A HIERARCHICAL GRAPH-BASED MARKOVIAN CLUSTERING APPROACH FOR THE UNSUPERVISED SEGMENTATION OF TEXTURED COLOR IMAGES

Rachid Hedjam[1] *Max Mignotte*[2]

DIRO, Département d'Informatique et de Recherche Opérationnelle, Université de Montréal,
Succ. Centre-Ville, P.O. 6128, Montréal (Québec),H3C 3J7.
E-mail : [1]rhedjam@synchromedia.ca, [2]mignotte@iro.umontreal.ca

ABSTRACT

In this paper, a new unsupervised hierarchical approach to textured color images segmentation is proposed. To this end, we have designed a two-step procedure based on a grey-scale Markovian over-segmentation step, followed by a Markovian graph-based clustering algorithm, using a decreasing merging threshold schedule, which aims at progressively merging neighboring regions with similar textural features. This Hierarchical segmentation method, using two levels of representation, has been successfully applied on the Berkeley Segmentation Dataset and Benchmark (BSDB[1]). The experiments reported in this paper demonstrate that the proposed method is efficient in terms of visual evaluation and quantitative performance measures and performs well compared to the best existing state-of-the-art segmentation methods recently proposed in the literature.

Key words : Hierarchical Markovian segmentation, textural segmentation, graph partitioning, regions merging, image Berkeley database.

1. INTRODUCTION

Image segmentation is an essential tool for most image analysis tasks which consists of achieving a compact region-based description of the image scene by decomposing it into spatially coherent regions with similar attributes. This low-level vision task is often the preliminary (and also crucial) step for many image understanding algorithms and computer vision applications.

To date, a number of segmentation techniques have been proposed and studied in the last decades to solve the difficult problem of textured color image segmentation. Amongst them, we can cite clustering algorithms [2], graph-based segmentation methods (exploiting the connectivity information between neighboring pixels or regions) [3, 4, 5], hierarchical graph-based methods [6, 7], Mean-Shift-based techniques [8, 9] or finally split and merge and growing techniques (sometimes directly expressed by a global energy function to be optimized [10]).

The segmentation algorithm presented in this paper is both part of the hierarchical graph-based segmentation and the region-based split and merge procedures. More precisely, our technique relies on a two-step hierarchical procedure whose first step is a classical unsupervised Markovian over-segmentation into K classes of the input image [11][1]. This step allows us to produce a segmentation map with spatially coherent regions in the grey level sense (also called *superpixels* [12]). In this first low-level representation level, the input image is modeled by a MRF prior model defined by a graph whose pixels correspond to nodes connected to their 4 nearest neighbors. For the degradation (or likelihood) model, we have taken a Gaussian law to describe the luminance distribution within each class and parameters of this distribution are estimated thanks to an iterative method called iterative conditional estimation (ICE) [13]. In a second step, this over-segmented region map is converted into a regions adjacency graph (RAG) which is then exploited by a recent graph-based Markov Clustering[14] (MCL) approach which works by simulating random walks in graphs. In order to render this final graph partitioning procedure unsupervised and to achieve a more reliable and accurate segmentation result, we have proposed herein a decreasing merging threshold schedule in order to progressively merge neighboring regions with similar color textural features.

The remainder of this paper is organized as follows. Section 2 describes the regions adjacency graph used by the MCL algorithm and built from the Markovian over-segmented region map. Section 3 presents our iterative graph-partitioning method based on the MCL approach. Finally, Section 4 presents a set of experimental results on the Berkeley image database and comparisons with existing segmentation techniques.

2. REGION ADJACENCY GRAPH

To decrease the computational load, a preliminary merging step is achieved on the over-segmented region map that simply consists of fusing each small region (i.e., whose size is below $\Gamma = 30$ pixels) with the region sharing its longest boundary. After this pre-processing step, the over-segmented region map (R) is then modeled by a classical regions adjacency graph (RAG) in which each node represents a region to be clustered and each edge (linking two nodes of this graph) is endowed with a weight representing a color textural similarity measure between two adjacent regions.

More precisely, our textural similarity measure exploits the set of color values contained within each squared 5×5 window F_p and G_q, centered respectively at location p and q (belonging to two adjacent regions and located within a *search window* of fixed size 15×15 centered at p). In our application, this color-based textural measure is defined by

$$d(F_p, G_q) = \exp\Big(-\frac{1}{2\sigma^2}\sum_{i=1}^{N_c}(\mu_{pi} - \mu_{qi})^2\Big) \qquad (1)$$

[1]C++ Code of this unsupervised Markovian segmentation is publicly available at the following http address
www.iro.umontreal.ca/∼mignotte/ResearchMaterial/

where the first summation is done for each channel of respectively the RGB, HSV, LUV, YIQ, XYZ and LAB color spaces (each color channel has been normalized between 0 and 255) and μ_{pi} designates the mean of the color values (located in the window F_p) of the ith considered channel. σ acts as a scale parameter that allows to differentiate more or less these different distances.

In order to define a reliable textural similarity estimation between each adjacent region that takes into account the textural diversity (existing in each adjacent textured region), we have decided to take the average of the 8 smallest textural similarity distances defined in Eq. (1). This technique also allows us to be, first, largely insensitive to the parameter σ and second, it provides a more robust estimation of the textural dissimilarity measure between two adjacent regions, compared to a classical mean value of similarity measures. In this way, each edge, linking two adjacent regions r_i and r_j, is endowed with the following weight

$$\mathcal{D}_r(r_i, r_j) = \frac{1}{|r_i|} \sum_{p \in r_i} \sum_{q \in r_j} \Psi\big[d(F_p, G_q)\big] \qquad (2)$$

Where $|.|$ is the region cardinal, $\Psi[.]$ is the operator computing the average of the 8 smallest distances.

By associating a weight with each edge, this constitutes a directed RAG. In order to convert it in a undirected graph, each edge is finally associated with $\max\{\mathcal{D}_r(r_i, r_j), \mathcal{D}_r(r_j, r_i)\}$.

3. SEGMENTATION BY GRAPH PARTITIONING

Once the undirected RAG is built, the rationale of all graph partitioning methods is based on the observation that if a group a nodes (i.e., a set of regions in a RAG) is strongly connected inside and has few connections to the outside, a cluster is found. A cluster is thus defined to be a strongly connected sub-graph. According our criterion based on a textural similarity measure, this sub-graph also defines one of the spatially coherent region (in a textural sense) to be detected in the input image.

3.1. Markov Clustering (MCL)

The MCL [14] is a recent, fast, and efficient clustering algorithm for graphs, based on simulation of random walks in graphs. This algorithm is based on the property that in a graph, a random walk inside a dense cluster (i.e., a strongly connected cluster) will visit many of the nodes before leaving the cluster. Another interpretation consists of simulating flow in the RAG and promoting flow where connections are strong and demoting it where they are weak, so that flow between clusters dies out but not within clusters. Rather than simulating random walks with a (computationally demanding) stochastic approach, the MCL algorithm simulates flow using (alternating) two simple algebraic operations on the similarity matrix (i.e., the adjacency matrix associated to the RAG). The first operation is expansion, which coincides with normal matrix multiplication. Expansion models the spreading out of flow. The second is inflation, which is mathematically speaking, a Hadamard power followed by a diagonal scaling. Inflation models the contraction of flow ; it becomes thicker in regions of higher current and thinner in regions of lower current. The MCL process causes flow to spread out within natural clusters and evaporate between different clusters. The process converges towards a partition of the graph, with a set of high-flow regions (the clusters) separated by boundaries with no flow. The value of the inflation parameter r controls cluster granularity and thus influences the number of clusters and it acts as a classical regularization parameter.

Starting from $G = (V, E$, our RAG (V is the vertex set, E is the edge set) and its associated similarity matrix $A = A(G)$, the detail of the MCL clustering algorithm is given in Algorithm 1.

□ **Algorithm :** Markov Clustering (MCL)

$G = (V, E)$	Region Adjacency Graph
A	Similarity Matrix
e	Expansion Parameter
r	Inflation Parameter

while A *is not fix-point* **do**
  $A \Leftarrow A^e$ □ EXPANSION
  forall $u \in V$ **do**
    forall $v \in V$ **do**
      $A_{uv} \Leftarrow A_{uv}^r$ □ INFLATION
    forall $v \in V$ **do**
      $A_{uv} \Leftarrow \frac{A_{uv}}{\sum_{w \in V} A_{uw}}$

$C \Leftarrow$ graph induced by non-zero entries of A

Algorithm 1: MCL clustering algorithm

3.2. Iterative Graph-Partitioning Approach

In order to render this MCL-based region merging process unsupervised and to achieve a more reliable and accurate segmentation result, we have used a decreasing merging threshold schedule in order to progressively merge neighboring regions with similar color textural features. This is achieved by iterating the MCL algorithm with a slowly decreasing inflation parameter r according to the following decreasing schedule, herein (empirically) defined as a negative exponential function of the number of iterations n

$$r_n = \max\Big\{r_0 \exp\big(-\frac{n}{\tau}\big), 1\Big\} \qquad (3)$$

where $r_0 = 1.4$ in our tests and $\tau = 25$ is a constant in our application. This iterative merging process allows efficiently significant regions to progressively emerge of the background. This strategy seems especially well suited when foreground objects, to be segmented, are blended with the background due to camouflage (see Fig. 1).

It remains that the segmentation is inherently an ill-posed problem which exhibits multiple solutions for different possible values of the number of textural classes not *a priori* known. This is due to the fact that each human or each segmentation algorithm chooses to segment an image at different levels of detail. To render this problem well-posed with a unique solution, some constraints on the segmentation process are necessary, favoring over segmentation or, on the contrary, merging regions. In our iterative graph-partitioning approach, this constraint is specified by a lower bound on the final number of regions (noted N_r) to be detected. The Merging MCL process stops when the lower bound of regions is reached. In our application, our algorithm depends on two conditions for ending its execution, the first is the number of detected regions in the final segmentation, the second is the maximum number of iterations allowed (noted $N_{\max}$).

FIG. 1 – Iterative MCL segmentation. >From top to bottom and left to right, (a) original image, (b) initial oversegmentation, (c) MCL (1 iteration), (d) MCL (2 iterations).

4. EXPERIMENTAL RESULTS

4.1. Set Up

In all the experiments, we have thus considered the following internal parameters for our segmentation model. For the Markovian over-segmentation, we have considered $K = 6$ classes. For the iterative MCL clustering, the size of the window F or G and the window search are respectively set to 5×5 and 15×15. The initial inflation parameter is set to $r_0 = 1.4$ and its decreasing schedule (see Eq. (3)) uses $\tau = 25$. The expansion parameter is set to $e = 2$ (as [14]). Finally, our *a priori* lower bound on the final number of regions is $N_r = 75$ and the maximum number of iterations allowed is $N_{\max} = 4$.

4.2. Performance Measures

We have replicated the scenario used in the evaluation of state-of-the-art segmentation methods described in [15][2]. In these experiments, we have tested our segmentation algorithm (called HMC for Hierarchical Markov Clustering) on the Berkeley segmentation database [1] consisting of 300 natural color images of size 481×321. For each color image, a set of benchmark segmentation results, provided by human observers (between 4 and 7), is available and will be used to quantify the reliability of the proposed segmentation algorithm. As proposed in [15], we have compared our segmentation algorithm against five unsupervised algorithms, available publicly. For each of these algorithms, their internal parameters are set to their optimal values (see [15, 16, 17]) and/or corresponds to the internal values suggested by the authors. These algorithms are namely the Mean-Shift [8] (with $h_s = 13$, $h_r = 19$), Ncuts [4] (with a number of segments $K = 20$, agreeing with the average number of regions found in the segmentation maps given by the human observers [15]), and FH [18] (with a smoothing parameter $\sigma = 0.5$, a threshold value $k = 500$ and a minimal region size equals to 200 pixels) and the CTM (Compression-based Texture Merging) algorithm proposed in [15, 19] (with $\eta = 0.1$ and $\eta = 0.2$) and finally the FCR [17] fusion method.

As in [15], all color images are normalized to have the longest side equals to 240 pixels. The comparison is based on the following performance measures, namely the PRI measure (higher probability is better) along with three metrics VoI, GCE, BDE (lower distance is better). The qualitative meaning of these three metrics are outlined below.

[2]We have used the Matlab code, proposed by Allen Y. Yang in order to estimate the quantitative performance measures presented in this Section. This code is kindly available on-line at address http ://www.eecs.berkeley.edu/∼yang/software/ lossy_segmentation/.

	PERFORMANCE MEASURES			
Algorithms	PRI	VoI	GCE	BDE
Human	0.8754	1.1040	0.0797	4.9940
$HMC_{[1]}$	**0.7835**	3.9900	0.2900	**9.5700**
$HMC_{[2]}$	0.7816	3.8700	0.3000	**8.9300**
FCR	0.7882	2.3035	0.2114	8.9951
$CTM_{\eta=0.1}$	0.7561	2.4640	0.1767	9.4211
$CTM_{\eta=0.2}$	0.7617	2.0236	0.1877	9.8962
Mean-Shift	0.7550	2.4770	0.2598	9.7001
NCuts	0.7229	2.9329	0.2182	9.6038
FH	0.7841	2.6647	0.1895	9.9497

TAB. 1 – Performances measures (higher is better for PRI and lower is better for VoI, GCE and BDE). $HMC_{[2]}$ is the proposed algorithm with the internal parameters given in section 4.1. For $HMC_{[1]}$, we used the same parameters, except we don't take into account the stopping criterion using a lower bound of the maximal number of regions.

1. The Probabilistic Rand index PRI [20] counts the fraction of pairs of pixels whose labellings are consistent between the computed segmentation and the ground truth. The result is averaged across all human segmentations of a given image.
2. The Variation of Information (VoI) metric [21] is based on relationship between a point and its cluster. It uses mutual information metric and entropy to approximate the distance between two clusterings across the lattice of possible clusterings.
3. The Global Consistency measure (GCE) [1] measures the extent to which one segmentation map can be viewed as a refinement of another segmentation. For a perfect match (in this metric sense) every region in one of the segmentation must be identical to, or a refinement (i.e., a subset) of a region in the other segmentation. Segmentation which are related in this manner are considered to be consistent, since they could represent the same natural image segmented at different levels of detail (as the segmented images produced by several human observers for which a finer level of detail will merge in such a way that they yield the larger regions proposed by a different observer at a coarser level).
4. The Boundary Displacement Error (BDE) [18] measures the average displacement error of one boundary pixels and the closest boundary pixels in the other segmentation.

4.3. Comparison With State-Of-The-Art Methods

Table 1 shows that the proposed algorithm gives competitive results in terms of PRI measure. This measure is highly correlated with human hand-segmentations and has a perceptual meaning since this performance measure is also a rate of good classification (a score equals to PRI$= 0.78$ simply means that on average 78% of pairs of pixels labels are correctly classified in the segmentation results). Table 1 also shows that the algorithm proposed in this paper is much more efficient than the Mean-Shift, NCut and FH in terms of BDE. Finally, it gives a good compromise between all these complementary performance measures, except in terms of VoI measure (certainly because our method rather favors oversegmentations). As also shown in [17], the use of many color

spaces (in our color-based textural measure) allows to noticeably improve the performance measures of our algorithm. Indeed, experiments with the same strategy but with only one color space (namely RGB) gives the following score : [PRI=0.7419, VoI=3.8315, GCE=0.3073, BDE=10.4570]. Our algorithm takes, on average, about 80 seconds for an AMD thurion 64 bits 1.5 GHz and non-optimized code running on Linux.

5. CONCLUSION

In this paper, we have presented a hierarchical Markovian segmentation method of textured color images. This technique is based on a grey-scale Markovian over-segmentation step, followed by an iterative Markovian graph-based clustering algorithm that uses a decreasing regularization parameter which aims at progressively merging neighboring regions with similar textural features. This method remains simple to implement and efficient for the difficult problem of textured color image segmentation. Besides, let us finally add that our procedure is also perfectible (e.g., by using an initial color over-segmentation or a more elaborate color-based textural measure) and also well suited for other kind of images since prior information on the size, the number, or the shape of the regions can be easily integrated in this segmentation procedure. The proposed method can be limited by the presence of camouflage in the image.

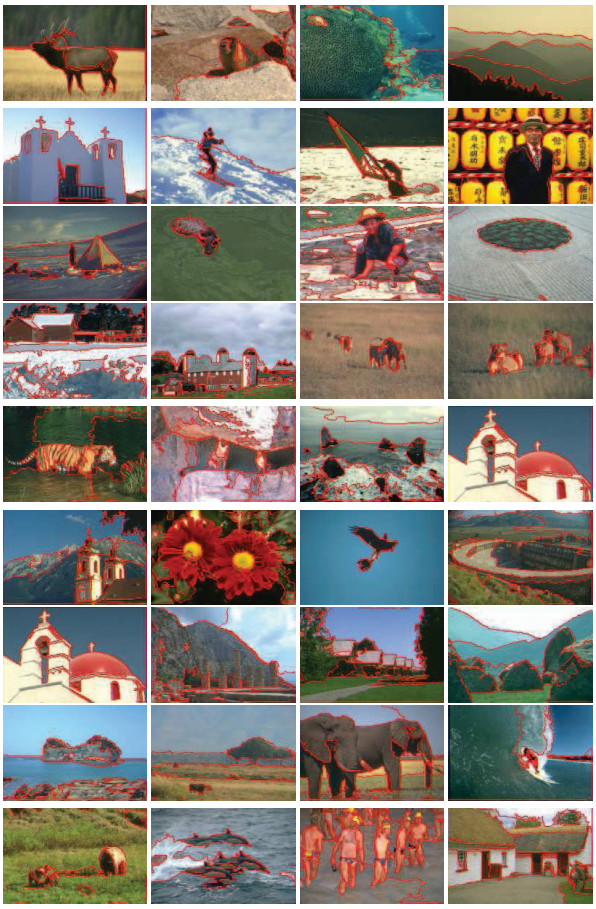

FIG. 2 – HMC segmentation results on some images of the Berkeley database.

6. REFERENCES

[1] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. on Computer Vision (ICCV)*, volume 2, pages 416–423, Vancouver, BC (Canada), July 2001.

[2] S. Banks. *Signal processing, image processing and pattern recognition*. Prentice Hall, 1990.

[3] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society*, B-48 :259–302, 1986.

[4] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Machine Intell.*, 22(8) :888–905, 2000.

[5] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2) :167–181, 2004.

[6] R. Gaetano, G. Scarpa, G. Poggi, and J. Zerubia. Unsupervised hierarchical image segmentation based on the TS-MRF model and fast mean-shift clustering. In *Proc. European Signal Processing Conference, (EUSIPCO)*, Lausanne (CH), August 2008.

[7] G. Scarpa, M. Haindl, and J. Zerubia. A hierarchical finite-state model for texture segmentation. In *Proc. 32nd IEEE Int'l Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, volume 1, pages 1209–1212, Honolulu, Hawai'i, (USA), April 2007.

[8] D. Comaniciu and P. Meer. Mean shift : A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Machine Intell.*, 24(5) :603–619, 2002.

[9] Q. Luo and T.M. Khoshgoftaar. Unsupervised multiscale color image segmentation based on MDL principle. *IEEE Trans. Image Processing*, 15(9) :2755–2761, 2006.

[10] S. Zhu and A. Yuille. Region competition : unifying snakes, region growing, and Bayes/MDL for multiband image segmentation. *IEEE Trans. Pattern Anal. Machine Intell.*, 18(9) :884–900, 1996.

[11] M. Mignotte. Image denoising by averaging of piecewise constant simulations of image partitions. *IEEE Trans. Image Processing*, 16(2) :523–533, 2007.

[12] A. P. Moore, S. Prince, J. Warrell, U. Mohammed, and G. Jones. Superpixel lattices. In *Proc. IEEE Int'l Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, Anchorage, Alaska (USA), June 2008.

[13] W. Pieczynski. Statistical image segmentation. *Machine Graphics and Vision*, 1(1/2) :261–268, 1992.

[14] S. Van Dongen. *Graph clustering by flow simulation*. PhD thesis, Ultrecht University, Netherland, May 2000.

[15] A. Y. Yang, J. Wright, Y. Ma, and S. Sastry. Unsupervised segmentation of natural images via lossy data compression. *Computer Vision and Image Understanding*, 110(2) :212–225, 2008.

[16] M. Krinidis and I. Pitas. Color texture segmentation based on the modal energy of deformable surfaces. *IEEE Trans. Image Processing*, 18(7) :1613–1622, 2009.

[17] M. Mignotte. Segmentation by fusion of histogram-based K-means clusters in different color spaces. *IEEE Trans. Image Processing*, 17(5) :780–787, 2008.

[18] J. Freixenet, X. Munoz, D. Raba, J. Marti, and X. Cufi. Yet another survey on image segmentation : Region and boundary information integration. In *Proc. 7th European Conf. on Computer Vision (ECCV)*, volume III, pages 408–422, Copenhagen, Denmark, June 2002.

[19] Y. Ma, H. Derksen, W. Hong, and J. Wright. Segmentation of multivariate mixed data via lossy coding and compression. *IEEE Trans. Pattern Anal. Machine Intell.*, 29(9) :1546–1562, 2007.

[20] R. Unnikrishnan, C. Pantofaru, and H. Martial. Toward objective evaluation of image segmentation algorithms. *IEEE Trans. Pattern Anal. Machine Intell.*, 29(6) :929–944, 2007.

[21] M. Meila. Comparing clusterings - An axiomatic view. In *Proc. 22nd Int'l Conf. on Machine Learning (ICML)*, volume 14, pages 577–584, Bonn, Germany, August 2005.