A time-consistent video segmentation algorithm designed for real-time implementation

Mohammed Elhassani, Delphine Rivasseau, Stéphanie Jehan-Besson, Marinette Revenu, David Tschumperlé, Luc Brun, Marc Duranton

A Time-Consistent Video Segmentation Algorithm designed for Real-Time Implementation

Mohammed El Hassani and Delphine Rivasseau
Philips Semiconductors
Caen, France

Stéphanie Jehan-Besson and Marinette Revenu
Laboratory GREYC
Caen University, France

David Tschumperlé and Luc Brun
Laboratory GREYC
Caen University, France

Marc Duranton
Philips Research
Eindhoven,Netherlands

***Abstract*— In this paper, we propose a time consistent video segmentation algorithm designed for real-time implementation. Our segmentation algorithm is based on a region merging process that combines both spatial and motion information. The spatial segmentation takes benefit of an adaptive decision rule and a specific order of merging. Our method has proven to be efficient for the segmentation of natural images (flat or textured regions) with few parameters to be set. Temporal consistency of the segmentation is ensured by incorporating motion information through the use of an improved change detection mask. This mask is designed using both illumination differences between frames, and region segmentation of the previous frame. By considering both pixel and region levels, we obtain a particularly efficient algorithm at a low computational cost, allowing its implementation in real-time on the TriMedia processor for CIF image sequences.**

I. INTRODUCTION

Segmentation of videos into homogeneous regions is an important issue for many video applications such as region-based motion estimation, image enhancement (since different processing may be applied for different regions), 2D to 3D conversion (segmentation can be used for depth estimation). These applications require two main features from segmentation: accuracy of regions boundaries in the spatial segmentation and temporal stability of the segmentation from frame to frame. Spatial segmentation can be classified into two main categories, namely contour-based and region-based methods. In the first category, edges are computed and connected components are extracted [1]. The main drawback of such an approach is that the computation of the gradient is prone to large errors especially in noisy images. Moreover we cannot take benefit of statistical properties of the considered image regions. The second category of methods, i.e. region-based, is less sensitive to noise. We are interested here in a bottom-up segmentation approach. In these methods, two important points must be considered: the first one is the order of merging and the second one is the similarity criterion, see for example [2], [3], [4]. When dealing with video segmentation, the temporal dimension must be added. Various approaches have been tested. Some authors consider video data as a volume [5], while others take benefit of motion information, such as scene change detection or motion field [6], [7]. Other works propose the object tracking [8] which is beyond the topic of this paper.

In this paper, we propose a spatial segmentation that takes benefit of both an adaptive decision rule and an original order of merging. This method gives good results for spatial segmentation with few parameters to be set. We use motion information to improve the temporal consistency. Motion estimation is a real bottleneck for real-time implementation, so we rather propose to combine an improved Change Detection Mask (CDM) with spatial segmentation in order to improve the temporal consistency of our segmentation. By making comparisons both at a pixel level and at a region level, we obtain an efficient algorithm for video segmentation at a low computational cost. Our method runs in real-time on the TriMedia processor for CIF image sequences. Moreover, experimental results conducted on real video sequences demonstrate a good temporal consistency.

The paper is organized as follows. The spatial segmentation method is detailed in section II. The temporal consistency improvement is detailed in section III. In section IV, we discuss the implementation of the algorithm. Experimental results and measures are given in section V.

II. SPATIAL SEGMENTATION

Let us consider an image I, the notation $|.|$ represents the cardinal and $I(p,n)$ the pixel intensity at position $p=(x,y)^T$ in the frame n.

A region-based segmentation problem aims at finding a relevant partition of the image domain in m regions $\{S_1, S_2, .., S_m\}$. The algorithm of segmentation detailed here uses an implicit encoding of the initial 4 connected planar sampling grid of pixels. It combines a specific order of fusion with an adaptive threshold. Both steps are detailed thereafter.

A. ORDER OF MERGING

The order of merging is built on the edges weights as in [9], [10]. The idea behind this order of merging is to merge first similar regions rather than different ones. An edge e denotes a couple of pixels (p,p') in a 4-connectivity scheme. The similarity between pixels is measured by computing the distance between two pixel colors as follows:

$$w(p,p',n)=\sqrt{\sum_{I\in\{Y,U,V\}}\left(I(p,n)-I(p',n)\right)^2}. \qquad (1)$$

For our algorithm, we consider the YUV color space, since this is the native color space of CIF sequences. The color space $(L^*a^*b^*)$ provides partitions with a little greater subjective quality but with a higher computational cost.

The edges are sorted in increasing order of their weights and corresponding couples of pixels are processed in this order for merging. As far as the implementation is concerned, the image is only scanned twice for this sorting: One time, in order to compute the number of edges with same weights and store this number in a table (edge weight histogram). Second, in order to store each edge in the memory part corresponding to its weight.

B. THE CRITERION OF MERGING

Given two regions S_1 and S_2, we want to know if these two regions have to be merged. A similarity criterion between the two regions must then be chosen and evaluated. Couple of adjacent regions will be considered as similar and merged if their merging criterion is below a given threshold. The choice of the corresponding threshold is often difficult. In this paper, we propose an adaptive threshold for a similarity criterion based on the means of the intensities of these two regions. This adaptive threshold is justified using statistical inequalities as in [9] but we here propose a simpler statistical interpretation of the image which leads to a more adapted criterion for real-time implementation. Such a predicate gives very good segmentation results as shown in Fig.2.

1) Merging predicate: The mean of the intensities of the region S_i is computed as follows:

$$\overline{S_i} = \frac{1}{|S_i|} \sum_{k=0}^{k=|S_i|} I_i(p_k, n)$$

where $I_i(p_k, n)$ is the intensity of the k^{th} pixel of region S_i. The merging predicate is then :

$$P(S_1, S_2) = \begin{cases} true & \text{if} \quad (\overline{S_1} - \overline{S_2})^2 \leq Qg^2(\frac{1}{|S_1|} + \frac{1}{|S_2|}) \\ false & \text{otherwise} \end{cases} \tag{2}$$

where g is the maximum level of I ($g = 255$ for grayscale images). The parameter Q allows to tune the coarseness of the segmentation. In the experiments, we choose $Q = 2$ which gives good results for CIF (Common Input Format $= 352 \times 288$) images.

2) Statistical justification of the predicate: Classically, the image I is considered to be an observation of a perfect image I^* and then pixel intensities are considered as observations of a vector of random variable (r.v.) noted $\mathbf{X} = (X_1, .., X_n)^T$. The merging predicate is based on the use of statistical inequalities and especially the McDiarmid inequality as in [9].

Theorem 2.1: (The independent bounded difference inequality [11]) Let $\mathbf{X} = (X_1, X_2, ..., X_n)$ be a family of n independent r.v. with X_k taking values in a set A_k for each k. Suppose that the real valued function f defined on $\prod_k A_k$ satisfies $|f(x) - f(x')| \leq c_k$ whenever vectors x and x' differ only in the k^{th} coordinate. Let $E(f(\mathbf{X}))$ be the expected value of the r.v. $f(\mathbf{X})$, then for any $\tau \geq 0$,

$$Pr(|f(\mathbf{X}) - E(f(\mathbf{X}))| > \tau) \leq 2\exp\left(-2\tau^2 / \sum_k c_k^2\right) \tag{3}$$

When testing the deviation between two adjacent regions S_1, S_2, we consider the following vector of random variables:

$$(X_1, ..., X_n) \quad = \quad \left(I_1^*(p_1), ..., I_1^*(p_{|S_1|}), I_2^*(p_1), ..., I_2^*(p_{|S_2|})\right)$$

where $I_i(p_j)$ is the intensity of the j^{th} pixel of S_i corresponding to the observation of the random variable $I_i^*(p_j)$. In this case, the size of the random vector is $n = |S_1| + |S_2|$. In order to apply Theorem 2.1, we then choose $f(\mathbf{x}) = \left(\overline{S_1} - \overline{S_2}\right)$. By inversion of the theorem, we have with a probability of at least $1 - \delta$ ($0 < \delta \leq 1$):

$$|(\overline{S_1} - \overline{S_2}) - E\left(\overline{S_1} - \overline{S_2}\right)| \leq g\sqrt{Q\left(\frac{1}{|S_1|} + \frac{1}{|S_2|}\right)}$$

where $Q = \frac{1}{2} * ln(\frac{2}{\delta})$, and $E(\overline{S})$ is the expected value of $\overline{S}$. If S_1 and S_2 belong to the same region in I^*, the expectation $E\left(\overline{S_1} - \overline{S_2}\right)$ will be null and the predicate follows.

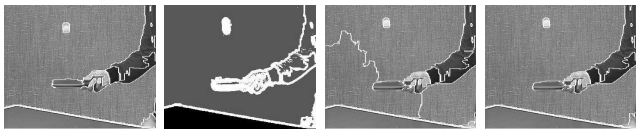
C. SEGMENTATION ALGORITHM

Our spatial segmentation could be divided in three steps. In the first one, we compute the weights of edges and their histogram. In the second step we sort edges in an increasing way of their weights. In the last step we merge pixels or regions connected by edges following their order. The implementation of this last step is given in section IV.

III. TIME CONSISTENCY IMPROVEMENT

In video segmentation, the quality of the spatial segmentation is not the only requirement, time consistency is also a very important one. If in two successive frames, one region is segmented very differently because of noise, occlusion or deocclusion, results of segmentation would be very difficult to exploit for any application like image enhancement, depth estimation and motion estimation. Many works, see for example [7], use motion estimation to improve time consistency in video segmentation. However, motion estimation is a real bottleneck for real-time implementation and is even sometimes unreliable. In this paper, we combine an improved Change Detection Mask (CDM) with spatial segmentation in order to improve the temporal consistency of our segmentation.

The CDM is designed using both illumination differences between frames and region segmentation of the previous frame. We first detect changing pixels using the frame difference. Then, we take benefit of the region segmentation of the previous frame in order to classify the pixels not only at a pixel level but also at a region level. Given the current frame $I(:, n)$ and the previous one $I(:, n-1)$, the frame difference FD is given by $FD(p) = |I(p, n) - I(p, n-1)|$. Classically, FD is thresholded in order to distinguish changing pixels from noise. A pixel p, with $L(p) = 1$ is denoted a changing pixel. We then use the previous segmentation in order to convert the CDM from the pixel level to a region level which is more reliable. For each region in the previous segmentation S_i, we compute $\tau(S_i)$ which represents the ratio of changing pixels in the region S_i. We have $\tau(S_i) = \frac{N_{i,changing}}{|S_i|}$ where $N_{i,changing}$ is the number of changing pixels in the region S_i. Pixels are then classified using three categories:



(a) (b) (c) (d)

Fig. 1. (a) Segmentation of a frame n-1. (b) CDM computed from the segmentation of the frame n-1 and frame difference between frame n-1 and frame n. (c) Segmentation of the frame n without CDM. (d) Segmentation of the frame n with CDM.

$$CDM(p) = \begin{cases} 0 & if\ (\tau(S_i) \leq tr_2). \\ 1 & if\ (\tau(S_i) > tr_2)\ and\ (L(p) = 1). \\ 2 & if\ (\tau(S_i) > tr_2)\ and\ (L(p) = 0). \end{cases} \quad (4)$$

where tr_2 is a positive constant. In the experiments, we take $tr_2 = 0.01$ (i.e. a region is a moving region when it contains at least 1% of moving pixels). The value of the threshold is chosen so that we don't miss any moving region.
Every pixel of regions qualified as static are labelled using $CDM(p) = 0$. The two other labels concern pixels within moving regions. Depending on the value of the frame difference, the pixel is qualified as a changing one ($CDM(p) = 1$) or a no changing one ($CDM(p) = 2$). Such a classification is then used to segment the current frame. Firstly, static regions are kept as they were segmented in the previous frame. Secondly we apply a connected component labelling (CCL) algorithm [12] to extract connected component of pixels with $CDM(p) = 2$. This second step builds seeds from the segmentation of the previous frame. These seeds link the current segmentation to the previous one in a time consistent way. Thirdly, we apply the spatial segmentation only on edges (p, p') connecting changing pixels, and those connecting changing pixels with seeds built in the second step.

We show here an example of segmentation results with and without time consistency in Fig.1(d) and 1(c). In Fig.1(b), the different labels (i.e. values of CDM) are given using three values of intensity (black for $CDM(p) = 0$, gray for $CDM(p) = 2$ and white for $CDM(p) = 1$).

In section V, we propose the computation of an objective measure for temporal consistency. The results show a real improvement for time consistency for different sequences. Moreover, the way we exploit the CDM decreases also the amount of computation of the algorithm since the edges in static area are not reconsidered, and those linking no changing pixels in moving area are simply processed by a CCL algorithm.

IV. Implementation considerations

In this section, we propose to describe optimizations that have been made to allow a real time treatment. The whole algorithm of video segmentation is summarized in figure 3. The algorithm 1 describes more particularly the merging loop which is the critical part for a real time implementation. The term N_e represents the number of edges within the image I in the 4-connectivity. In the merging process, we use the UNION-FIND data structure [13]. The UNION function fuses two disjoint regions into one region, and the FIND function identifies the regions a certain pixel belongs to.

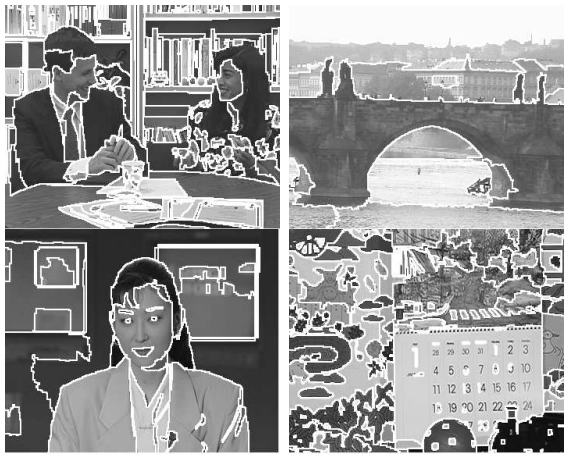


Fig. 2. Results of segmentation of frames from the sequences $Paris$, $Bridge$, $Akiyo$ and $Mobile$.

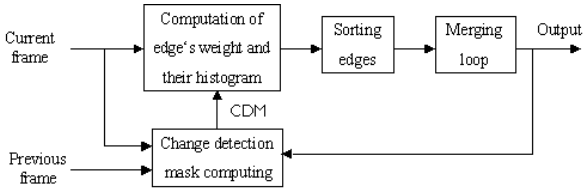


Fig. 3. The general diagram of video segmentation

Apart from this merging loop, all other functions access pixels data in a predictable way (for example from top to bottom left to right). The cache memory benefits from this regularity, since it exploits spatial and temporal locality of data, and consequently causes less cache misses. In the merging loop, the UNION-FIND data structure is unpredictable, and consequently causes an important data cache stalls. To reduce the data cache stalls cycles, we investigate two optimizations, which are independent of the cache memory type (size, mapping, block size, replacement algorithm...).

The first optimization concerns the organization of the data. Each pixel in the image could be a representative of one region so the data are stored with an array A of size $|I|$, where each element $A(p)$ stores the mean colors $(\overline{Y}, \overline{U}, \overline{V})$ and the cardinal $|S|$ of the region whose representative is p.

Algorithm 1 The merging loop

for $i := 1$ to N_e **do**
  Read the i^{th} edge: (p_1, p_2);
  $S_1 = FIND(p_1)$;
  $S_2 = FIND(p_2)$;
  if $P(S_1, S_2) = True$ **then**
    $UNION(S_1, S_2)$
  end if
end for

We verify that this approach reduces the number of data cache stalls. The second optimization concerns the UNION-FIND data structure. It is encoded separately thanks to an array F of size $|I|$, where $F(p)$ stores the address of the father of p. Each region is thus encoded by a tree using such father-child relationships. The alternative solution, which consists in storing both the labels and the data in the same structure, would induce the storage of useless data within the cache during the FIND operations.

We experimented these optimizations on the TriMedia processor [14]. The cache memory of this particular TriMedia is 128 $KByte$, 4 way associative, with block of 128 $Byte$. The replacement algorithm used is LRU. We get a reduction of 3 $MCycles$ per frame. Notice that these optimizations are useful for many segmentation algorithms that use the UNION-FIND data structure.

There is an important amount of DLP (data level parallelism) in our algorithm (computation of edge's weight, frame difference, classification of pixels in CDM). This allows to increase the throughput (i.e. amount of pixels processed per unit time), by processing data in parallel when it is possible. The core of TriMedia is a VLIW architecture with 5 issues slots. Each slot has some functional unit, and each functional unit could process 4 bytes in parallel (SIMD mode). The ILP (Instruction Level Parallelism) is extracted by the compiler, while the DLP could be exploited through the use of custom operations, loop unrolling, and grafting. So we use these optimizations to exploit the DLP available in our algorithm.

V. Experimental Results

In this section we present experimental results of our algorithm run on TriMedia with many very known CIF sequences. The table I shows the following measures:

1) We use a classical measure to evaluate time consistency. Given the segmentation of previous frame $SEG(n-1)$ and the segmentation of the current one $SEG(n)$, we find the correspondence between regions in $SEG(n-1)$ and $SEG(n)$. Two regions ($S_{i,n-1} \in SEG(n-1)$ and $S_{j,n} \in SEG(n)$ correspond if they have the most overlapping area $Overlap(i,j) = |S_{i,n-1} \cap S_{j,n}|$. We sum the number of pixels in the overlapping areas of corresponding regions over the image. The consistency measure is the percentage of this number to the size of the image.
2) We give the number of Mcycles the algorithm takes on TriMedia. The exploitation of the CDM reduces the computational cost. This reduction depends on the correlation between two successive frames. Since the execution of spatial segmentation without CDM is the critical case of our algorithm, we show the number of Mcycles corresponding to this critical case.

When enforcing consistency through the CDM, time consistency is higher, and visually, segmentation is more stable from frame to frame and still fit very well regions boundaries as shown in Fig.2. With a 450 MHz TriMedia, we are able to process more than 25 frames per second, which is sufficient for real-time execution.

TABLE I
Experimental Measures of our algorithm

Sequence	Akiyo	Tennis Table	Paris	Mobile
Time Consistency (without CDM)	0.88	0.73	0.89	0.84
Time Consistency (with CDM)	0.98	0.92	0.97	0.92
Mcycles per frame (without CDM)	15.68	15.84	16.59	16.20

VI. Conclusion

Designing usable algorithms for video processing requires low computational methods. Directed by this constraint, we propose here an efficient time consistent algorithm for video segmentation. This method gives accurate results with high temporal consistency. It is based on a statistical spatial segmentation and on the computation of an improved change detection mask. It runs in real-time on only one processor. Experimental results demonstrate the applicability of this method.

Acknowledgement

The authors would like to thank Patrick Meuwissen, O.P. Gangwal and Zbigniew Chamski for their constructive suggestions.

References

[1] G. Iannizzotto and L. Vita, "Fast and accurate edge-based segmentation with no contour smoothing in 2-d real images," *IEEE Transactions on Image Processing*, vol. 9, Issue 7, pp. 1232 – 1237, 2000.

[2] L. Vincent and P. Soille, "Watersheds in digital spaces: an efficient algorithm based on immersion simulations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, Issue 6, pp. 583–598, 1991.

[3] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, Issue 8, pp. 888 – 905, 2000.

[4] E. Sharon, A. Brandt, and R. Basri, "Fast multiscale image segmentation," *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 70 – 77, 2000.

[5] H.-Y. Wang and K.-K. Ma, "Automatic video object segmentation via 3d structure tensor," *International Conference on Image Processing, ICIP*, vol. 1, 14-17, pp. 153–156, 2003.

[6] Y. Deng and B. Manjunath, "Unsupervised segmentation of color-texture regions in images and video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, Issue 8, pp. 800 – 810, 2001.

[7] D. Wang, "Unsupervised video segmentation based on watersheds and temporal tracking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, ISSUE 5, pp. 539 – 546, 1998.

[8] F. Moscheni, S. Bhattacharjee, and M. Kunt, "Spatio-temporal segmentation based on region merging," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, Issue 9,, pp. 897 – 915, 1998.

[9] R. Nock and F. Nielsen, "Statistical region merging," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, 2004.

[10] P. Felzenszwalb and D. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision*, vol. 59, Issue 2, pp. 167–181, 2004.

[11] M. Habib, C. McDiarmid, J. Ramirez-Alfonsin, and B. Reed, "Probabilistic methods for algorithmic discrete math," *Springer Verlag*, vol. 20, Issue 9,, pp. 1 – 54, 1998.

[12] K. Wu, E. Otoo, and A. Shoshani, "Optimizing connected component labeling algorithms," *Proceedings of SPIE*, vol. 5747, pp. 1965–1976, 2005.

[13] C. Fiorio and J. Gustedt, "Two linear time union-find strategies for image processing," *Theoretical Computer Science*, vol. 154, pp. 165–181, 1996.

[14] "pnx1500 databook," *available at http://www.tcshelp.com/public_files.html*.