Buffering hierarchical representation of color video streams for interactive object selection

François Merciol, Sébastien Lefèvre

▶ **To cite this version:**

François Merciol, Sébastien Lefèvre. Buffering hierarchical representation of color video streams for interactive object selection. Advanced Concepts for Intelligent Vision Systems, Nov 2015, Catane, Italy. hal-01194373

HAL Id: hal-01194373
https://hal.science/hal-01194373v1

Submitted on 13 Nov 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Buffering Hierarchical Representation of Color Video Streams for Interactive Object Selection

François Merciol, Sébastien Lefèvre

Univ. Bretagne-Sud, UMR 6074, IRISA, F-56000 Vannes, France
{francois.merciol, sebastien.lefevre}@irisa.fr

Abstract. Interactive video editing and analysis has a broad impact but it is still a very challenging task. Real-time video segmentation requires carefully defining how to represent the image content, and hierarchical models have shown their ability to provide efficient ways to access color image data. Furthermore, algorithms allowing fast construction of such representations have been introduced recently. Nevertheless, these methods are most often unable to address (potentially endless) video streams, due to memory limitations. In this paper, we propose a buffering strategy to build a hierarchical representation combining color, spatial, and temporal information from a color video stream. We illustrate its relevance in the context of interactive object selection.

Keywords: Hierarchical representation, α-tree, interactive segmentation, color video processing

1 Introduction

Proliferation of high-resolution low-cost digital video recorders results in vast amounts of video data that need to be further processed for personal or professional use. Efficient video processing solutions are required to allow popular management of video files and librairies on standard computers as well as mobile terminals (e.g., tablets, smartphones). Indeed, real-time processing allowing interactivity with the user greatly eases the subsequent user acceptability of the proposed solutions. We focus in this paper on object selection (or segmentation), that is one of the most desired tools for video editing and analysis.

Video segmentation gathers a strong research interest for more than a decade [2–4, 8, 14, 17, 22]. To illustrate, one of the most recent techniques [23] dedicated to fast segmentation allows a processing frame rate of 1.3-1.5 fps, far below current video broadcast standards. Such an example is representative of the state-of-the-art where accuracy is sought at the cost of computational complexity. Interactive object selection from color video streams is thus hardly achievable with existing techniques.

We address here this issue and propose a new interactive object selection technique. Efficiency is achieved through the design of a hierarchical representation that replaces the raw (pixel) dataset in the different steps of the process. While such an approach has already shown promising results [12], it requires storing in memory the representation of the whole video sequence. We build upon this previous work and propose here a way to deal with (potentially endless) video streams without facing memory limitations, inspired from recent works on streaming or causal segmentation [5, 7, 13, 18, 25]. Furthermore, we introduce an improved object selection scheme based on bounding box input provided by the user. We also increase robustness to occlusions and motion w.r.t. our previous work [12].

The rest of the paper is organized as follows. In Sec. 2, we recall the existing work our method builds upon, namely the α-tree and its use in image/video segmentation. Our contribution is described in Sec. 3, while Sec. 4 presents its experimental evaluation on a standard dataset. We end the paper with concluding remarks and future directions.

2 Background

Our method builds upon a previous work [12] based on a hierarchical structure called the α-tree. Both are recalled in this section.

2.1 The α-tree model

An α-tree is a multiscale representation of an image through its α-connected components (or α-CCs). While it finds roots in early work in computer vision, it has been revisited only recently by Soille and Ouzounis [15, 19]. This paradigm is very related to the single linkage procedure used in data clustering. It provides a compact representation of the image that allows its real-time processing. Furthermore, efficient algorithms have been recently introduced to ensure fast computation of this representation from complex images [9].

The concept of α-CC is an extension of the connected component (or CC). We recall that the latter is defined as a set of adjacent pixels that share the same value (either scalar for panchromatic images, or vectorial for multi- or hyperspectral ones). Representing an image by its CCs allows for higher-level analysis (similarly to computer vision techniques relying on superpixels). However, the possibly great number of CCs in an image prevents their practical use. Indeed, adjacent pixels may belong to the same structure but have slightly different values, thus belonging to different CCs. The concept of α-CC has been introduced to allow such slight variations, leading to the following definition: an α-CC is a set of adjacent pixels that share similar values i.e., values with a difference lower or equal to a threshold α. The α-CC of a pixel p will thus contain all pixels q that can be reached with a path over neighboring pixels p_i $\langle p_1 = p, \ldots, p_n = q \rangle$ from p to q such that $d(p_i, p_{i+1}) \leq \alpha$ (d being a predefined dissimilarity measure). The complexity and number of α-CCs are directly related to α. It allows

building a hierarchical representation of an image, and performing subsequent multiscale analysis (e.g., in an object-oriented strategy). This representation is called an α-tree. Each level of the tree is indexed by an α value, and its nodes are the corresponding α-CCs. A leaf in the tree is a 0-CC i.e., a standard CC in the image. Increasing α leads to the connection of α-CCs, resulting in the creation of higher nodes in the tree, until the root that contains the whole image.

2.2 Video segmentation based on α-tree

In a previous work [12], we have already proposed to use the α-tree to perform video segmentation. However, the tree was computed on the complete video sequence assuming space-time connectivity and representing the video as a spatio-temporal volume. More precisely, each pixel was defined by a triplet (x, y, t) with two spatial and one temporal coordinates, and the neighborhood was using the 6-connectivity (i.e., two pixels (x, y, t) and (x', y', t') are neighbors if $|x - x'| + |y - y'| + |t - t'| = 1$). The dissimilarity measure d used to build the α-CC is the Chebyshev distance computed between the colors $\mathbf{c} = (r, g, b)$ and $\mathbf{c}' = (r', g', b')$ of adjacent pixels p and p', i.e. $d(p, p') = \max(|r - r'|, |g - g'|, |b - b'|)$. This allows keeping the number of possible dissimilarity values as low as the input range of each color component (e.g. 256 levels), conversely to Euclidean distance. The height of the resulting α-tree is then bounded by this range. RGB color channels were used directly in order to avoid the additional cost of a color space transformation.

Once the α-tree representation of the full video sequence is computed, it is enriched by associating some features (size, average brightness and hue) to each node of the tree. Such features are computed incrementally, starting from the leaves of the tree up to the root, thus limiting the computational complexity. Averaging hue information is done in a specific manner to ensure the reliability of this feature (see [12] for a complete description of the method). The video representation is then ready to be analyzed for interactive segmentation. To do so, the user picks one pixel from a video frame, i.e. a leaf in the α-tree. Object selection is then achieved through a traversal of the tree in order to find the most relevant node in the path from the selected leaf to the tree root. More precisely, the size (number of pixels) of each traversed node is analyzed, and if this measure is stable for a significant number of levels in the tree, the node is used to define the object selection. Let us note that this process shares some similarity with the extraction of Maximally Stable Extremal Regions (MSER) [11].

While our previous method [12] showed promising results, it came with several limitations: (i) the α-tree has to be computed on the full video sequence before any further processing (such as interactive object selection); long video sequences, as well as (potentially endless) video streams cannot be addressed; (ii) to perform interactive object selection, user input consists of a single pixel only (i.e., a leaf in the tree); such an initialization is very error prone, and hardly provides an accurate description of complex objects (with heterogeneous content); (iii) the selection process ends with a single node from the tree, while the

object might be better represented by several nodes with no heritage relations; (iv) the α-tree is computed on the spatio-temporal volume, assuming spatio-temporal continuity of the objects; this is not the case in the presence of object motion and occlusions, that could result in disconnected components that might have only the root as common ancestor in the tree. These different issues are addressed in the new method proposed in this paper.

3 Proposed method

The method we are proposing in this paper starts with a first input from the user to define the initial contour of the object. It then propagates the selection in the following frames in an online setting. An additional step is added to deal with spatio-temporal discontinuities. We describe here these different steps.

3.1 User-driven object selection

The video object selection scheme is interactive. In a frame of the video sequence (generally the second frame, see discussion below), we ask the user to delineate the object of interest through a bounding box (A in Fig. 1). We assume the object to be completely included in this box. We also compute two α-trees, one for the frame where user selection occurs (Fig. 2(b)) and one for its immediate preceding frame (Fig. 2(a)). Both α-trees are merged into a single tree (Fig. 2(c)). The goal is then to identify, among the nodes of this merged α-tree, the ones corresponding to the selected object. To do so, we first remove all nodes of the tree that overlap the background, or in other words that are not completely included in the box provided by the user (an example of such discarded node is C in Fig. 1). This can be efficiently achieved starting from the leaves corresponding to the pixels included in the box, and scanning their ancestors until the latter span over the initial bounding box. The last (i.e., closest to the root) ancestors that fit in the bounding box are kept. The selection thus results in one or several nodes. When a node is selected, all its children are too. In other words, one or several subtrees (i.e., a forest) are extracted from the α-tree to denote the selected object. However, we have observed that this strategy was prone to foreground and background mixing, since it might select nodes that are located on the interior edges of the user box. We thus add an additional constraint, relying on a reduced user selection (D in Fig. 1) built automatically using a given reducing ratio (here 50%). Nodes whose centers do not belong to this reduced zone are also discarded from the object selection (an example is E in Fig. 1). To ease understanding, only the user input and the selected objects are shown in the left image of Fig. 1. Figure 2(d) also provide some tree illustration.

3.2 Selection propagation in the video stream

After having refined a selection from a user input, the next step is to propagate it in the following frames of the video sequence. Figure 2 illustrates the

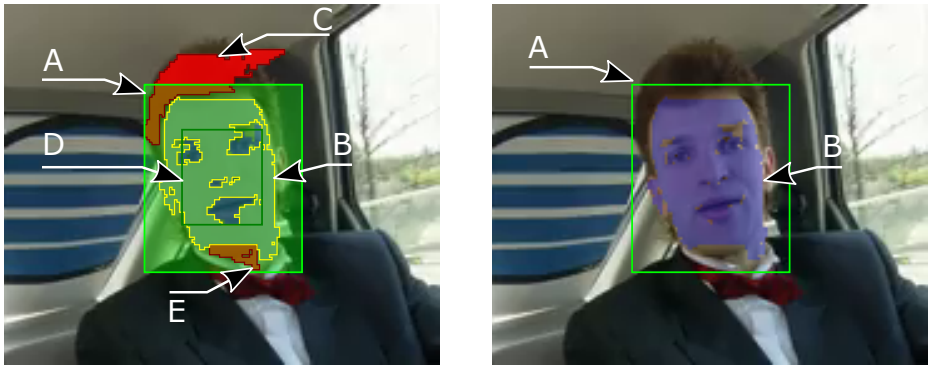

Fig. 1. Illustration of the selection process: user input (A, light green), nodes corresponding to selected object (B, yellow), discarded nodes due to background overlapping (C, orange/red), reduced user selection (D, dark green), and discarded nodes due to non strong overlapping with reduced user selection (E, orange/red).

propagation process. This step is required since, as already stated, we do not compute an α-tree for the full video sequence but we rather process the video on a frame-by-frame basis. This allows for processing very long video sequences as well as video streams (i.e. potentially endless). The selection process described previously and applied on an initial frame leads to two α-trees respectively built from this frame (a) and its preceding frame (b), that are subsequently merged into a single unified tree (c). To do so, temporal connectivity is considered in computing the α-CCs, possibly leading to merging two nodes at lower α values than if considering the individual trees (useless tree edges are shown dashed). Once a selection has been defined by the user (d), we keep only leaves corresponding to the current frame. Selection propagation in the next frame requires first computing the tree in this new frame (e), and then to merge both trees (f) while keeping the labels for selected nodes. The temporal connectivity allows merging spatially disconnected nodes. Again, the merged tree is filtered to keep only leaves corresponding to current pixels (g). Furthermore, in order to limit the memory footprint, we also prune the tree and remove the intermediary nodes that are not on the selection branch, thus forgetting outdated information. But storing the connectivity information (spatial from the previous frame as well as temporal for the last couple of frames) results in a tree that could not have been built from a single frame only, e.g. see (e) and (g). The process is repeated all along the video sequence, with each successive frame leading to a new individual tree (h), merged tree (i), and filtered one (j) where selection is propagated. In this last example (g)-(i), we can observe that the selection can extend to some new objects if their color similarity to the selection becomes higher. Duplicate nodes (shown in grey) along the path from the selection to the root are temporarily buffered to allow reconnection/disconnection operations.

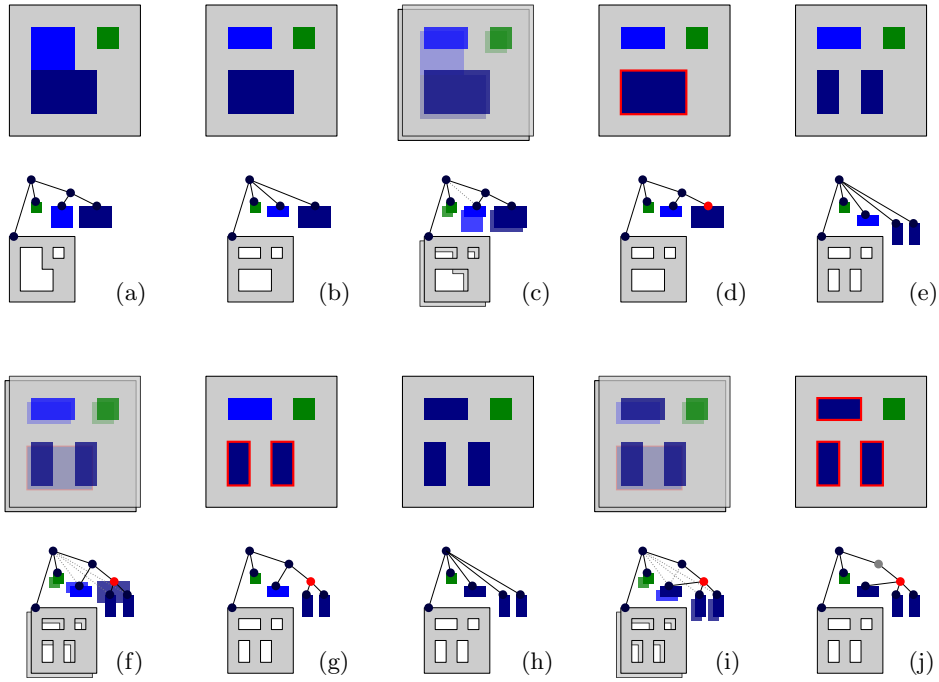

Fig. 2. Illustration of the buffering process: individual trees (a,b,e,h), after merging (c,f,i) and filtering (d,g,j). Selected nodes, useless relations, and residual nodes are shown in red, dashed, and in grey respectively.

3.3 Dealing with spatio-temporal discontinuities

Extracting the object of interest in the frame where user input has been provided is much easier than in the following frames. Indeed, the selection is propagated from one frame to the next using the scheme described previously, and the selection accuracy might decrease along time. Model updating is required and several strategies are available. We have explored several automatic strategies but their performances were not satisfying, so we rather chose here to rely on a manual strategy. It needs to identify the frames for which an updated input is required from the user, and we use here a size criterion. More precisely, once a selection has been propagated in an incoming frame, we compare the size of the updated selection with the size of the last user input. If both sizes differ from a ratio higher than T (here $T = 2$), we consider the selection to be inaccurate since it corresponds to either a too small or too large component. This occurs in the presence of spatio-temporal discontinuities, especially observed with object motion and occlusion.

4 Experiments

We evaluate the proposed approach on a standard dataset and compare our results with the state-of-the-art. More precisely, we use the SegTrack dataset [20] that contains 6 video sequences, with length of 20–70 images of rather small size (from 320×240 pixels to 414×352 pixels), together with a reference segmentation (ground truth).

Fig. 3. Sample frames for the 3 video sequences from SegTrack used for quantitative evaluation: Birdfall, Parachute, and Girl (Tab. 1).

For the sake of illustration, we compare our method with some recent techniques based on selection propagation that have reported results on this dataset [10, 21]. Both methods rely on probabilistic modeling with Markov Random Fields (MRF), but while the former operates on pixels, the latter imposes a space-time graph structure on the data. The 3 video sequences used in this paper, BIRDFALL, PARACHUTE, and GIRL, are illustrated in Fig. 3. An illustration

	Ours	[21]	[10]
birdfall	313 (0.003)	405 (0.005)	189 (0.002)
parachute	1337 (0.009)	1042 (0.007)	228 (0.002)
girl	6632 (0.052)	8575 (0.067)	2883 (0.023)
time	11–27	480–600	120–180

Table 1. Comparative results for interactive object selection. Accuracy is expressed as the average number (and ratio) of mis-segmented pixels (false positive plus false negative) per frame. Runtime is provided in seconds. Results are reported from [10].

of the results obtained with our method is given in Fig. 4, with the error rate per frame plotted in Fig. 5. On this PARACHUTE sequence, user selection has been required 4 times. This is due to the complexity of the video sequence, with some important changes in both object illumination and pose. While these user inputs allow preventing a severe increase of the error rate, they were not sufficient enough to keep it as low as on the first frames of the video sequence (for which the tracking was less challenging).

Comparative results are provided in Tab. 1. We can see that the proposed method performs slightly better than [21], but worse than [10]. Let us recall that we are not using any motion information conversely to the state-of-the-art. Furthermore, the proposed approach is still deterministic while probabilistic models have achieved great successes in computer vision for decades. Besides, the α-tree model considered here leads to some spatio-temporal chaining effects that cannot be overcome without any post-processing or constraint imposed on the connectivity between pixels. These different limitations are directions for future work.

More interestingly, we can observe from Tab. 1 that the proposed solution based on tree structures brings a significant gain in terms of performance. The reported runtime of our method is $6 - 11$ times less than [10], and $22 - 43$ times less than [21] (let us note however that the implementation and runtime details, e.g. coding langage, CPU speed, etc. are not provided in [10, 21]). Our method has been benchmarked on a Java implementation and a standard laptop configuration, thus allowing fair comparison with recent works from the state-of-the-art. We have observed CPU time wastes due to the Java Garbage Collector that call for further optimization.

We believe that it is possible to build upon the proposed technique to introduce more complex (but still tree-based) video analysis strategies, to ensure both high accuracy and efficiency. To illustrate, let us recall that many video segmentation methods rely on a first segmentation into superpixels, that might be easily produced by cutting the tree. But while extracting superpixels from a SegTrack video requires at least 500 seconds with the most efficient techniques from the state-of-the-art [24] (measured with a Dual Quad-core Intel Xeon CPU E5620 2.4 GHz, 16GB RAM running Linux), our algorithm is able to provide a set of superpixels in less than 50 seconds with a Java implementation and a

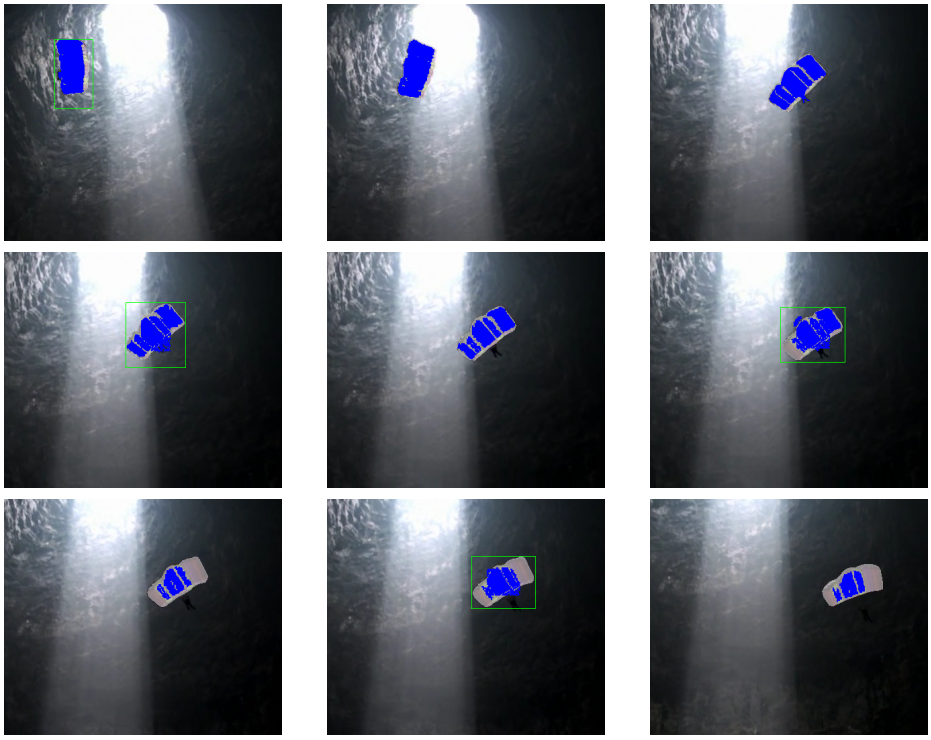

Fig. 4. Sample results for the Parachute video sequence (frames 2, 12, 30, 31, 34, 35, 39, 40, 51): manual selection is shown in green, selected regions in blue, ground truth pixels are brighter.

standard laptop configuration (Dual-core Intel Core CPU i5-3320M 2.60GHz, 6GB RAM running Linux).

5 Conclusion

In this paper, we have explored a new way to represent video pixels through a hierarchical representation. Such a representation has been used to derive an efficient solution allowing interactive object selection from a color video stream. Results obtained on the SegTrack standard dataset are promising, with accuracies similar to the state-of-the-art but computation times much lower. Nevertheless, experimental evaluation and comparison with state-of-the-art has to be pursued, especially considering other streaming strategies recently introduced in the literature [5, 7, 13, 18, 25].

While efficiency is definitely a strength of the proposed solution, segmentation accuracy could be further improved to meet user requirements. This can be achieved following several directions. Chaining effects, a known drawback of single-linkage representations such as the α-tree, can be alleviated using more

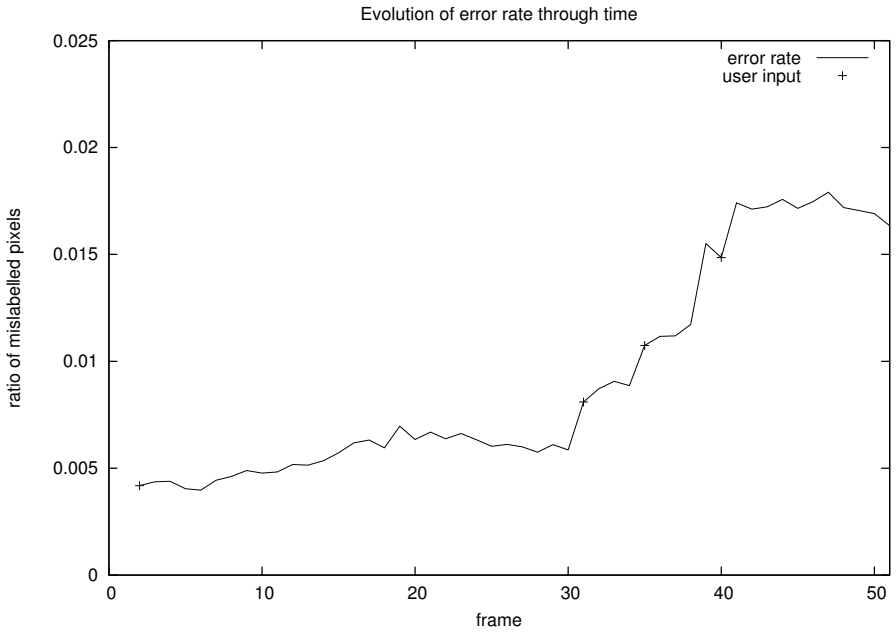

Fig. 5. Evolution of the error rate (ratio of mislabelled pixels per frame). Crosses denote frames where user input was required.

complex tree models, e.g. binary partition tree [6, 16]. Improving the way the selection is propagated between two successive frames can be ensured by exploiting the object motion that is not taken into account yet. Furthermore, the introduction of probabilistic models would strengthen the robustness of the method and could also lead to better accuracy (as demonstrated recently with binary partition tree-based image segmentation [1].

More generally, we also consider applying the hierarchical representations to other computer vision problems that are facing computational and memory issues. Indeed, the proposed framework is particularly adapted to online settings. As such, a fully automatic solution that will not require manual initialization will be also appealing for addressing big video data.

References

1. Al-Dujaili, A., Merciol, F., Lefèvre, S.: GraphBPT: An efficient hierarchical data structure for image representation and probabilistic inference. In: International Symposium on Mathematical Morphology. Lecture Notes in Computer Science, vol. 9082, pp. 301–312 (2015)
2. Bai, X., Wang, J., Simons, D., Sapiro, G.: Video snapcut: robust video object cutout using localized classifiers. In: Proceedings of the SIGGRAPH. pp. 1–11 (2009)

3. Boykov, Y., Funka-Lea, G.: Graph cuts and efficient n-d image segmentation. International Journal of Computer Vision 70(2), 109–131 (2006)
4. Boykov, Y., Jolly, M.: Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In: Proceedings of the ICCV. pp. 105–112 (2001)
5. Couprie, C., Farabet, C., LeCun, Y., Najman, L.: Causal graph-based video segmentation. In: IEEE International Conference on Image Processing. pp. 4249–4253 (2013)
6. Dorea, C., Pardas, M., Marques, F.: A motion-based binary partition tree approach to video object segmentation. In: IEEE International Conference on Image Processing. vol. 2, pp. 430–433 (2005)
7. Gangapure, V., Nanda, S., Chowdhury, A., Jiang, X.: Causal video segmentation using superseeds and graph matching. In: Graph-Based Representations in Pattern Recognition. Lecture Notes in Computer Science, vol. 9069, pp. 282–291 (2015)
8. Grundmann, M., Kwatra, V., Han, M., Essa, I.: Efficient hierarchical graph based video segmentation. IEEE CVPR (2010)
9. Havel, J., Merciol, F., Lefèvre, S.: Efficient schemes for computing α-tree representations. In: International Symposium on Mathematical Morphology. LNCS, vol. 7883, pp. 111–122. Springer (2013)
10. Jain, S., Grauman, K.: Supervoxel-consistent foreground propagation in video. In: European Conference on Computer Vision. Lecture Notes in Computer Science, vol. 8692, pp. 656–671 (2014)
11. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide-baseline stereo from maximally stable extremal regions. Image and vision computing 22(10), 761–767 (2004)
12. Merciol, F., Lefèvre, S.: Fast image and video segmentation based on α-tree multiscale representation. In: International Conference on Signal Image Technology Internet Systems. Naples, Italy (November 2012)
13. Mukherjee, D., Wu, Q.: Streaming spatio-temporal video segmentation using gaussian mixture model. In: IEEE International Conference on Image Processing. pp. 4388–4392 (2014)
14. Noma, A., Graciano, A., Jr, R.C., Consularo, L., I.Bloch: Interactive image segmentation by matching attributed relational graphs. Pattern Recognition 45, 1159–1179 (2012)
15. Ouzounis, G.K., Soille, P.: Pattern spectra from partition pyramids and hierarchies. In: International Symposium on Mathematical Morphology. pp. 108–119. Verbania-Intra, Italy (2011)
16. Palou, G., Salembier, P.: Hierarchical video representation with trajectory binary partition tree. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 2099–2106 (2013)
17. Price, B., Morse, B., Cohen, S.: Livecut: Learning-based interactive video segmentation by evaluation of multiple propagated cues. In: IEEE International Conference on Computer Vision (2009)
18. Pu, S., Zha, H.: Streaming video object segmentation with the adaptive coherence factor. In: IEEE International Conference on Image Processing. pp. 4235–4238 (2013)
19. Soille, P.: Constrained connectivity for hierarchical image partitioning and simplification. IEEE Transactions on Pattern Analysis and Machine Intelligence 30(7), 1132–1145 (2008)
20. Tsai, D., Flagg, M., Rehg, J.: Motion coherent tracking with multi-label mrf optimization. British Machine Vision Conference (2010)

21. Vijayanarasimhan, S., Grauman, K.: Active frame selection for label propagation in videos. In: European Conference on Computer Vision. Lecture Notes in Computer Science, vol. 7576, pp. 496–509 (2012)
22. Wang, J., Bhat, P., Colburn, R., Agrawala, M., Cohen, M.: Interactive video cutout. ACM Transactions on Graphics 24(3), 585–594 (2005)
23. Wang, T., Han, B., Collomosse, J.: Touchcut: Fast image and video segmentation using single-touch interaction. Computer Vision and Image Understanding 120, 14–30 (2014)
24. Xu, C., Corso, J.J.: Evaluation of super-voxel methods for early video processing. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (2012)
25. Xu, C., Xiong, C., Corso, J.: Streaming hierarchical video segmentation. In: European Conference on Computer Vision. Lecture Notes in Computer Science, vol. 7577, pp. 626–639 (2012)