CAUSAL GRAPH-BASED VIDEO SEGMENTATION

Camille Couprie *

IFP Energies nouvelles
92500 Rueil Malmaison, France

Clement Farabet, Yann LeCun

CIMS, New York University,
New York, USA

Laurent Najman

Université Paris Est, LIGM
ESIEE, Noisy-le-Grand, France

ABSTRACT

Among the different methods producing superpixel segmentations of an image, the graph-based approach of Felzenszwalb and Huttenlocher is broadly employed. One of its interesting properties is that the regions are computed in a greedy manner in quasi-linear time by using a minimum spanning tree. The algorithm may be trivially extended to video segmentation by considering a video as a 3D volume, however, this can not be the case for causal segmentation, when subsequent frames are unknown. In a framework exploiting minimum spanning trees all along, we propose an efficient video segmentation approach that computes temporally consistent pixels in a causal manner, filling the need for causal and real time applications.

***Index Terms*—** Optimization, superpixels, graph-matching

1. INTRODUCTION

A segmentation of video into consistent spatio-temporal segments is a largely unsolved problem. While there have been attempts at video segmentation, most methods are non causal and non real-time. This paper proposes a fast method for real time video segmentation, including semantic segmentation as an application.

An large number of approaches in computer vision makes use of superpixels at some point in the process. For example, semantic segmentation [6], geometric context identification [11], extraction of support relations between object in scenes [15], etc. Among the most popular approach for superpixel segmentation, two types of methods are distinguishable. Regular shape superpixels may be produced using normalized cuts or graph cuts[17, 18] for instance. More object – or part of object – shaped superpixels can be generated from watershed based approaches. In particular the method of Felzenswalb and Huttenlocher [7] produces such results.

It is a real challenge to obtain a decent delineation of objects from a single image. When it comes to real-time data analysis, the problem is even more difficult. However, additional cues can be used to constrain the solution to be temporally consistent, thus helping to achieve better results. Since many of the underlying algorithms are in general super-linear, there is often a need to reduce the dimensionality of the video. To this end, developing low level vision methods for video segmentation is necessary. Currently, most video processing approaches are non-causal, that is to say, they make use of future frames to segment a given frame, sometimes requiring the entire video [10]. This prevents their use for real-time applications.

Some approaches have been designed to address the causal video segmentation problem [16, 14]. [16] makes use of the mean shift method [3]. As this method works in a feature space, it does not necessary cluster spatially consistent superpixels. A more recent approach, specifically applied for semantic segmentation, is the one of Miksik *et al.* [14]. The work of [14] is employing an optical flow method to enforce the temporal consistency of the semantic segmentation. Our approach is different because it aims to produce superpixels, and possibly uses the produced superpixels for smoothing semantic segmentation results. Furthermore, we do not use any optical flow pre-computation that would prevent us having real time performances on a CPU.

Some works use the idea of enforcing some consistency between different segmentations [8, 13, 12, 9, 19]. [8] formulates a co-clustering problem as a Quadratic Semi-Assignment Problem. However solving the problem for a pair of images takes about a minute. Alternatively, [13] and [9] identify the corresponding regions using graph matching techniques. [19] proposes like us to exploit Felzenszwalb *et al.* superpixels in causal video processing. The complexity of this approach is super-linear because of a hierarchical segmentation, preventing with the current implementation real time applications.

The idea developed in this paper is to perform independent segmentations and match the produced superpixels to define markers. The markers are then used to produce the final segmentation by minimizing a global criterion defined on the image. We show how Minimum Spanning Trees can be used at every step of the process, leading to gains in speed, and real-time performances on a single core CPU.

*Performed most of the work while at New York University

2. METHOD

Given a segmentation S_t of an image at time t, we wish to compute a segmentation S_{t+1} of the image at time $t+1$ which is consistent with the segments of the result at time t.

2.1. Independent image segmentation

The superpixels produced by [7] have been shown to satisfy the global properties of being not too coarse and not too fine according to a particular region comparison function. In order to generate superpixels close to the ones produced by [7], we first generate independent segmentations of the 2D images using [7]. We name these segmentations $S'_1, ..., S'_t$. The principle of segmentation is fairly simple. We define a graph G_t, where the nodes correspond to the image pixels, and the edges link neighboring nodes in 8-connectivity. The edge weights ω_{ij} between nodes i and j are given by a color gradient of the image.

A Minimum Spanning Tree (MST) is build on G_t, and regions are merged according to a criterion taking into account the regions sizes and a scale parameter k.

Once an image is independently segmented, resulting in S'_{t+1}, we then face the question of the propagation of the temporal consistency given the non overlapping contours of S_t and S'_{t+1}.

Our solution is the development of a cheap graph matching technique to obtain correspondences between segments from S_t and these of S'_{t+1}. This first step is described in Section 2.2. We then mine these correspondences to create markers (also called seeds) to compute the final labeling S_{t+1} by solving a global optimization problem. This second step is detailed in Section 2.3.

2.2. Graph matching procedure

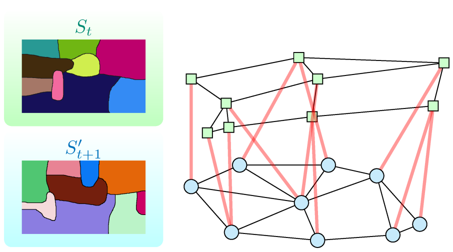


Fig. 1. Illustration of the graph matching procedure

The basic idea is to use the segmentation S_t and segmentation S'_{t+1} to produce markers before a final segmentation of image at time $t+1$. Therefore, in the process of computing a new segmentation S_{t+1}, a graph G is defined. The vertices of G comprises to two sets of vertices: V_t that corresponds to the set of regions of S_t and V'_{t+1} that corresponds to the set of regions of S'_{t+1}. Edges link regions characterised by a small distance between their centroids. The edges weights between vertex $i \in V_t$ and $j \in V'_{t+1}$ are given by a similarity measure taking into account distance and differences between shape and appearance

$$w_{ij} = \frac{(|r_i| + |r_j|)d(c_i, c_j)}{|r_i \cap r_j|} + a_{ij}, \tag{1}$$

where $|r_i|$ denotes the number of pixels of region r_i, $|r_i \cap r_j|$ the number of pixels present in r_i and r_j with aligned centroids, and a_{ij} the appearance difference of regions r_i and r_j. In our experiments a_{ij} was defined as the difference between mean color intensities of the regions.

The graph matching procedure is illustrated in Figure 1 and produces the following result: For each region of S'_{t+1}, its best corresponding region in image S_t is identified. More specifically, each node i of V_t is associated with the node j of V'_{t+1} which minimizes w_{ij}. Symmetrically, for each region of S_t, its best corresponding region in image S'_{t+1} is identified, that is to say each node i of V'_{t+1} is associated with the node j of V_t which minimizes w_{ij}. This step may also be viewed as the construction of two minimum spanning trees, one spanning V_t, and the other V_{t+1}.

2.3. Final segmentation procedure

The final segmentation S_{t+1} is computed using a minimum spanning forest procedure. This seeded segmentation algorithm that produces watershed cuts [5] is strongly linked to global energy optimization methods such as graph-cuts [2, 4] as detailed in Section 2.4. In addition to theoretical guaranties of optimality, this choice of algorithm is motivated by the opportunity to reuse the sorting of edges that is performed in 2.1 and constitutes the main computational effort. Consequently, we reuse here the graph $G_{t+1}(V, E)$ built for the production of independent segmentation S'_{t+1}.

The minimum spanning forest algorithm is recalled in [20]. The seeds, or markers, are defined using the regions correspondences computed in the previous section, according to the procedure detailed below. For each segment s' of S'_{t+1} four cases may appear:

1. s' has one and only one matching region s in S_t: propagate the label l_s of region s. All nodes of s' are labeled with the label l_s of region s.

2. s' has several corresponding regions $s_1, ..., s_r$: propagate seeds from S_t. The coordinates of regions $s_1, ..., s_r$ are centered on region s'. The labels of regions $s_1, ..., s_r$ whose coordinates are in the range of s' are propagated to the nodes of s'.

3. s' has no matching region : The region is labeled by the label l'_s itself.

4. If none of the previous cases is fulfilled, it means that s' is part of a larger region s in S_t. If the size of s' is small, a new label is created. Otherwise, the label l_s is propagated in s' as in case 1.

Before applying the minimum spanning forest algorithm, a safety test is performed to check that the map of produced markers does not differ two much from the segmentation S'_{t+1}. If the test shows large differences, an eroded map of S'_{t+1} is used to correct the markers.

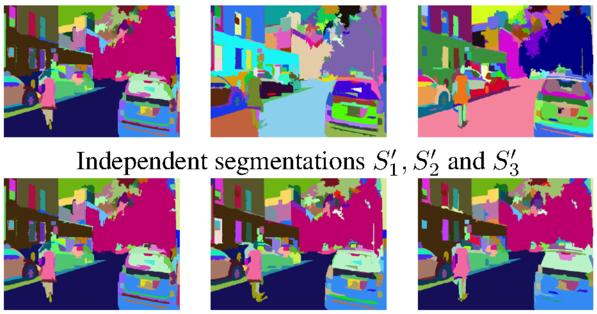


Fig. 2. Segmentation results on 3 consecutive frames of the NYU-Scene dataset.

2.4. Global optimization guaranties

Several graph-based segmentation problems, including minimum spanning forests, graph cuts, random walks and shortest paths have recently been shown to belong to a common energy minimization framework [4]. The considered problem is to find a labeling $x^* \in \mathbb{R}^{|V|}$ defined on the nodes of a graph that minimizes

$$E(x) = \sum_{e_{ij} \in E} w_{ij}^p |x_j - x_i|^q + \sum_{v_i \in V} w_i^p |l_i - x_i|^q, \qquad (2)$$

where l represents a given configuration and x represents the target configuration. The result of $\lim_{p \to \infty} \arg\min_x E(x)$ for values of $q \geq 1$ always produces a cut by maximum (equivalently minimum) spanning forest. The reciprocal is also true if the weights of the graph are all different.

In the case of our application, the pairwise weights w_{ij} is given by an inverse function of the original weights ω_{ij}. The pairwise term thus penalizes any unwanted high-frequency content in x and essentially forces x to vary smoothly within an object, while allowing large changes across the object boundaries. The second term enforces fidelity of x to a specified configuration l, w_i being the unary weights enforcing that fidelity.

The enforcement of markers l_s as hard constrained may be viewed as follows: A node of label l_s is added to the graph, and linked to all nodes i of V that are supposed to be marked. The unary weights ω_{i,l_s} are set to arbitrary large values in order to impose the markers.

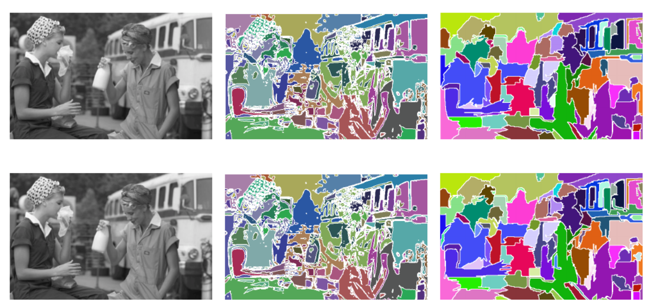
2.5. Applications to optical flow and semantic segmentation

An optical flow map may be easily estimated from two successive segmentations S_t and S_{t+1}. For each region r of S_{t+1}, if the label of r comes from a label present in a region s of the segmentation S_t, the optical flow in r is computed as the distance between the centroid of r and the centroid of s. The optical flow map may be used as a sanity check for region tracking applications. By principle, a video sequence will not contain displacements of objects greater than a certain value.

For each superpixel s of S_{t+1}, if the label of region s comes from the previous segmentation S_t, then the semantic prediction from S_t is propagated to S_{t+1}. Otherwise, in case the label of s is a new label, the semantic prediction is computed using the prediction at time $t+1$. As some errors may appear in the regions tracking, labels of regions having inconsistent large values in optical flow maps are not propagated. For the specific task of semantic segmentation, results can be improved by exploiting the contours of the recognized objects. Semantic contours such as for example transition between a building and a tree for instance, might not be present in the gradient of the raw image. Thus, in addition to the pairwise weights ω described in Section 2.1, we add a constant in the presence of a semantic contour.

3. RESULTS

We now demonstrate the efficiency and versatility of our approach by applying it to simple superpixel segmentation and semantic scene labeling.



Original frames | Mean shift results [16] | Our results

Fig. 3. Comparison with the mean-shift segmentation method of Paris [16] on Frame 19 and 20. $k = 200, \delta = 400, \sigma = 0.5$.

Following the implementation of [7], we pre-process the images using a Gaussian filtering step with a kernel of variance σ is employed. A post-processing step that removes regions of small size, that is to say below a threshold δ is also performed. As in [7], we denote the scale of observation parameter by k.

3.1. Superpixel segmentation

Experiments are performed on two different types of videos: videos where the camera is static, and videos where the camera is moving. The robustness of our approach to large variations in the region sizes and large movements of camera is illustrated on Figure 2.

A comparison with the temporal mean shift segmentation of Paris [16] is displayed at Figure 3. The superpixels produced by the [16] are not spatially consistent as the segmentation is performed in the feature (color) space in their case. Our approach is slower, although qualified for real-time application, but computes only spatially consistent superpixels.

(a) Independent segmentations with no temporal smoothing

(b) Result using the temporal smoothing method of [14]

(c) Our temporally consistent segmentation

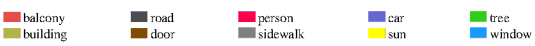


Fig. 4. Comparison with the temporal smoothing method of [14]. Parameters used: $k = 1200, \delta = 100, \sigma = 1.2$.

3.2. Semantic scene labeling

We suppose that we are given a noisy semantic labeling for each frame. In this work we used the semantic predictions of [6] and [21].

On the NYU Depth dataset [15], we compare independent segmentation performances with our temporally smoothed results on four video sequences of indoor scenes. Unless specified, the same choice of parameters was performed in all our comparisons. We also compare our results with the results of [14] on the NYU-Scene dataset. The dataset consists in a video sequence of 73 frames provided with a dense semantic labeling and ground truths. The provided dense labeling being performed with no temporal exploitation, it suffers from sudden large object appearances and disappearances. As illustrated in Figure 4 our approach reduces this effect, and improves the classification performance of more than 5% as reported in Table 2.

3.3. Computation time

The experiments were performed on a laptop with 2.3 GHz Intel core i7-3615QM. Our method is implemented on CPU

	Frame by frame	Our method
dining room	**63.8**	58.5
living room	65.4	**72.1**
classroom	56.5	**58.3**
office	56.3	**57.4**
mean	60.5	**61.6**

Table 1. Overall pixel accuracy (%) for the semantic segmentation task on the NYU Depth dataset. Parameter used: $\delta = 100, \sigma = 1.2, k = 800, 1000, 1000, 920$.

	Frame by frame	Miksik et al.[14]	Our method
Accuracy	71.11	75.31	**76.27**
#Frames/sec		1.33*	**10.5**

Table 2. Overall pixel accuracy (%) for the semantic segmentation task on the NYU Scene video. *Note that the reported timing does not take into account the optical flow computation needed by [14].

only, in C/C++, and makes use of only one core of the processor. Superpixel segmentations take 0.1 seconds per image of size 320×240 and 0.4 seconds per image of size 640×380, thus demonstrating the scalability of the pipeline. All computations are included in the reported timings. The mean segmentation time using [19] for a frame of size 320×240 is 4 seconds. The timings of the temporal smoothing method of Miksik *et al.*[14] are reported in Table 2. We note that the processor used for the reported timings of [14] has similar characteristics as ours. Furthermore, Mistik *et al.* use an optical flow procedure that takes only 0.02 seconds per frame when implemented on GPU, but takes seconds on CPU. Our approach is thus more adapted to real time applications for instance on embedded devices where a GPU is often not available. The code, as well as some data and video are available at [22].

4. CONCLUSION

The proposed approach demonstrates the ability of minimum spanning trees to fulfill accuracy and competitive timing requirements in a global optimization framework. Unlike many video segmentation techniques, our algorithm is causal and does not require any computation of optical flow. Our experiments on challenging videos show that the obtained superpixels are robust to large camera or objects displacement. Their use in semantic segmentation applications demonstrate that significant gains can be achieved and lead to state-of-the-art results. Furthermore, by being 8 times faster than the competing methods for temporal smoothing of semantic segmentation, and up to 25 times faster if the use of GPU is not available, the proposed approach has by itself a practical interest.

5. REFERENCES

[1] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus, "Indoor segmentation and support inference from RGBD images," in *ECCV*, 2012.

[2] C. Allène, J.-Y. Audibert, M. Couprie, and R. Keriven. Some links between extremum spanning forests, watersheds and min-cuts. *Image and Vision Computing*, 28(10):1460–1471, 2010.

[3] D. Comaniciu, P. Meer, and S. Member. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:603–619, 2002.

[4] C. Couprie, L. J. Grady, L. Najman, and H. Talbot. Power watershed: A unifying graph-based optimization framework. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 33(7):1384–1399, 2011.

[5] J. Cousty, G. Bertrand, L. Najman, and M. Couprie. Watershed cuts: Minimum spanning forests and the drop of water principle. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 31(8):1362–1374, 2009.

[6] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2013. in press.

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

[8] D. Glasner, S. N. Vitaladevuni, and R. Basri. Contour-based joint clustering of multiple segmentations. In *Proc. of IEEE Computer Vision and Pattern Recognition (CVPR 2011)*, pages 2385–2392, Washington, DC, USA, 2011.

[9] C. Gomila and F. Meyer. Graph-based object tracking. In *Proc. of IEEE International Conference on Image Processing (ICIP 2003)*, volume 2, pages II – 41–4 vol.3, sept. 2003.

[10] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph-based video segmentation. In *Proc. of IEEE Computer Vision and Pattern Recognition (CVPR 2010)*, 2010.

[11] D. Hoiem, A. Efros, and M. Hebert. Geometric context from a single image. In *IEEE International Conference on Computer Vision (ICCV 2005)*, volume 1, pages 654 – 661 Vol. 1, oct. 2005.

[12] A. Joulin, F. Bach, and J. Ponce. Multi-class cosegmentation. In *Proc. of IEEE Computer Vision and Pattern Recognition (CVPR 2012), Providence, RI, USA*, pages 542–549, 2012.

[13] J. Lee, J.-H. Oh, and S. Hwang. Clustering of video objects by graph matching. In *Proc. of the IEEE International Conference on Multimedia and Expo, ICME 2005, July 6-9, 2005, Amsterdam, The Netherlands*, pages 394–397, 2005.

[14] O. Miksik, D. Munoz, J. A. D. Bagnell, and M. Hebert. Efficient temporal consistency for streaming video scene analysis. Technical Report CMU-RI-TR-12-30, Robotics Institute, Pittsburgh, PA, September 2012.

[15] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *Proc. of IEEE European Conference on Computer Vision (ECCV 2012)*, 2012.

[16] S. Paris. Edge-preserving smoothing and mean-shift segmentation of video streams. In *Proc. of IEEE European Conference on Computer Vision (ECCV 2008), Marseille, France*, pages 460–473, 2008.

[17] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 1997.

[18] Olga Veksler, Yuri Boykov, and Paria Mehrani, "Superpixels and supervoxels in an energy optimization framework," in *Proc. of 11th European Conference on Computer Vision (ECCV'10), Heraklion, Crete, Greece, September 5-11, Part V*, 2010, pp. 211–224.

[19] Chenliang Xu, Caiming Xiong, and Jason J. Corso, "Streaming hierarchical video segmentation," in *Proc. of ECCV 2012 - 12th European Conference on Computer Vision, Florence, Italy, October 7-13, Part VI*, 2012, pp. 626–639.

[20] Camille Couprie, Clément Farabet, and Yann LeCun, "Causal graph-based video segmentation," *CoRR*, vol. abs/1301.1671, 2013.

[21] Camille Couprie, Clément Farabet, Laurent Najman, and Yann LeCun, "Indoor semantic segmentation using depth information," 2013.

[22] Camille Couprie, Source code for Causal graph-based video segmentation, www.esiee.fr/∼coupriec/code.html, 2013.