosity) is non-linearly calculated from RGB. The Hue channel is the angle of the color on the chromatic circle; therefore we implemented the circular distance and circular mean to use this feature adequately. The HSL channels are highly relevant, and do boost the segmentation accuracy [5].

We also use the standard deviation of the grayscale cooccurrence matrix calculated in sliding windows. This feature is generally not suited for ancient manuscript images but can improve the results on natural or dithered images.

Other features such as Euclidean colorspaces like LU*V* [7] or more texture information [8] can be used, since our classifier is designed to be used with feature vectors whose dimension is not preset.

For general purpose we use both the HSL and RGB channels, thus having 6 dimensionnal feature space.

[1] the PAL television luminance/chrominance

2.3 Initialization of the Algorithm

As our document images are noisy and unequally illuminated, some pixels of a same class located at different parts of an image may be very different. Furthermore, some classes may have very tight borders (e.g. verso text versus deteriorated recto text), and we need to describe those borders quite precisely, that is to define several clusters for each logical class. Therefore, we use a hierarchical classification: meta-classes that are made of several clusters in the feature space [2, 3].

Generally in color document images, color clusters are not Gaussian [4], thus it is difficult to determine the amount of clusters that is required. This explains why the number of logical classes, the number of clusters for each class and their initial centers must be provided by the user.

In our application, a user interface allows us to set the number of logical classes and to select the multiple color samples for each logical class that initialize the original centers of the clusters, which we will call $\left\{C_i^{\text{init}}\right\}_{i\in[1\ldots k]}$. Those color samples are the means of the pixels in rectangles designated in the image by the user. This initialization is done only once for all the images from a same book. The amount k of clusters is equal to the number of color samples and not the amount M of logical classes. We assume that the user provides at least one sample of color for each logical class ($k \geq M$). Figure 2a shows all the colors from the image figure 4a and the centers of clusters in the three dimensional RGB space. There are three classes ($M = 3$) in this image that are described with different amounts of clusters (ω_1 is red text with 1 center, ω_2 is black text with 2 centers and ω_3 is the background and transparent verso with 5 centers).

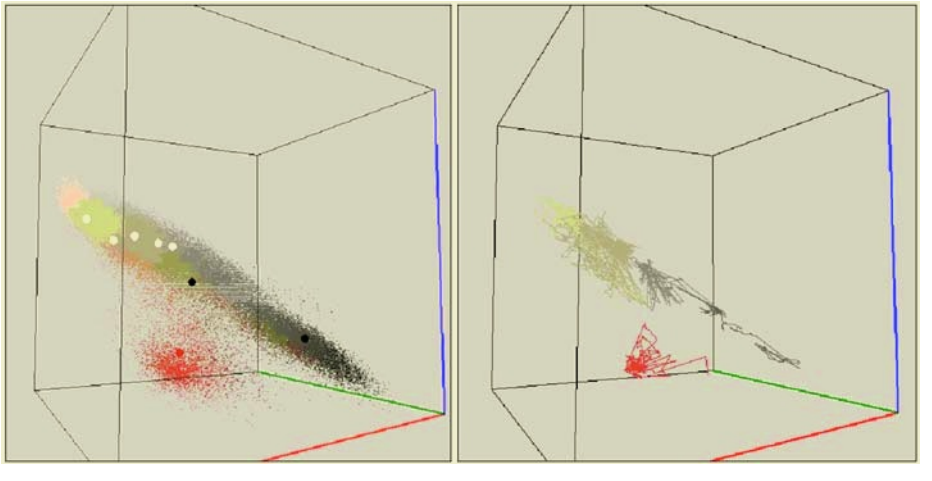

Fig. 2. a) RGB space and the clusters' centers, b) Displacement of the clusters' centers along a scanline.

2.4 Class Swapping Prevention

An automatic unsupervised classifier like the k-means algorithm can easily swap the cluster centers. If serialized, this creates inversions between the clusters and class labels. We prevent our algorithm from centers swapping after the centers' stabilization in order to guarantee the convergence of the classification and the relevance of the labelling.

To do that, for each window w_P, after the training of the k-means and before the labelling of P, we compute the distance between each new cluster center $C_i^{w_P}$ to a set of reference centers $\left\{C_i^{\text{ref}_{w_P}}\right\}_{i\in[1\ldots k]}$. If the nearest reference center is not the one with the same label, then $C_i^{w_P}$ is moved (back) to the reference center $C_i^{\text{ref}_{w_P}}$:

$$\forall i \in [1\ldots k],\ \underset{j\in[1\ldots k]}{\text{argmin}}\, d\left(C_i^{w_P}, C_j^{\text{ref}_{w_P}}\right) \neq i \ \Rightarrow\ C_i^{w_P} \leftarrow C_i^{\text{ref}_{w_P}}$$

The reference centers, which are re-calculated for each window, are linear[2] combinations of the previous centers $\left\{C_i^{w_{P^-}}\right\}_{i\in[1\ldots k]}$ and the original centers $\left\{C_i^{\text{init}}\right\}_{i\in[1\ldots k]}$ that were selected by the user in the initialization step.

$$\forall i \in [1\ldots k], C_i^{\text{ref}_{w_P}} = (1-\lambda)\cdot C_i^{\text{init}} + \lambda \cdot C_i^{w_{P^-}},\ \lambda \in [0,1]$$

The best results are achieved with $0 \leq \lambda \leq 0.7$.

Such a limitation of the centers displacement around the original centers does not restrain the adaptation of clusters. Figure 2b shows the adaptation of the centers of the clusters in the RGB space along a scanline of the image given in figure 4a with reference centers equal to the previous centers.

2.5 Labelling of the Pixels

In a normal case, a point P is affected the label of the nearest cluster center. This is troublesome when dealing with dithered images because an apparent color can be the result of the juxtaposition of two different colors. Let us consider an image with two colors, c_1 and c_2, and one level of dithering. We'll try to segment this image into three classes. As our algorithm initializes the centers with the mean feature vector calculated in windows given by the user, the centers will be C_1^{init} of color c_1, C_2^{init} of color c_2 and C_3^{init} of color $\mu_f(c_1,c_2)$, with μ_f the mean function adequated to the feature-vector the user chose. If we do not pay attention, no pixel will be classified in the cluster C_3 as the only pixels that are present in the image are of color c_1 or c_2. This is a problem that no common classifier, even a serialized and adaptative one, can overcome.

Assuming that, in a given window, $C_3^{w_P}$ has been assigned no pixel, if the clusters $C_1^{w_P}$ and $C_2^{w_P}$ have approximatively the same amount of samples, we calculate the spatial barycenters[3] B_1 and B_2 of the pixels of the window that are of color c_1 and c_2 (as shown in figure 3). From this, we can conclude that:

[2] or circular for the Hue feature

[3] i.e. the centroids

1. if B_1 and B_2 are distinct, the window is centered on the edge of two uniform regions of color c_1 and c_2,
2. if B_1 is very close to B_2, the window is centered on a dithered part of the image *or* the window is centered on a particular kind of symmetric pattern. Here, a frequency analysis can lead to a more precise separation.

In case 2, we decide to apply a specific algorithm to manage dithered images. This can lead to the misclassification of some symetric patterns but it is a lesser harm than having C_3 attributed to no pixel.

Generally a smoothing algorithm is used to process dithered images but to much smoothing destroys the thin strokes of the characters and smears close patterns.

Therefore we adapt the classification in case 2 by choosing the label of a pixel with the following criterion:

$$l = \underset{i\in[1\ldots k]}{\operatorname{argmin}} \left\{ d\left(f\left(P\right), C_i^{w_P}\right), d\left(f\left(\mu_\sigma\left(w_P\right)\right), C_i^{w_P}\right)\right\}$$

where l is the label given to the pixel P, $\{C_i^{w_P}\}$ the cluster centers for the window w_P, f the function we apply to a pixel to obtain its feature vector and $\mu_\sigma\left(w_P\right)$ the Gaussian mean of w_P with a standard deviation σ and centered on P.

The σ parameter allows to overcome very local stains by smoothing the classification of the pixels without smoothing the image. Images with noise or dithering are to be processed with $0.7 \leq \sigma \leq 1.4$ to smooth the classification, whereas clean documents images must be processed with no smoothing ($\sigma \leq 0.5$).

Another parameter ρ, allows to control the serialization of the algorithm. It defines a distance between a pixel and its closest center from which the pixel will not be used to calculate the new center. Therefore, if $\rho = 0$, the serialization is blocked and the algorithm is just a windowed k-means. If $\rho \to \infty$, the algorithm is fully serialized.

It is useful to control the serialization when processing dithered images for in that case, if fully serialized, the algorithm would be able to adapt to each color of the dithering and the result will also be dithered. Generally, for dithered images, we use $\sigma > 0.5$ and $\rho = 0$ (no serialization) and for shaded images having illumination variations, we use $\sigma \leq 0.5$ and $\rho \to \infty$ (fully serialized).

Fig. 3. Spatial barycenters of a dithered image and a shaded image.

2.6 Summary

Let us call f the function we apply to a point P to obtain its feature vector. For each w_P centered on P, at first we calculate the new cluster centers. This is a normal k-means process to which we add the ρ condition:

$$\forall i \in [1\dots k], C_i^{w_P} = \mu_f \left(\left\{ f(Q), Q \in w_P / \left\{ \begin{array}{l} \underset{j\in[1\dots k]}{\operatorname{argmin}}\ d\left(f(Q), C_j^{w_{P^-}}\right) = i \\ \underset{j\in[1\dots k]}{\min}\ d\left(f(Q), C_j^{w_{P^-}}\right) < \rho \end{array} \right\} \right)$$

where $C_j^{w_{P^-}}$ is the center of the cluster number j from the previous window.

Then, we prevent the centers from swapping by stacking them to reference centers if they moved too far:

$$\forall i \in [1\dots k], C_i^{\mathrm{ref}_{w_P}} = (1-\lambda)\cdot C_i^{\mathrm{init}} + \lambda \cdot C_i^{w_{P^-}},\ \lambda \in [0,1]$$

$$\forall i \in [1\dots k],\ \underset{j\in[1\dots k]}{\operatorname{argmin}}\ d\left(C_i^{w_P}, C_j^{\mathrm{ref}_{w_P}}\right) \neq i \ \Rightarrow\ C_i^{w_P} \leftarrow C_i^{\mathrm{ref}_{w_P}}$$

We calculate the spatial barycenters of the two most populated clusters:

$$a = \underset{i\in[1\dots k]}{\operatorname{argmin}} \operatorname{card}\left(C_i^{w_P}\right),\ b = \underset{i\in[1\dots k]\setminus a}{\operatorname{argmin}} \operatorname{card}\left(C_i^{w_P}\right)$$

$$B_1 = \mu\left(\left\{ Q \in w_P / \underset{i\in[1\dots k]}{\operatorname{argmin}}\ d(f(Q), C_i^{w_P}) = a \right\}\right)$$

$$B_2 = \mu\left(\left\{ Q \in w_P / \underset{i\in[1\dots k]}{\operatorname{argmin}}\ d(f(Q), C_i^{w_P}) = b \right\}\right)$$

Finally, we classify the point P according to whether the part of the image in w_P is dithered or not:

$$\left\{ \begin{array}{l} \dfrac{\operatorname{card} C_a^{w_P}}{\operatorname{card} C_b^{w_P}} \approx 1 \\ d(B_1, B_2) < \varepsilon \end{array} \right. \text{(dithered)} \Rightarrow\ l(P) = \underset{i\in[1\dots k]}{\operatorname{argmin}} \left\{ d\left(f(P), C_i^{w_P}\right), d\left(f\left(\mu_\sigma(w_P)\right), C_i^{w_P}\right) \right\}$$

$$\begin{array}{l} \dfrac{\operatorname{card} C_a^{w_P}}{\operatorname{card} C_b^{w_P}} \neq 1 \\ \text{or } d(B_1, B_2) \geq \varepsilon \end{array} \text{(not dithered)} \Rightarrow\ l(P) = \underset{i\in[1\dots k]}{\operatorname{argmin}}\ d\left(f(P), C_i^{w_P}\right)$$

3 Results

3.1 Document Image Segmentation

We validated our algorithm on ancient manuscripts images from the *Mazarine* library that were provided by the IRHT[4]. Those images contain black and red text and illuminated marks. The verso text is transparently visible on the recto and the characters from the recto are partially faded (figure 4a).

[4] the French *Institute of Research on Texts' History*

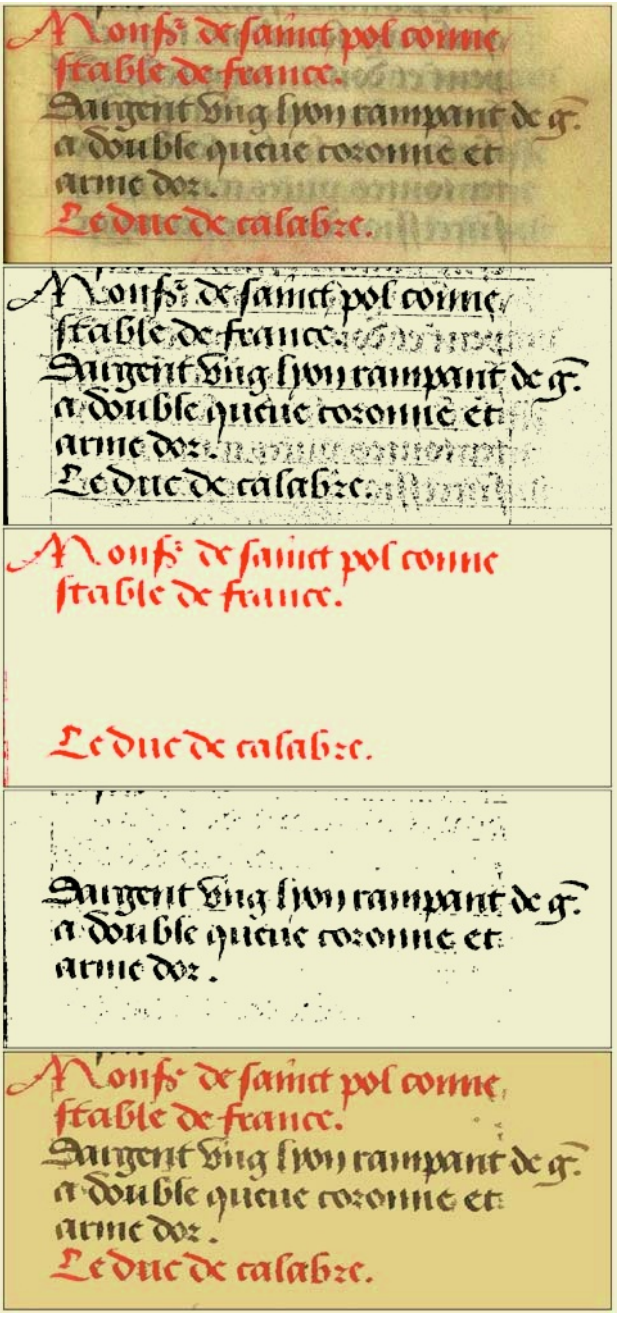

Fig. 4. a) Original ancient manuscript image from *Mazarine* library, b) Application of Sauvola's algorithm, c) Red text extracted by a serialized *k*-means, d) Black text extracted by a serialized *k*-means, e) Restaurated image.

A traditional binarization cannot lead to an accurate segmentation of the image. Figure 4b shows an application of Sauvola's algorithm for example. We use the initialization described in figure 2. The segmentation in 3 classes follows in figures 4c and 4d (the background is not represented here). The parameters are a 6×6 window size, $\lambda = 0.5$, $\sigma = 0.5$ and $\rho = 50000$ and the features are the RGB and HSL channels. Once filtered, those images can be used as stencils to create an artificial image of the document and partially restaurate the image. The background is to be replaced by its mean and the red and black layers can be copied as is (figure 4e).

3.2 Binarization

We have also studied our algorithm capability to directly binarize color images compared to other adaptative binarization algorithms, such as Niblack's and Sauvola's, applied on luminance images. Figure 5b shows Sauvola's algorithm applied to the luminance channel from figure 5a in a 18×18 window. It creates residues at the borders of the stain. Our algorithm, applied in an only 6×6 window with $\lambda = 0$, $\sigma = 0.5$ and $\rho = 50000$ on the RGB and HSL channels (figure 5c) with two classes, takes advantage of the direct analysis of the color features and shows its adaptativity.

3.3 Video Image Segmentation

Our algorithm is generic enough to be applied on a large variety of images like noisy images from compressed video. Its adaptativity can overcome the MPEG compression noise (figure 6). The parameter are $\lambda = 0.5$, $\sigma = 1$ and $\rho = 50000$ in a 6×6 window and the features are the RGB and HSL channels.

3.4 Color Segmentation of Dithered Images

Figure 7 shows an example of a dithered map that is segmented and therefore indexed. This image comes from the *MediaTeam DataBase* (image "`P03048.jpg`"). The features are the RGB and HSL channels. The parameters are a 6×6 window size, $\lambda = 0$, $\sigma = 0.8$ and $\rho = 0$. We used the color caption of the map to initialize the clusters.

3.5 Performance

By reusing the centers of the previous classification along the scanline, we reduce the number of iteration of the *k*-means to up to 17% of the amount with a basic windowed *k*-means. The average number of iteration for each window displacement is situated between 2.2 and 3.

Nevertheless the proposed algorithm, like most of the adaptative segmentation methods, is computationally expensive according to the size of the window and the dimension of the feature space. The processing of a 3000×2000 image with a 6×6 window with 6-dimensions feature vectors, $\lambda = 0$, $\sigma = 0.5$ and $\rho = 50000$ is achieved in 500 seconds on 1.5GHz PC.

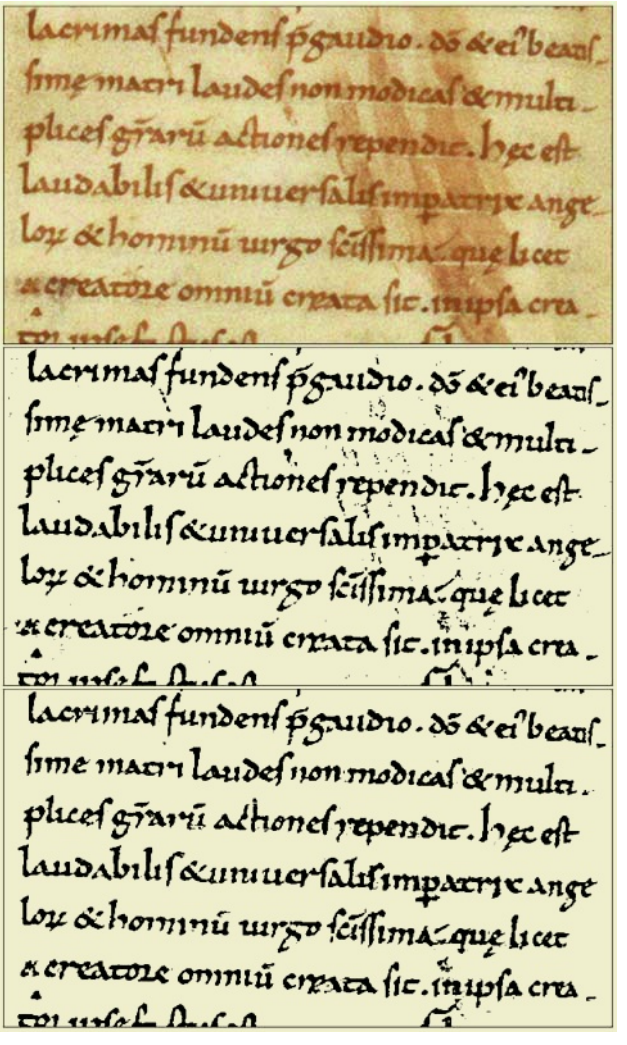

Fig. 5. a) Original ancient manuscript image from *Vendôme* library, b) Binarization achieved by Sauvola's thresholding, c) Binarization achieved by a serialized 2-means algorithm.

4 Conclusion

We presented an adaptative segmentation algorithm suited for color document images analysis based on the serialization of the *k*-means algorithm applied sequentially along each scanline. We also proposed to represent each logical class with several clusters in the feature space. A small variation of the initialization of the centers of the clusters is not critical because of the adaptativity of the algorithm. The number of clusters per class and the size of the window have a limited influence on the segmentation results. A chart will be produced to help the user in the choice of the λ, σ and ρ parameters, which can have a heavy impact on the segmentation accuracy and on the processing time.

Fig. 6. Application of the serialized *k*-means to video images.

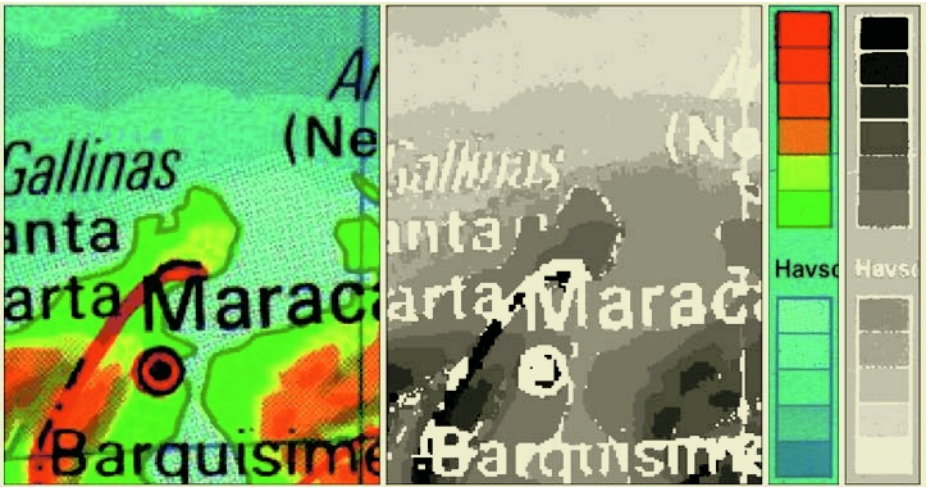

Fig. 7. Indexing of a dithered map image. The caption was used to initialize the process.

The results are quite good on all the ancient manuscript images we have processed with optimal parameters (number and position of clusters, window size, features choices). We also tested our algorithm on many kinds of images such as video frames, natural images and maps.

We plan to simplify the usage of this algorithm and assist the user to define automatically some parameters like the number of clusters for each class and the color samples for each class.

References

1. F. Le Bourgeois and al., "Document Images Analysis solutions for Digital libraries", *First International Workshop on Document Image Analysis for Libraries* (DIAL), Palo Alto, 2004.
2. J. Puzicha and S. Belongie, "Model-based halftoning for color image segmentation", *International Conference on Pattern Recognition* (ICPR) 2000, pp. 3633-3637.
3. P. Lambert and H. Grecu, "A quick and coarse color image segmentation", *International Conference on Image Processing* (ICIP) 2003, pp. 965-968.
4. L. Todoran and M. Worring, "Segmentation of Color Document Images", *ISIS technical report series*, 2000, vol 21.
5. C. Zhang and P. Wang, "A new method of color image segmentation based on intensity and hue clustering", *International Conference on Pattern Recognition* (ICPR) 2000, pp. 3617-3621.
6. S. Wesolkowski, S. Tominaga, and R.D. Dony, "Shading and Highlight Invariant Color Image Segmentation", *SPIE*, 2001, Vol. 4300, pp. 229-240.
7. D. Comaniciu and P. Meer, "Robust Analysis of Feature Spaces: Color Image Segmentation". *IEEE Conference on Computer Vision and Pattern Recognition*, Puerto Rico 1997, pp 750-755.
8. V. Eglin and S. Bres, "Analysis and interpretation of visual saliency for document functional labeling", to be published in *International Journal of Document Analysis and Recognition* (IJDAR), 2004
9. J. Sauvola, T. Seppänen, S. Haapakoski and M. Pietikäinen, "Adaptive Document Binarization", *International Conference on Document Analysis and Recognition* (ICDAR), Ulm 1997, vol. 1, pp. 147-152.