Streaming graph-based hierarchical video segmentation by a simple label propagation

Kleber J. F. de Souza,
Arnaldo de A. Araújo
NPDI Lab - DCC
UFMG
Belo Horizonte, Brazil
Email: {kleberjfsouza, arnaldo}@dcc.ufmg.br

Silvio J. F. Guimarães,
Zenilton K. G. do Patrocínio Jr.
VIPLAB - ICEI
PUC Minas
Belo Horizonte, Brazil
Email: {sjamil, zenilton}@pucminas.br

Matthieu Cord
LIP6
UPMC - Sorbonne Universités
Paris, France
Email: matthieu.cord@lip6.fr

Fig. 1. An example of streaming graph-based hierarchical video segmentation obtained by our proposed method.

***Abstract*—In this paper, we present an approach to streaming graph-based hierarchical video segmentation by simple label propagation. Here, we transform the streaming video segmentation into a graph partitioning problem in which each part corresponds to one region of the video; furthermore, we apply a simple method for merging the segmentations of two consecutive blocks to achieve the temporal coherence. The spatial-temporal coherence is given, only, by color information instead of more complex features. We provide an extensive comparative analysis among our method and methods in the literature showing efficiency, ease of use, and temporal coherence of ours. According to the experiments, our method produces good results when applied to video segmentation besides presenting a low space and time cost, compared to other methods.**

***Keywords*-Streaming video segmentation; Observation scale; Graph-based segmentation; Label propagation;**

I. Introduction

The interpretation of the data in a video is a complex activity, so a step of video segmentation may be necessary to partition the data set into structures with relevant semantic content to aid in the analysis process. We can find in the literature several algorithms of video segmentation, which mostly are extensions of image segmentation techniques. Some of these algorithms simply apply image techniques segmentation to the video frames without considering temporal coherence [1], [2], others can preserve the temporal information as supervoxels, which is a set of spatially contiguous voxels (a voxel has three coordinates (x; y; t), in which time is represented as the third dimension) that have similar appearance (intensity, color, texture, etc.) [3], [4], [5], [6], [7], [8], [9].

The methods based on Nystrom [4] and segmentation by weighted aggregation (so-called SWA, Segmentation by Weighted Aggregation) [5], both optimize the same normalized cut criterion. In [4], the Nystrom approximation was applied to short videos with low-resolution, while the approach based on SWA proposed by [5] computes iteratively the hierarchy. In [3], the authors introduced the Morse theory to interpret mean shift as a topological decomposition of the feature space, the algorithm is applied to temporal sequences (so-called MeanShift). For video segmentation, the method proposed by [10] (so-called GBH, Graph-Based Hierarchical) taking into account the same criterion of [11] (so-called GB, Graph-Based) iteratively computes different hierarchical levels, using an adjacency region graph. Even though this method presents high quality segmentations with a good temporal coherence and with stable region boundaries, for computing a video segmentation according to a specified level, it is necessary to compute all lower (finer) segmentations.

In [8], [9], the authors proposed another hierarchical segmentation method – HOScale (Hierarchical video segmentation using an Observation Scale) – based on the same criterion

of [11] that removes the need for parameter tuning and for the computation of video segmentation at finer levels. In their proposal, the video segmentation is not dependent on the hierarchical level, and consequently, it is possible to compute any level without computing the previous ones, thus the time for computing a segmentation is almost the same for any specified level. Moreover, according to experimental results presented in [9], HOScale produces good quantitative and qualitative results when compared to other methods.

Although these methods have presented good results for segmenting videos, as shown in [9], [12], there are still some factors that prevent their use in real applications, *i.e.*, memory consumption and processing time make unfeasible to apply those methods to medium and large videos. In [7], the authors proposed the first streaming hierarchical video segmentation method – StreamGBH. The experimental results indicate that StreamGBH outperforms other streaming video segmentation methods and performs nearly as well as the full-video hierarchical graph-based method GBH. But, since it adopts GBH [10] as a component, it also has the same limitations, *i.e.*, it is necessary to compute all lower (finer) segmentations.

This paper presents a new graph-based streaming segmentation method, that represents an evolution of the method presented in [8], [9] – StreamHOScale – in which the original video is divided in frame blocks and the streaming video segmentation is transformed into a graph partitioning problem for each frame block. Furthermore, a simple method is adopted to merge the segmentations generated for each block propagating the spatio-temporal information simply and efficiently in order to achieve the temporal coherence. The great advantage of StreamHOScale method is the ability to run a video stream without the need of having all the video in memory, achieving to segment of consecutive frames blocks considering the temporal information present throughout the video. Experimental results have shown the effectiveness of our method.

This work is organized as follows. In Section II, our approach for streaming video segmentation is presented along with simple examples to better explain how it really works. Then, experimental results are presented in Section III. Finally, Section IV presents final remarks and discusses possible research lines for future works.

II. Streaming video segmentation

Although some video segmentation methods have presented good results for segmenting videos, as shown in [9], [12], the memory consumption and processing time are prohibitive issues for their use in real applications, mainly due to huge quantity of video information to be processed. Thus, instead of considering all video information, we divide the video into k-sized frame blocks in order to cope with those problems without losing the qualitative performance for the segmentations. So, in this work, we propose a graph-based streaming video segmentation, so called StreamHOScale in which we apply the method proposed in [9] to segment each k-sized frame block, followed by a new and simple strategy for merging the segmentations of two consecutive blocks.

As can be seen in Fig. 2, our method can be outlined in some steps: (1) creation of a video graph for each k-sized frame block; (2) computation of hierarchical scales for each video graph; (3) identification of video segments for each block; (4) computation of temporal coherence between video segments belonging to consecutive blocks; and (5) creation of a segmented video.

For each k-sized frame block, the underlying graph composed by both the 26-adjacency pixel relationship and the 10 nearest neighbors in $RGBXYZ$ space is created. The edges are weighted by a simple color gradient computed by the Euclidean distance in the RGB space. For each video graph, we compute (hierarchical) observation scales between any two adjacent regions using the same graph-based hierarchical method proposed in [9], *i.e.*, instead of computing these scales directly from the k-sized frame block, our approach computes them on a graph generated from this block. Note that the modeling used to transform the video into an edge-weighted graph may influence the calculation of the scales (Step 1 in Fig. 2), and, consequently, it may modify the results that will be obtained by our method. In this section, we focus the discussion over the process for calculating the hierarchical scales from video graph (Step 2 in Fig. 2) and for computing temporal coherence (Step 4 in Fig. 2).

Thanks to the method proposed in [9], we compute the hierarchical observation scales using the method called *cp*-HOScale (or simply HOScale), in which the adjacent regions that are evaluated depend on the order of the merging in the fusion tree (or simply, the order of the merging between connected components on the minimum spanning tree – MST – of the original graph). Generally speaking, a new edge-weighted tree is created from this MST in which each edge weight corresponds to the scale from which two adjacent regions connected by this edge are correctly merged, *i.e.*, there are no two sub-regions of these regions that might be merged before these regions. For computing the new weight map, we consider the criterion for region-merging proposed in [11] which measures the evidence for a boundary between two regions by comparing two quantities: one based on intensity differences across the boundary, and the other based on intensity differences between neighboring pixels within each region. More precisely, in order to know whether two regions must be merged, two measures are considered. The *internal difference* $Int(X)$ of a region X is the highest edge weight among all the edges linking two vertices of X in the MST. The *difference* $Diff(X, Y)$ between two neighboring regions X and Y is the smallest edge weight among all the edges that link X to Y. Then, two regions X and Y are merged when:

$$Diff(X, Y) \leq \min\{Int(X) + \frac{\lambda}{|X|}, Int(Y) + \frac{\lambda}{|Y|}\} \quad (1)$$

where λ is a parameter used to prevent the merging of large regions (*i.e.*, larger λ forces smaller regions to be merged).

The merging criterion defined by Eq. (1) depends on the scale λ at which the regions X and Y are observed. More

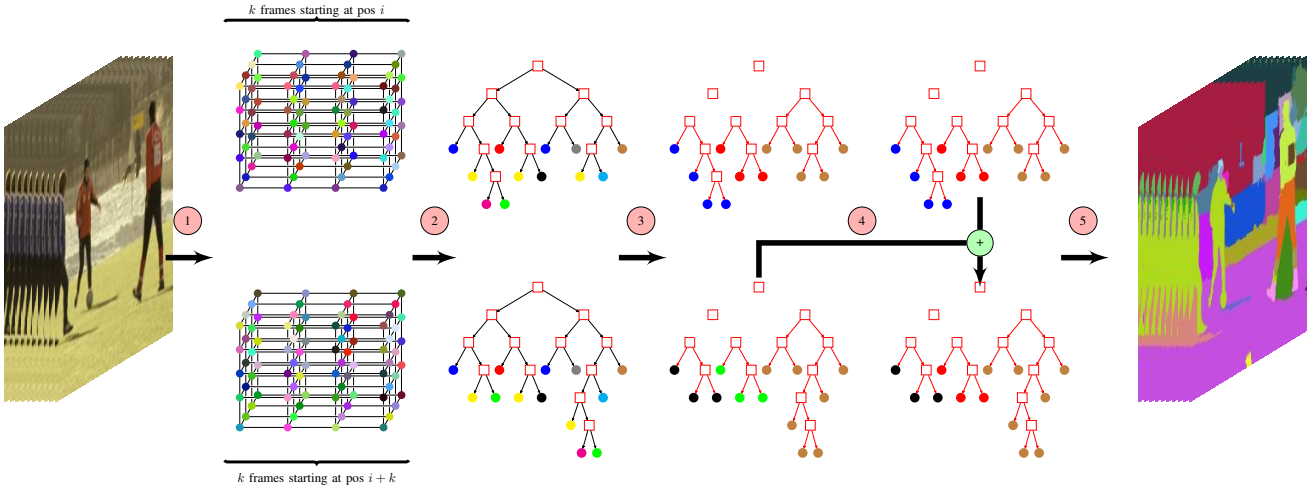


Fig. 2. Outline of our method. The method can be divided in five main steps: in step 1, the video is transformed into a video graph; in step 2, the hierarchy is computed from the video graph; in step 3, the identification of video segments is made from hierarchy; in step 4, the supervoxels output is calculated based on previously processed frames; and finally, in step 5, the video graph is transformed in the video segmented.

precisely, let us consider the *(observation) scale* $S_Y(X)$ *of* X *relative to* Y as a measure based on the difference between X and Y, on the internal difference of X and on the size of X:

$$S_Y(X) = (Diff(X,Y) - Int(X)) \times |X|. \quad (2)$$

Then, the *scale* $S(X,Y)$ is simply defined as:

$$S(X,Y) = \max(S_Y(X), S_X(Y)). \quad (3)$$

Thanks to this notion of a scale, Eq. (1) can be written as:

$$\lambda \geq S(X,Y). \quad (4)$$

The core of HOScale [9] is the identification of the smaller scale value that can be used to merge the largest region to another region while guaranteeing that the internal differences of these merged regions are larger than the value calculated for the smaller scale. In fact, instead of computing the hierarchy of partitions, a weight map constructed using the notion of scale presented in Eq. (1) is produced from which the desired hierarchy can be inferred (Step 3 in Fig. 2), *e.g.*, by removing from those edges whose weight is greater than the desired scale.

The main challenge here is to compute video segmentation as well as the method proposed in [9] preserving the performance measures. Thus, after computing segments for two consecutive k-sized frame blocks, a temporal coherence must be computed for producing consistent video segments. Regarding the temporal coherence, in this work, we use a new and simple strategy for merging two consecutive segmented blocks for identifying continuous supervoxels in the time. In order to do that, starting on the second block, the last frame of the previous block is incorporated at the begining of following block. Therefore a block consists of one "old" (processed) frame and k "new" (unprocessed), *i.e.*, two consecutive frame blocks are overlapped by one frame for providing the temporal coherence. The main idea for performing temporal coherence is to identify, in the overlapped frame, the supervoxels that contain voxels of both consecutive k-sized blocks. After this identification, we propagate the labels of previous block to new one. Thus, we identify the supervoxels which are continuous in the time.

III. Experimental analysis

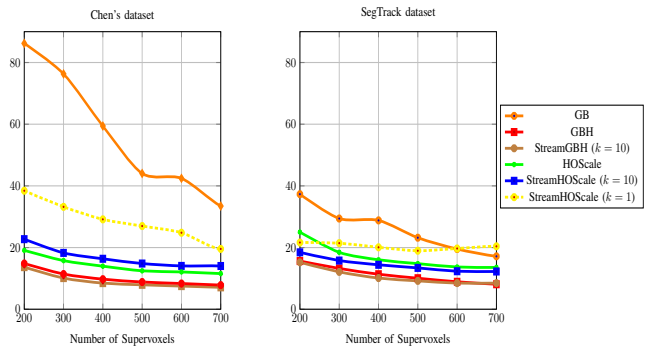
In order to provide a comparative analysis, we take into account the benchmark and library LIBSVX proposed in [12], since the implemented methods are the state of the art for video segmentation, including streaming video segmentation. The benchmark is composed, among others, by: (i) two datasets with groundtruth - Chen Xiph.org [13], SegTrack [14]; (ii) one dataset without groundtruth - GaTech dataset [10]; and (iii) implementations of the methods GB [11], GBH [10] and StreamGBH [7] applied to video segmentation.

A. Implementation issues

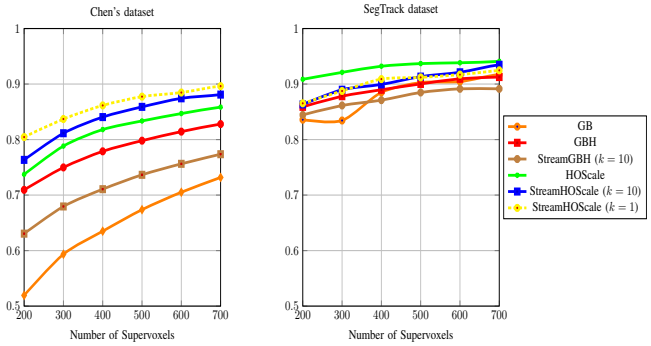
To efficiently implement our method, we use some data structures similar to the ones proposed in [15]; in particular, the management of the collection of partitions is made using Tarjan's union-find. Furthermore, we made some algorithmic optimizations to speed up the computations of the hierarchical observation scales. So, in order to create the video graphs, and when it is necessary, we employ a KD-tree for identifying the nearest neighbors in $RGBXYZ$ space. Our method is implemented in C++ and all experiments were made in a Quad-Core Intel Xeon E5620 2.4 Ghz 24GB RAM with Ubuntu 12.04.1 LTS.

B. Quantitative analysis

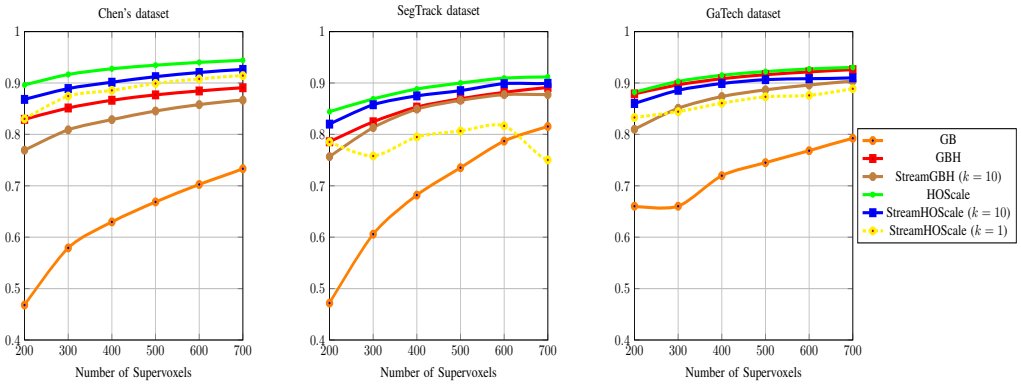
Using the library LIBSVX, developed by [12], it is possible to compute, among others, the following metrics: (i) 3D boundary recall; (ii) 3D undersegmentation errors; (iii) explained variation; and (iv) mean duration. For computing the first two metrics, a groundtruth is needed. The 3D boundary



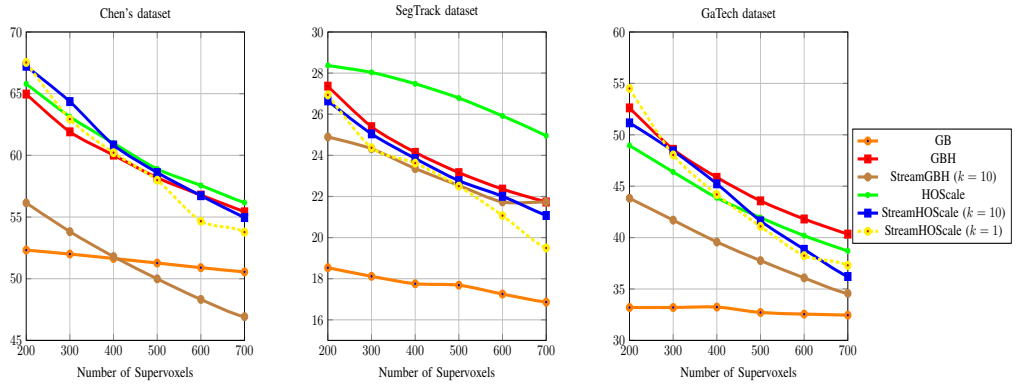
(a) 3D Undersegmentation Error



(b) 3D Boundary Recall



(c) Explained variation



(d) Mean duration

Fig. 3. A comparison between our method, StreamHOScale, and the methods GB, GBH, and StreamGBH when applied to Chen's, SegTrack and GaTech datasets (a) 3D Undersegmentation Error (b) 3D Boundary Recall (c) Explained Variation (d) Mean Duration.

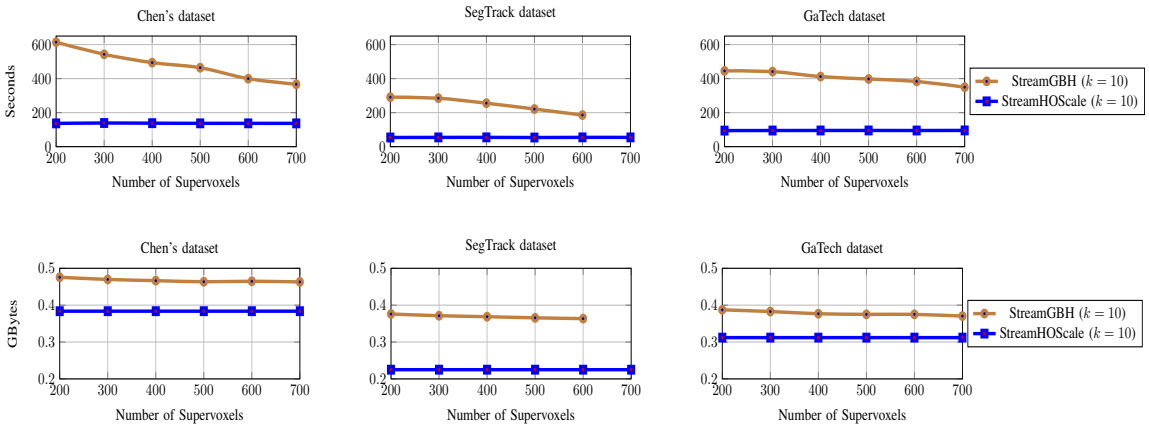


Fig. 4. A comparison between our method StreamHOScale and the method StreamGBH when applied to (from left to right) Chen's (i), SegTrack (ii) and GaTech (iii) datasets. The comparison is based on the following metrics: (top) total time; (down) space cost for all methods.

recall assesses the quality of the spatiotemporal boundary detection, while the 3D undersegmentation error measures what fraction of supervoxels exceeds the volume boundary of the groundtruth and the explained variation is a human independent measure assessing spatiotemporal uniformity. Finally, the mean duration quantifies the average duration, in terms of number of frames, of the video segments (or supervoxels).

Fig. 3 illustrates the average values of computed metrics for Chen's, SegTrack and GaTech datasets varying the number of desired supervoxels between 200 and 700, in which the parameters for our method are tuned per dataset, like in GBH and StreamGBH. For GB, the parameters are tuned per video. The strategy for filtering out small regions is the same adopted in [12], in which the size of the regions to be filtered out increases when the number of supervoxels decreases. In order to compare StreamGBH and StreamHOScale, we consider the size of each frame block equal to 10. Furthermore, for better understanding the behavior of StreamHOScale, we also consider $k = 1$. Without loss of generality, StreamHOScale can be considered as HOScale when $k = \infty$. As can be seen in Fig. 3 and as expected, the performance measures increase when the block size increases. However, it is worth to mention the behavior of StreamHOScale ($k = 10$) when compared to GBH since the former presents better results than latter, but GBH uses the full video information.

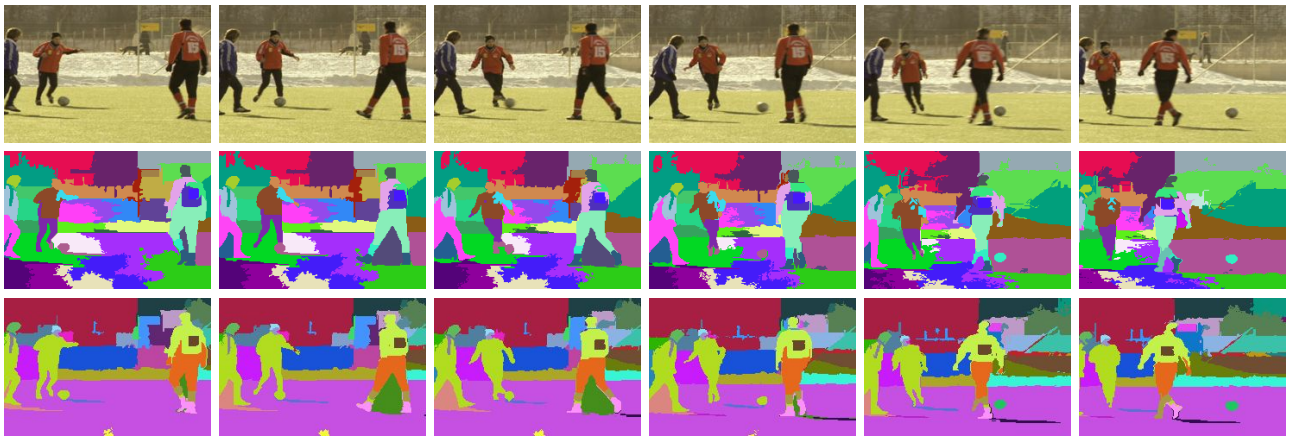
Concerning memory consumption and processing time cost, the proposed method StreamHOScale ($k = 10$) uses less memory than StreamGBH ($k = 10$), and it is much faster when we consider the processing time. Moreover, both memory consumption and time cost are quite constant for StreamHOScale ($k = 10$), independently of the number of supervoxels, as can be seen in Fig. 4.

Finally, Fig. 5 illustrates some results for StreamHOScale ($k = 10$) and StreamGBH ($k = 10$) when applied to videos from Chen and GaTech dataset, where we noted that the StreamHoscale method presents a better spatial coherence in the generated supervoxels. The StreamHOScale demonstrates to be a good method for large video segmentation, where the limite of memory and running time can be a problem.

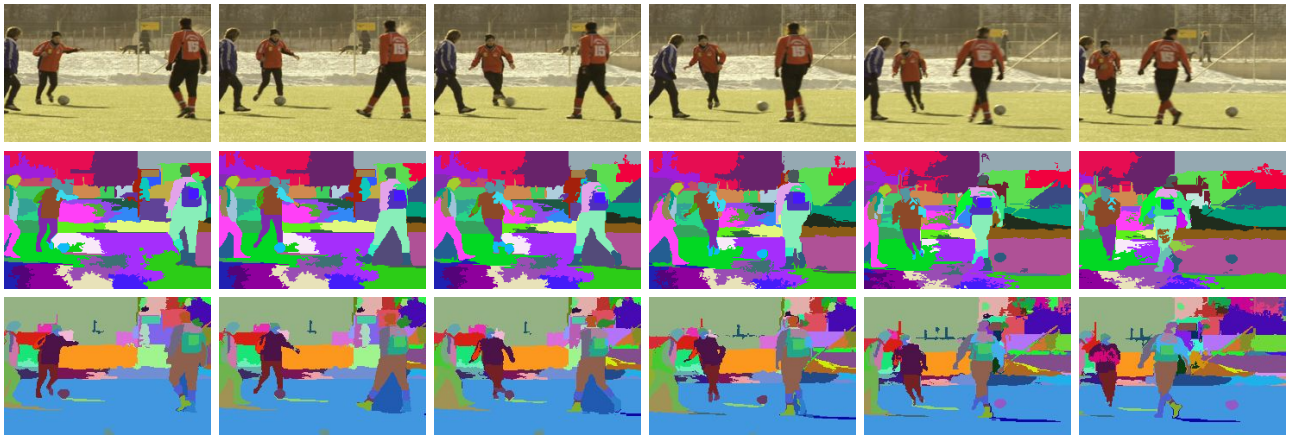
IV. Conclusions and Future Works

In this work, we proposed a method for streaming video segmentation based on computation of hierarchical observation scales – StreamHOScale. It can be divided into 4 (four) main steps: (i) graph creation for k-sized frame block; (ii) calculation of hierarchical scales for each graph; (iii) inference of video segmentation using thresholding for each graph; and (iv) computation of temporal coherence video segments belonging to consecutive blocks.

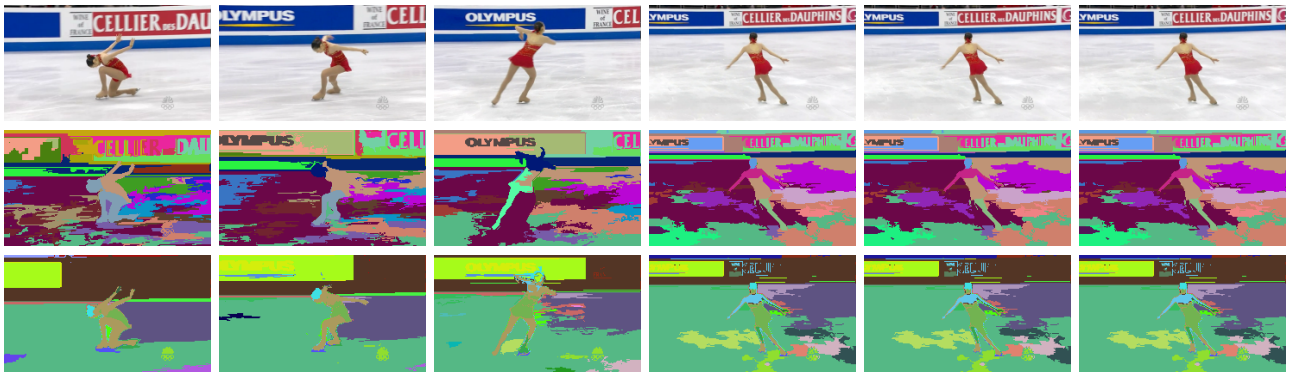
To compute the hierarchical scales for each k-sized frame block, we apply the hierarchical graph-based video segmentation proposed in [9]. This methodology reweights the minimum spanning tree computed from the video graph, based on a criterion that measures the evidence for a boundary between two regions, by comparing the intensity differences across the boundary and the intensity difference between neighboring voxels within each region. Finally, the partitioning of the graph, after the reweighting, is based on removing the edges whose weights (which represent the scales) are greater than or equal to a specified scale. According to experiments presented in [9], the hierarchies infered by *cp*-HOScale (or simply HOScale), produce good quantitative and qualitative results when applied to video segmentation. However, even though the method is quite fast, we propose in this paper a strategy for speed-up the computation of video segmentation without losing much quality. Thanks to the new and simple strategy for merging the results of two consecutive k-sized frame blocks, we produce good results preserving as much as possible the same quality measure of original one. Moreover, we outperform the compared methods for all performance measures, including memory consumption and processing time. For further works, we will study different ways for computing the video graph, mainly, in order to decrease the time for its creation and assess its impact on video segmentation results. Moreover, we will study how to compute streaming video segmentation in real time.



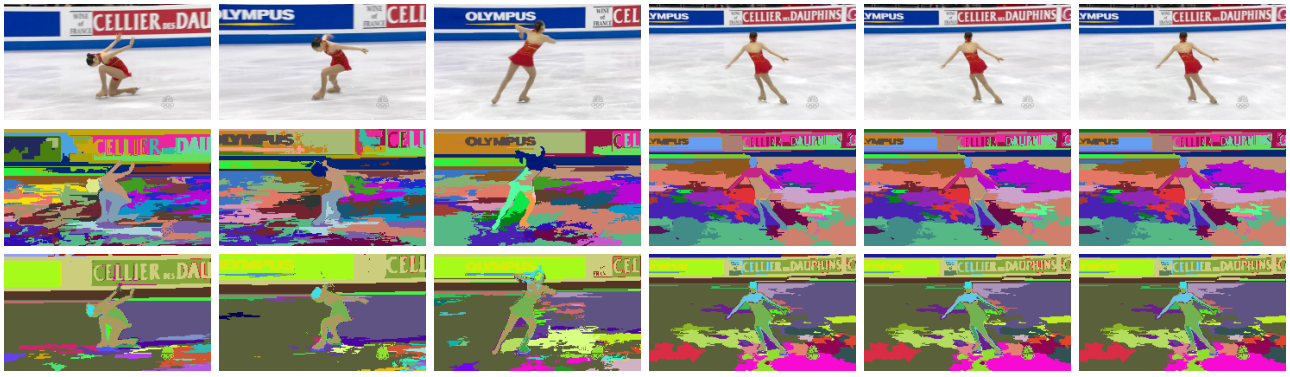
(a) 50 segments - soccer video from Chen dataset



(b) 100 segments - soccer video from Chen dataset



(c) 50 segments - yunakim_long2 video from GaTech dataset



(d) 100 segments - yunakim_long2 video from GaTech dataset

Fig. 5. Examples for videos from Chen and GaTech dataset. The original frames are illustrated in the first row. The following rows, from top to bottom, illustrate the results obtained by StreamGBH ($k = 10$) and StreamHOScale ($k = 10$). The parameters were tunned to obtain thereabout: (a) 50 and (b) 100 video segments to soccer video from Chen dataset and (c) 50 and (d) 100 video segments to yunakim_long2 video from GaTech dataset.

ACKNOWLEDGMENT

The authors are grateful to PUC Minas, UFMG, CNPq, CAPES, COFECUB and FAPEMIG for the financial support of this work.

REFERENCES

[1] J. Chen, S. Paris, and F. Durand, "Real-time edge-aware image processing with the bilateral grid," in *ACM SIGGRAPH 2007 papers*, ser. SIGGRAPH '07. New York, NY, USA: ACM, 2007.

[2] H. Winnemoller, S. C. Olsen, and B. Gooch, "Real-time video abstraction," *ACM Trans. Graph*, vol. 25, p. 2006, 2006.

[3] S. Paris and F. Durand, "A topological approach to hierarchical segmentation using mean shift," in *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, 2007, pp. 1–8.

[4] C. Fowlkes, S. Belongie, and J. Malik, "Efficient spatiotemporal grouping using the nystrom method," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1, 2001, pp. I–231–I–238 vol.1.

[5] J. Corso, E. Sharon, S. Dube, S. El-Saden, U. Sinha, and A. Yuille, "Efficient multilevel brain tumor segmentation with integrated bayesian model classification," *Medical Imaging, IEEE Transactions on*, vol. 27, no. 5, pp. 629–640, 2008.

[6] M. Grundmann, V. Kwatra, M. Han, and I. Essa, "Efficient hierarchical graph-based video segmentation," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, june 2010, pp. 2141 –2148.

[7] C. Xu, C. Xiong, and J. J. Corso, "Streaming hierarchical video segmentation," in *Proceedings of European Conference on Computer Vision*, 2012.

[8] K. J. F. De Souza, A. de Albuquerque Araújo, Z. K. Patrocínio Jr, J. Cousty, Y. Kenmochy, L. Najman, and S. J. F. Guimarães, "Hierarchical video segmentation using an observation scale," in *SIBGRAPI 2013 (XXV Conference on Graphics, Patterns and Images)*, N. Hirata, L. Nedel, C. Silva, and K. Boyer, Eds., Arequipa, Peru, august 2013.

[9] K. J. F. De Souza, A. De Albuquerque Araújo, Z. K. G. Do Patrocínio, Jr., and S. J. F. Guimarães, "Graph-based hierarchical video segmentation based on a simple dissimilarity measure," *Pattern Recogn. Lett.*, vol. 47, pp. 85–92, Oct. 2014. [Online]. Available: http://dx.doi.org/10.1016/j.patrec.2014.02.016

[10] M. Grundmann, V. Kwatra, M. Han, and I. Essa, "Efficient hierarchical graph based video segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[11] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *IJCV*, vol. 59, pp. 167–181, September 2004.

[12] C. Xu and J. J. Corso, "Evaluation of super-voxel methods for early video processing," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[13] A. Y. C. Chen and J. Corso, "Propagating multi-class pixel labels throughout video frames," in *Proceedings of Western New York Image Processing Workshop*, 2010.

[14] D. Tsai, M. Flagg, and J. M.Rehg, "Motion coherent tracking with multi-label mrf optimization," *BMVC*, 2010.

[15] S. J. F. Guimarães, J. Cousty, Y. Kenmochi, and L. Najman, "A hierarchical image segmentation algorithm based on an observation scale," in *SSPR/SPR*, 2012, pp. 116–125.