# Streaming Video Segmentation via Short-Term Hierarchical Segmentation and Frame-by-Frame Markov Random Field Optimization

Won-Dong Jang(✉) and Chang-Su Kim

School of Electrical Engineering, Korea University, Seoul, Korea
wdjang@mcl.korea.ac.kr, changsukim@korea.ac.kr

**Abstract.** An online video segmentation algorithm, based on short-term hierarchical segmentation (STHS) and frame-by-frame Markov random field (MRF) optimization, is proposed in this work. We develop the STHS technique, which generates initial segments by sliding a short window of frames. In STHS, we apply spatial agglomerative clustering to each frame, and then adopt inter-frame bipartite graph matching to construct initial segments. Then, we partition each frame into final segments, by minimizing an MRF energy function composed of unary and pairwise costs. We compute the unary cost using the STHS initial segments and the segmentation result at the previous frame. We set the pairwise cost to encourage similar nodes to have the same segment label. Experimental results on a video segmentation benchmark dataset, VSB100, demonstrate that the proposed algorithm outperforms state-of-the-art online video segmentation techniques significantly.

**Keywords:** Video segmentation · Online segmentation · Streaming segmentation · Agglomerative clustering · Graph matching

## 1 Introduction

Segmentation, the task of partitioning data into disjoint subsets based on the underlying data structure, is one of the most fundamental problems in computer vision. For image segmentation, contour-based algorithms [1,2] have achieved great success recently. As the state-of-the-art contour detector [3] presents comparable performance to the human visual system, the contour-based image segmentation can provide more promising performance. On the other hand, video segmentation is the process to divide a video into volumetric segments. It is applicable to a wide variety of vision applications, such as action recognition, scene classification, video summarization, content-based video retrieval, and 3D reconstruction. However, video segmentation still remains a challenging problem due to object and camera motion, occlusion, and contour ambiguities. To overcome these issues, many attempts have been made.

Video segmentation algorithms can be categorized into offline or online ones. Offline algorithms [4–10] divide a video into segments by processing all frames


B. Leibe et al. (Eds.): ECCV 2016, Part VI, LNCS 9910, pp. 599–615, 2016.
DOI: 10.1007/978-3-319-46466-4_36

at once. On the other hand, online (or streaming) algorithms [11–13] extract segments sequentially from the first to the last frames. Note that the offline algorithms can achieve more accurate segmentation by exploiting the entire information in a video jointly, but they require huge memory space for a long video. Thus, the online algorithms, which use regular memory space regardless of the duration of a video, can be used more versatilely in practical applications.

We propose a novel online video segmentation algorithm. The proposed algorithm consists of two steps: short-term hierarchical segmentation (STHS) and Markov random field (MRF) optimization. In the first pass, STHS generates initial segments sequentially, by sliding a short window of frames, to identify newly appearing segments effectively. It attempts to prevent the propagation of erroneous segments by processing each frame independently of the previous segmentation results. In the second pass, we define an MRF energy function for obtaining the final segmentation result of each frame, which consists of unary and pairwise costs. The unary cost takes into account the segmentation result at the previous frame and the initial STHS result at the current frame. The pairwise cost is computed based on node affinities. Then, we achieve temporally coherent and spatially accurate video segmentation by minimizing the energy function. Experimental results demonstrate that the proposed algorithm outperforms the state-of-the-art conventional algorithms in [11–13] on the video segmentation benchmark (VSB) dataset [14]. To summarize, this paper has three main contributions.

- Development of STHS, which combines spatial agglomerative clustering and temporal bipartite graph matching to detect newly appearing objects and achieve initial video segmentation reliably.
- Proposal of the MRF optimization scheme, which refines the initial segmentation results and yield temporally coherent and spatially accurate segments.
- Remarkable performance achievement on the VSB dataset, which consists of challenging video sequences.

## 2 Related Work

### 2.1 Offline Video Segmentation

An offline video segmentation algorithm processes all frames in a video simultaneously. Corso *et al.* [4] developed a graph-based video segmentation algorithm using a hierarchical structure. Grundmann *et al.* [5] also proposed a hierarchical algorithm, which merges similar superpixels sequentially in a spatiotemporal graph. Galasso *et al.* [7] first applied the spectral clustering [15] to the video segmentation problem. Galasso *et al.* [14] assessed video segmentation algorithms by introducing a benchmark dataset, called VSB100. Khoreva *et al.* [8] introduced learning-based must-link constraints, which enforce two nodes to belong to the same cluster during spectral clustering. Also, Khoreva *et al.* [9] trained a classifier to determine affinities between superpixels, and selected edges to construct a sparse efficient graph. Yi and Pavlovic [10] proposed an MRF model, whose

node potentials are obtained from the results of [5]. Yu *et al.* [16] introduced a parametric graph partitioning method to identify and remove between-cluster edges. While these offline algorithms provide promising segmentation results, they often demand huge memory space to process all frames simultaneously. Thus, they may fail to segment long video clips.

### 2.2 Online Video Segmentation

An online (or streaming) video segmentation algorithm sequentially partitions from the first to the last frames in a video sequence. To segment a frame, it uses only a few (usually less than 10) previous and subsequent frames. Vazquez-Reina *et al.* [17] proposed an online algorithm, which sequentially divides video frames into partitions by selecting optimal hypothesis flows of superpixels. Xu *et al.* [11] applied the hierarchical image segmentation algorithm in [18] to two consecutive frames to propagate segment labels temporally. Also, online supervoxel algorithms have been proposed to yield regularly sized spatiotemporal segments [19,20]. Recently, Galasso *et al.* [12] reduced the full graph for a video, by re-assigning edge weights, and achieved streaming segmentation by performing the clustering on the reduced graph. Moreover, Li *et al.* [13] decomposed an affinity matrix into low-rank ones to represent relations among supervoxels efficiently and applied the normalized cuts to the low-rank matrices.

### 2.3 Video Object Segmentation

Many attempts have been made to separate salient objects from the background in a video. Shi and Malik [21] clustered motions using the normalized cuts. Brox and Malik [22] exploited long-term point trajectories to determine object tracks. Ochs and Brox [6] converted sparse point trajectories into dense regions to yield pixel-wise object annotations. Ochs and Brox [23] employed the spectral clustering on point trajectories to delineate objects. Also, several algorithms [24–26] have been proposed to achieve video object segmentation using object proposal techniques. They first generate object proposals in all frames and then delineate objects by determining proposal tracks. Oneata *et al.* [27] developed a video object proposal algorithm by generating supervoxels. Wang *et al.* [28] adopted saliency detection techniques to segment a primary object. Giordano *et al.* [29] segmented moving objects by observing temporal consistency of sequential superpixels. Taylor *et al.* [30] analyzed occluder-occluded relations to ensure temporal consistency of objects. Jang *et al.* [31] minimized an energy function by performing the alternate convex optimization to discover a primary object sequentially. However, these video object segmentation algorithms may fail to segment temporally static or small objects, since they focus on moving, salient, and relatively large objects in general.

## 3 Proposed Algorithm

We propose a novel online video segmentation algorithm. The input is a set of consecutive video frames, and the output is a set of the corresponding pixel-wise

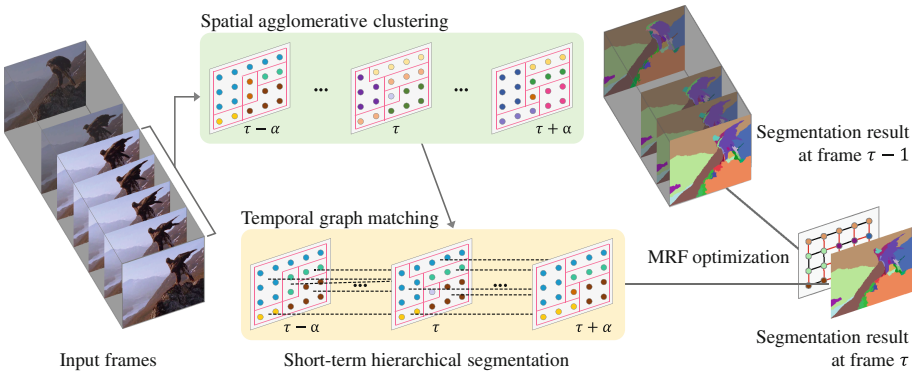



**Fig. 1.** Overview of the proposed algorithm. To partition the current frame $\tau$, we apply the short-term hierarchical segmentation (STHS) to a window of frames from $\tau - \alpha$ to $\tau + \alpha$. Then, we obtain the final segmentation result at frame $\tau$, by minimizing an MRF energy function, based on the initial STHS result and the previous segmentation result at frame $\tau - 1$

segment label maps. All pixels in a spatiotemporal segment are assigned the same label.

Figure 1 shows an overview of the proposed algorithm. First, we apply STHS to a short window of frames in order to segment the current frame initially. STHS merges spatially similar superpixels in each frame into clusters, and then links temporally coherent clusters. For the spatial and temporal merging, we adopt agglomerative clustering and bipartite graph matching, respectively. Second, we obtain final segment labels at the current frame $\tau$, by minimizing an MRF energy function that consists of unary and pairwise costs. The unary cost is defined using the initial STHS result and the previous segmentation result at frame $\tau - 1$, and the pairwise cost encourages similar nodes to have the same label. We perform this process sequentially from the first to the last frames to achieve streaming video segmentation.

### 3.1 Feature Extraction

For each frame $\tau$, we estimate both forward and backward optical flows using [32]. Also, we over-segment each frame into superpixels using the mean-shift algorithm [33]. For the mean-shift, we fix the parameters of spatial bandwidth and range bandwidth to 9 and 5, respectively, and set the minimum superpixel area to 0.1 % of the number of pixels in a frame. We extract three types of features: color feature, motion feature, and boundary feature. Let us describe these three features subsequently.

Color is a fundamental feature for image and video segmentation. We first represent each superpixel with a histogram of LAB colors. Each dimension is quantized into 20 bins independently. Also, we extract bag-of-words (BoW) features in the LAB and RGB color spaces, respectively. We generate the BoW using

the $K$-means algorithm, where $K$ equals 300 for both LAB and RGB spaces. By aggregating the encoded words in each superpixel, we obtain the LAB and RGB BoW histograms. Thus, to obtain the color feature $\mathbf{h}^{\mathrm{c}}$ of a superpixel, we concatenate the LAB histogram, the LAB BoW histogram, and the RGB BoW histogram. Consequently, the dimension of a color feature is 660, $\mathbf{h}^{\mathrm{c}} \in \mathbb{R}^{660\times 1}$.

Unlike image segmentation, video segmentation can exploit motion features, as well as color features. Motion features are complementary to color features, since they can distinguish similarly colored regions that move differently. To encode motion characteristics, we construct a BoW, by employing both backward and forward optical flows and setting $K$ to 100. We represent each superpixel with the backward and forward optical flow BoW histograms, respectively. Then, we construct the motion feature $\mathbf{h}^{\mathrm{m}}$ by cascading the two histograms. Therefore, the dimension of a motion feature is 200, $\mathbf{h}^{\mathrm{m}} \in \mathbb{R}^{200\times 1}$. In the first frame, the backward optical flow BoW histogram is unavailable, and thus copied from the forward one. Similarly, in the last frame, the forward histogram is copied from the backward one.

A good segment should be enclosed by a reliable boundary. Hence, we adopt a boundary feature to differently characterize superpixels that have strong boundaries between them. More specifically, we use results of the contour-based segmentation algorithms in [1,34], which generate segments with reliable boundaries. Let us consider segment labels as encoded words on each pixel. Then, we can obtain a histogram of the segment labels for each superpixel. Notice that the number of bins varies according to the number of segment labels. In this work, to exploit multiple levels of segmentation granularity, we generate three segmentation maps with thresholds 0.1, 0.3, and 0.5, respectively. Figure 2 shows segmentation results of [1] according to the thresholds. As the threshold increases, segments are divided by stronger boundaries only. We cascade the three label histograms to obtain the boundary feature $\mathbf{h}^{\mathrm{b}}$ of a superpixel. When two superpixels have different boundary features, there are a strong boundary between them. Notice that we use [34] for a faster version of the proposed algorithm.

We construct the LAB, RGB, and optical flow BoW features using the 40 training sequences in the VSB100 dataset [14]. We normalize each feature $\mathbf{h}^{\mathrm{c}}$, $\mathbf{h}^{\mathrm{m}}$, or $\mathbf{h}^{\mathrm{b}}$ to make its $l_2$-norm to 1, *i.e.* $\sum_i h_i^2 = 1$.

(a) Input frame (b) UCM (c) UCM> 0.1 (d) UCM> 0.3 (e) UCM> 0.5

**Fig. 2.** Examples of various segmentation results using an ultrametric contour map (UCM) [1]. As the threshold increases, segments are separated by stronger boundaries only

### 3.2 Short-Term Hierarchical Segmentation

In general, offline algorithms delineate newly appearing objects more effectively than online ones do, since they consider all frames at once. It is hard to find new objects using the current and previous frames only. Therefore, we develop STHS that performs initial segmentation of frame $\tau$, by sliding a short window of frames from $\tau - \alpha$ to $\tau + \alpha$, where $\alpha$ is set to 7. In other words, STHS consider the subsequent $\alpha$ frames, as well as the current and previous $\alpha$ frames, to identify object appearance more effectively. In general, the future frames are used in the streaming video segmentation algorithms [7,11,35]. Also, within the entire segmentation algorithm in Fig. 1, STHS helps to alleviate the propagation of segmentation errors in the previous frames, by providing an initial segmentation result, which is independent of the previous segmentation results. Figure 3 visualizes the efficacy of STHS in comparison with the spatial clustering, which uses the current frame only. It is observable that STHS describes the appearing man from the right more concisely with fewer segments.

STHS consists of spatial agglomerative clustering and temporal graph matching techniques. In the spatial clustering, all color, motion, and boundary features are used to merge superpixels in each frame. On the other hand, only color features are used in the temporal graph matching between frames, since motion and boundary features are not temporally coherent.

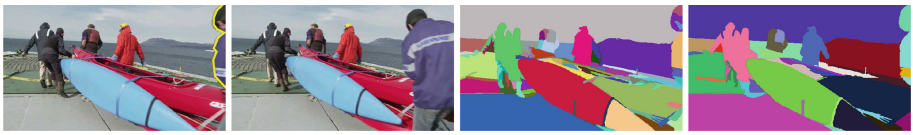
For each frame $t$ in the short-term window $\tau - \alpha \leq t \leq \tau + \alpha$, we adopt the simple agglomerative clustering [36] to merge the most similar pair of clusters iteratively. First, we define a graph $G^{(t)} = (V^{(t)}, E^{(t)})$ for frame $t$, where $V^{(t)} = \{x_1, \ldots, x_N\}$ is the set of nodes and $E^{(t)} = \{e_{ij}\}$ is the set of edges. The superpixels become the nodes. If two superpixels $x_i$ and $x_j$ share a boundary, they are connected by edge $e_{ij}$. Note that these graphs $\{G^{(1)}, \ldots, G^{(T)}\}$ are also used in the MRF optimization in Sect. 3.3, where $T$ denotes the number of frames in an input video. To perform the agglomerative clustering at frame $t$, we initially regard superpixels $\{x_1, \ldots, x_N\}$ as individual clusters $\{c_1, \ldots, c_N\}$. We measure the distances between these clusters by

$$d(c_i, c_j) = \begin{cases} d_{\chi^2}(x_i, x_j) & \text{if } e_{ij} \in E^{(t)}, \\ \infty & \text{otherwise,} \end{cases} \tag{1}$$

where $d_{\chi^2}$ denotes the chi-square distance between $x_i$ and $x_j$ in the feature space. We use the color feature $\mathbf{h}^{\mathrm{c}}$, motion feature $\mathbf{h}^{\mathrm{m}}$, and boundary feature $\mathbf{h}^{\mathrm{b}}$ by concatenating them. We normalize the concatenated feature again. Then, we iteratively merge the two clusters $c_i$ and $c_j$ that yield the minimum distance. The mergence yields a new cluster $c_n$. The distance between the new cluster $c_n$ and an existing cluster $c_k$ is updated by

$$d(c_n, c_k) = \min\{d(c_i, c_k), d(c_j, c_k)\} \tag{2}$$

according to the single link algorithm [36]. We terminate the merging when the minimum distance is higher than a threshold $\gamma$. Notice that this threshold $\gamma$ controls the segmentation granularity. Finally, we reassign the cluster indices



(a) Frame $\tau$ (b) Frame $\tau + \alpha$ (c) Spatial clustering (d) STHS

**Fig. 3.** Efficacy of STHS in comparison with the spatial clustering. The newly appearing man at frame $\tau$ is depicted by yellow boundaries. While the spatial agglomerative clustering divides the man into unnecessarily many segments, STHS represents him concisely with fewer segments by exploiting the information in the future frame $\tau + \alpha$

from 1 to the number of clusters. Let $c_u^{(t)}$ denote the resultant $u$th cluster at frame $t$.

After the intra-frame agglomerative clustering, we link the clusters temporally in the short-term window. To this end, we perform the temporal matching between two frames, $t$ and $t+1$, sequentially for $\tau - \alpha \leq t \leq \tau + \alpha - 1$. We first construct a bipartite graph $G^{(t,t+1)} = (U^{(t)}, U^{(t+1)}, E^{(t,t+1)})$, where $U^{(t)} = \{c_1^{(t)}, \ldots, c_N^{(t)}\}$ is the set of nodes at frame $t$. The clusters, produced by the intra-frame agglomerative clustering, become the nodes. $E^{(t,t+1)} = \{e_{uv}^{(t,t+1)}\}$ is the set of inter-frame edges. Node $c_u^{(t)}$ at frame $t$ is connected to node $c_v^{(t+1)}$ by edge $e_{uv}^{(t,t+1)}$, if at least one pixel within $c_u^{(t)}$ is mapped to a pixel within $c_v^{(t+1)}$ according to the forward or backward optical flow. Then, edge $e_{uv}^{(t,t+1)}$ is assigned an affinity weight, given by

$$w_{uv}^{(t,t+1)} = \begin{cases} \eta(c_u^{(t)}, c_v^{(t+1)}) & \text{if } e_{uv}^{(t,t+1)} \in E^{(t,t+1)}, \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

where $\eta$ is a similarity function between the two clusters. It is defined as

$$\eta(c_u^{(t)}, c_v^{(t+1)}) = \eta_{\mathrm{c}}(c_u^{(t)}, c_v^{(t+1)}) \times \eta_{\mathrm{o}}(c_u^{(t)}, c_v^{(t+1)}) \tag{4}$$

where $\eta_{\mathrm{c}}$ and $\eta_{\mathrm{o}}$ are color and overlap similarities, respectively. We measure the color similarity by

$$\eta_{\mathrm{c}}(c_u^{(t)}, c_v^{(t+1)}) = \exp\left(-d_{\chi^2}(c_u^{(t)}, c_v^{(t+1)})\right) \tag{5}$$

in which $d_{\chi^2}$ denotes the chi-square distance between the color features for the two clusters. As mentioned previously, for the inter-frame matching, we do not use the motion and boundary features due to their inter-frame irrelevance. We compute the overlap similarity by

$$\eta_{\mathrm{o}}(c_u^{(t)}, c_v^{(t+1)}) = \frac{1}{2}\left(\frac{|\mathcal{P}_u^{(t)} \cap \overleftarrow{\mathcal{P}}_v^{(t+1)}|}{\max_k |\mathcal{P}_k^{(t)} \cap \overleftarrow{\mathcal{P}}_v^{(t+1)}|} + \frac{|\overrightarrow{\mathcal{P}}_u^{(t)} \cap \mathcal{P}_v^{(t+1)}|}{\max_k |\overrightarrow{\mathcal{P}}_u^{(t)} \cap \mathcal{P}_k^{(t+1)}|}\right) \tag{6}$$

where $\mathcal{P}_u^{(t)}$ is the set of pixels that belongs to cluster $c_u^{(t)}$. Also, $\overrightarrow{\mathcal{P}}_u^{(t)}$ is the set of pixels at frame $t+1$, which are mapped from the pixels in $\mathcal{P}_u^{(t)}$ by the forward

optical flow vectors. Symmetrically, $\overleftarrow{\mathcal{P}}_v^{(t+1)}$ is the set of pixels at frame $t$, which are mapped from $\mathcal{P}_u^{(t+1)}$ by the backward vectors. The operator $|\cdot|$ returns the number of elements in a set. Note that a higher similarity $\eta_{\mathrm{o}}(c_u^{(t)}, c_v^{(t+1)})$ is assigned, as the two clusters are more overlapped by the forward or backward warping.

To represent temporal matching results, we define a matching variable $\mu_{uv}^{(t,t+1)}$, which equals 1 if $c_u^{(t)}$ is matched to $c_v^{(t+1)}$, and 0 otherwise. To determine the set of matching variables $\mathcal{M} = \{\mu_{uv}^{(t,t+1)}\}$, we maximize the objective function

$$\max_{\mathcal{M}} \sum_{t=\tau-\alpha}^{\tau+\alpha-1} \sum_{u \in U^{(t)}} \sum_{v \in U^{(t+1)}} \mu_{uv}^{(t,t+1)} \times w_{uv}^{(t,t+1)} \tag{7}$$

subject to the constraints

$$\sum_{v \in U^{(t+1)}} \mu_{uv}^{(t,t+1)} \leq 1, \quad \mu_{uv}^{(t,t+1)} \in \{0, 1\}. \tag{8}$$

This constrained maximization can be easily solved by performing the greedy bipartite matching from $t = \tau - \alpha$ to $t = \tau + \alpha - 1$ sequentially. In other words, for each cluster at frame $t$, we match it to the cluster at frame $t+1$ that is connected with the highest affinity. However, to reflect occlusion scenarios, if all affinities between cluster $c_u^{(t)}$ and clusters at frame $t+1$ are smaller than a threshold, we do not match $c_u^{(t)}$ to any cluster at frame $t+1$ and $\sum_{v \in U^{(t+1)}} \mu_{uv}^{(t,t+1)} = 0$. After the temporal matching, we assign an identical label to the set of clusters that are connected according to the matching variables. Finally, the label of a cluster becomes the initial segment labels of all superpixels that the cluster includes. Let $s_i^{(t)}$ denote the initial segment that includes superpixel $x_i$ at frame $t$. Notice that, in the following process, we only use the initial segment results at frame $\tau$.

Figure 4 exemplifies the temporal matching of inter-clusters. The temporal matching assigns segment labels in a temporally coherent manner, and groups

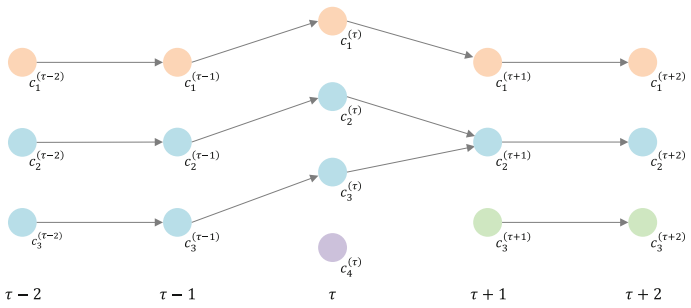



**Fig. 4.** An example of the temporal graph matching with $\alpha = 2$. Each circle denotes a cluster. The clusters in an identical color have the same label. $c_4^{(\tau)}$ is not matched to any cluster at frame $\tau + 1$. While $c_2^{(\tau)}$ and $c_3^{(\tau)}$ are not merged in the intra-frame clustering, they are assigned the same label after the temporal matching

**Algorithm 1.** Short-Term Hierarchical Segmentation (STHS)

**Input:** Superpixels in a window of frames from $\tau - \alpha$ to $\tau + \alpha$
**for** frame $t = \tau - \alpha$ to $\tau + \alpha$ **do**
  Set each superpixel as an individual cluster
  Compute the distance between each pair of neighboring superpixels ▷ (1)
  **repeat**
    Find a cluster pair of the minimum distance
    Merge the two clusters into a new cluster
    Update the distances between the new cluster and the existing clusters ▷ (2)
  **until** the minimum distance is higher than $\gamma$
**end for**
**for** frame $t = \tau - \alpha$ to $\tau + \alpha - 1$ **do**
  Construct a bipartite graph for the clusters in two consecutive frames $t$ and $t+1$ ▷ (3)
  Perform the inter-frame matching between clusters ▷ (7)
**end for**
Assign an identical label to each set of connected clusters
**Output:** Initial segment label of each superpixel in the window of frames

some clusters in a frame that are not merged in the agglomerative clustering. Thus, by employing STHS, we can detect newly appearing segments and also group distant superpixels that compose the same segment. Algorithm 1 summarizes the proposed STHS scheme.

### 3.3 MRF Optimization

Next, we partition the current frame $\tau$ into segments by employing the initial segments, which are obtained by STHS. To this end, we develop an MRF optimization scheme. We use the graph $G^{(\tau)} = (V^{(\tau)}, E^{(\tau)})$, which is already constructed for the intra-frame agglomerative clustering in Sect. 3.2. The node set $V^{(\tau)} = \{x_1^{(\tau)}, \ldots, x_N^{(\tau)}\}$ consists of the superpixels at frame $\tau$. We define a variable $y_i^{(\tau)}$ to indicate the label of node $x_i^{(\tau)}$. By combining unary and pairwise costs, the MRF energy function is defined as

$$\mathcal{E}(\mathbf{y}^{(\tau)}) = \sum_{i \in V^{(\tau)}} \psi(x_i^{(\tau)}, y_i^{(\tau)}) + \lambda \times \sum_{(i,j) \in E^{(\tau)}} \phi(x_i^{(\tau)}, x_j^{(\tau)}, y_i^{(\tau)}, y_j^{(\tau)}) \tag{9}$$

where $\lambda$ controls the relative importance between the unary and pairwise costs.

The unary cost is given by

$$\psi(x_i^{(\tau)}, l) = -\log p(l \,|\, x_i^{(\tau)}) \tag{10}$$

$$= -\log \frac{\theta_{\mathrm{s}}(x_i^{(\tau)}, l) + \theta_{\mathrm{t}}(x_i^{(\tau)}, l)}{\sum_k (\theta_{\mathrm{s}}(x_i^{(\tau)}, k) + \theta_{\mathrm{t}}(x_i^{(\tau)}, k))} \tag{11}$$

where $p(l \mid x_i^{(\tau)})$ is the probability that node $x_i^{(\tau)}$ belongs to the $l$th segment. In other words, the unary cost $\psi(x_i^{(\tau)}, l)$ gets lower, as node $x_i^{(\tau)}$ is more likely to be labeled as $l$. In (11), $\theta_{\mathrm{s}}$ and $\theta_{\mathrm{t}}$ are the STHS similarity function and the temporal consistency function, respectively.

Although STHS provides robust initial segmentation results, they are not harmonized with the labels at the previous frame $\tau - 1$. Hence, for each initial segment at the current frame $\tau$, we check if it is consistent with each label $l$ at frame $\tau - 1$. More specifically, we compute the STHS similarity function $\theta_{\mathrm{s}}(x_i^{(\tau)}, l)$ by

$$\theta_{\mathrm{s}}(x_i^{(\tau)}, l) = \begin{cases} \eta(z_l^{(\tau-1)}, s_i^{(\tau)}) & \text{if } \max_k \eta(z_k^{(\tau-1)}, s_i^{(\tau)}) > \beta, \\ 0 & \text{otherwise,} \end{cases} \tag{12}$$

where $\eta$ is the similarity function in (4), and the threshold $\beta$ is 0.5. Also, $z_l^{(\tau-1)}$ denotes the segment that has label $l$ at frame $\tau - 1$. If the initial segment $s_i^{(\tau)}$ including superpixel $x_i^{(\tau)}$ is similar to the segment $z_l^{(\tau-1)}$, the function $\theta_{\mathrm{s}}(x_i^{(\tau)}, l)$ yields a high value. Moreover, we generate new labels to consider newly appearing segments. Specifically, if $\max_k \eta(z_k^{(\tau-1)}, s_i^{(\tau)}) \leq \beta$, we declare that $s_i^{(\tau)}$ is not harmonized with any existing label at the previous frame. For this inharmonic initial segment, we assign a new label $\hat{l}$ and set the STHS similarity by $\theta_{\mathrm{s}}(x_i^{(\tau)}, \hat{l}) = 1$. Note that we regard all initial segments at the first frame as inharmonic.

To enforce temporal coherence of inter-frame segments, we adopt the temporal consistency function $\theta_{\mathrm{t}}(x_i^{(\tau)}, l)$ in the unary cost in (11), which is given by

$$\theta_{\mathrm{t}}(x_i^{(\tau)}, l) = \exp\left(-d_{\chi^2}(\overleftarrow{x}_i^{(\tau)}, x_i^{(\tau)})\right) \times \frac{|\mathcal{Z}_l^{(\tau-1)} \cap \overleftarrow{\mathcal{X}}_i^{(\tau)}|}{\max_k |\mathcal{Z}_k^{(\tau-1)} \cap \overleftarrow{\mathcal{X}}_i^{(\tau)}|} \tag{13}$$

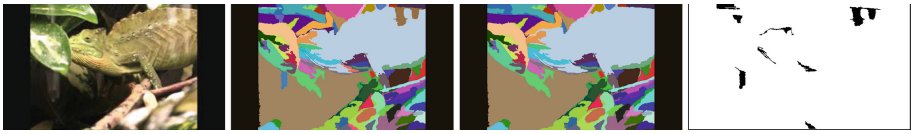
where $\overleftarrow{x}_i^{(\tau)}$ denotes the superpixel at frame $\tau - 1$, which is warped from $x_i^{(\tau)}$ by the backward optical flow vectors, and $\overleftarrow{\mathcal{X}}_i^{(\tau)}$ is the set of pixels in $\overleftarrow{x}_i^{(\tau)}$. Also, $\mathcal{Z}_l^{(\tau-1)}$ is the set of pixels within the $l$th segment at frame $\tau - 1$. The chi-square distance $d_{\chi^2}(\overleftarrow{x}_i^{(\tau)}, x_i^{(\tau)})$ is computed using only the color features. Note that $\theta_{\mathrm{t}}(x_i^{(\tau)}, l)$ yields a higher value when the color matching error is smaller and the warped area $\overleftarrow{\mathcal{X}}_i^{(\tau)}$ has a bigger overlap with the $l$th segment $\mathcal{Z}_l^{(\tau-1)}$. Thus, the temporal consistency function helps to propagate the segment labels at the previous frame to the current frame.

To encourage neighboring nodes with similar features to have the same segment label, we define the pairwise cost in (9) by

$$\phi(x_i^{(\tau)}, x_j^{(\tau)}, y_i^{(\tau)}, y_j^{(\tau)}) = \begin{cases} \exp(-d_{\chi^2}(x_i^{(\tau)}, x_j^{(\tau)})) & \text{if } y_i^{(\tau)} \neq y_j^{(\tau)}, \\ 0 & \text{otherwise,} \end{cases} \tag{14}$$

where $d_{\chi^2}$ is computed using all the color, motion, and boundary features.

We employ the graph-cut algorithm [37] to minimize the MRF energy function in (9). Consequently, we obtain the segment label of each superpixel at the current frame. This segmentation result is recorded for segmenting the next frame.



(a) Input frame (b) Before SLE (c) After SLE (d) Short-labels

**Fig. 5.** An example of the short-label elimination (SLE). Noisy labels with short durations are depicted in black in (d). They are erased, and the corresponding superpixels are re-labeled using the neighboring labels in (c). The frames are from "Chameleons"

### 3.4 Short-Label Elimination

In general, an online segmentation algorithm may produce noisy segments, which have short temporal durations. To suppress such noise, we develop the short-label elimination scheme. At frame $\tau$, we check the temporal duration of each segment at frame $\tau - \epsilon$, where $\epsilon = 10$. If the duration is shorter than $\epsilon$, we erase the labels of the corresponding superpixels. To re-label these erased superpixels, we apply the MRF optimization scheme again. For a non-erased superpixel, its unary cost is set to 0 for the original label, and 1 for the other labels. For an erased superpixel, its unary cost is set to 1 for all labels. Also, we use the same pairwise cost in (14). Then, we minimize the energy function using the graph-cut algorithm. Consequently, the erased superpixels are labeled consistently with the neighboring superpixels. Figure 5 shows an example of the short-label elimination.

## 4 Experimental Results

We test the proposed video segmentation algorithm on the VSB100 dataset [14], which consists of 40 training videos and 60 test videos. The spatial resolution of these video sequences are between $960 \times 720$ and $1920 \times 1080$. The ground-truth is annotated for every 20th frame by four subjects. The VSB100 sequences are very challenging due to motion blur, jerky camera motion, occlusion, object deformation, and ambiguous object boundaries. For efficient computation, we test the proposed algorithm after resizing video frames by a factor of 0.5 in both $x$ and $y$ directions. We use the same parameters for all experiments, unless otherwise specified.

We use two performance metrics, boundary precision-recall (BPR) and volume precision-recall (VPR), which were introduced in [14]. BPR measures the qualities of segmentation boundaries in the precision-recall framework after

the bipartite graph matching between computer-generated boundaries and the ground-truth boundaries. VPR assesses volumetric qualities of segmentation by computing the maximal overlap between computer-generated segments and the ground-truth segments. For both BPR and VPR, we calculate the average precision (AP), which is the area under the precision-recall curve. We report the optimal dataset scale (ODS) performance and the optimal segmentation scale (OSS) performance according to the aggregation strategy of F-measure scores. While ODS aggregates the scores of all sequences at a fixed segmentation scale, OSS discovers the optimal scale for each sequence. Hence, OSS yields a higher score than ODS. The proposed algorithm controls the scale of segmentation using the spatial merging threshold $\gamma \in \{0.1, \ldots, 0.6\}$, which is more practical than specifying the number of segments. Figure 6 visualizes segmentation results of the proposed algorithm according to the scale parameter $\gamma$. As $\gamma$ increases, more superpixels are merged, resulting in coarser segments. In addition, we count the number of segments (NCL) and compute the average length ($\mu$) and the standard deviation ($\delta$) of segment durations in ODS. We analyze running times of the proposed algorithm and the conventional methods [5–7,9,11] by seconds per frame (SPF) for "Arctic Kayak" sequence at $640 \times 360$ resolution. We test the methods on a PC with a 3.0 GHz CPU.

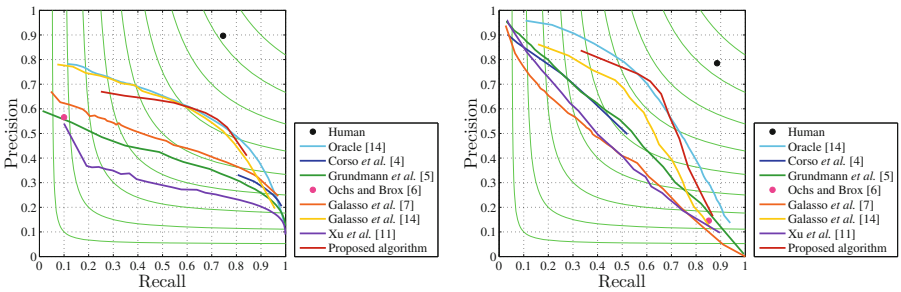
(a) Input frame (b) Ground-truth (c) $\gamma = 0.1$ (d) $\gamma = 0.2$ (e) $\gamma = 0.4$

**Fig. 6.** Segmentation results of the proposed algorithm in various scales according to the parameter $\gamma$. As $\gamma$ increases, the proposed algorithm generates coarser segments. The frames are from "Fish Underwater"

Table 1 compares the performance of the proposed algorithm on the VSB100 dataset with those of 12 conventional algorithms: nine offline methods [4–10,12,14] and three online methods [11–13]. The scores of the conventional algorithms are from [9,10,12,13]. The oracle method links the per-frame UCM segments [1] optimally using the ground-truth data as specified in [14]. We see that the proposed algorithm surpasses the conventional online video segmentation algorithms in terms of both BPR and VPR. Especially, in terms of VPR ODS, the proposed algorithm provides a 20 % gain, compared with the state-of-the-art online algorithm [12]. Moreover, the proposed algorithm even outperforms most offline video segmentation algorithms and provides comparable performances to the state-of-the-art offline algorithms [9,10]. In addition, we develop a faster version, which shortens the overall running time from 176.1 to 18.6 by employing lighter optical flow estimator, [32] without matching, and faster contour detector [34]. The faster version reduces the running time by 89 %, while sacrificing a small BPR-AP score only. The faster version of the proposed algorithm is faster than all conventional methods.

**Table 1.** Comparison of video segmentation performances on the VSB100 [14]. The best and the second best results are boldfaced and underlined, respectively

| Algorithm | BPR | | | VPR | | | Length | NCL | Time |
|---|---|---|---|---|---|---|---|---|---|
| | ODS | OSS | AP | ODS | OSS | AP | $\mu(\delta)$ | $\mu$ | SPF |
| Human | 0.81 | 0.81 | 0.67 | 0.83 | 0.83 | 0.70 | 83.24(40.04) | 11.90 | - |
| Oracle [14] | 0.61 | 0.67 | 0.61 | 0.65 | 0.67 | 0.68 | - | 118.56 | - |
| A. Offline segmentation algorithms | | | | | | | | | |
| Corso *et al.* [4] | 0.51 | 0.53 | 0.37 | 0.51 | 0.52 | 0.38 | 70.67(48.39) | 25.83 | - |
| Grundmann *et al.* [5] | 0.47 | 0.54 | 0.41 | 0.52 | 0.55 | 0.52 | 51.83(39.91) | 117.90 | **26.8** |
| Ochs and Brox [6] | 0.17 | 0.17 | 0.06 | 0.25 | 0.25 | 0.12 | 87.85(38.83) | 3.73 | 268.9 |
| Galasso *et al.* [7] | 0.51 | 0.56 | 0.45 | 0.45 | 0.51 | 0.42 | 80.17(37.56) | 8.00 | 425.6 |
| Galasso *et al.* [14] | 0.61 | 0.65 | 0.59 | 0.59 | 0.62 | 0.56 | 25.50(36.48) | 258.05 | - |
| Galasso *et al.* [12] | 0.62 | 0.66 | 0.54 | 0.55 | 0.59 | 0.55 | 61.25(40.87) | 80.00 | - |
| Khoreva *et al.* [8] | 0.61 | 0.64 | 0.51 | 0.58 | 0.61 | 0.58 | 60.48(43.19) | 50.00 | - |
| Khoreva *et al.* [9] | **0.64** | **0.70** | **0.61** | 0.63 | 0.66 | 0.63 | 83.41(35.27) | 50.00 | 416.2 |
| Yi and Pavlovic [10] | 0.63 | 0.67 | 0.57 | **0.65** | **0.67** | **0.64** | 35.76(38.72) | 168.93 | - |
| B. Online segmentation algorithms | | | | | | | | | |
| Xu *et al.* [11] | 0.38 | 0.46 | 0.32 | 0.45 | 0.48 | 0.44 | 59.27(47.76) | 26.58 | 39.2 |
| Galasso *et al.* [12] | 0.61 | **0.67** | 0.52 | 0.55 | 0.59 | 0.53 | 73.31(40.33) | 15.63 | - |
| Li *et al.* [13] | 0.54 | 0.58 | 0.40 | 0.53 | 0.60 | 0.46 | - | - | - |
| Proposed | **0.63** | 0.66 | **0.53** | **0.66** | **0.68** | **0.62** | 36.61(31.19) | 133.22 | 176.1 |
| Proposed (Faster ver.) | **0.63** | 0.66 | 0.51 | **0.66** | **0.68** | **0.62** | 37.01(31.27) | 140.97 | **18.6** |




(a) BPR (b) VPR

**Fig. 7.** Comparison of the precision-recall curves of the proposed algorithm and the conventional algorithms [4–7,11,14]

Figure 7 shows the precision-recall curves of BPR and VPR. We compare the proposed algorithm with the conventional algorithms [4–7,11,14], whose results are available in the benchmark [14]. Among them, only the proposed algorithm and [11] are online ones, and the others are offline ones. The curves of the proposed algorithm are mostly higher than those of the conventional algorithms.

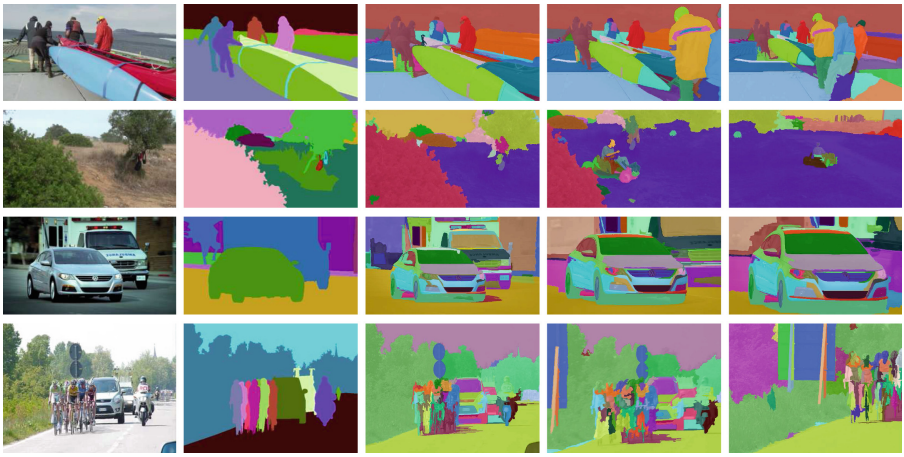



**Fig. 8.** Qualitative results of the proposed segmentation algorithm. (a), (b), and (c) show the input frame, the ground-truth, and the computed results at the same frames, respectively. (d) and (e) illustrate more segmentation results at different frames. From top to bottom, the frames are from "Arctic Kayak," "Gokart," "VW Commercial," and "Bicycle Race" in the VBS100 dataset [14]

Furthermore, the proposed algorithm partly outperforms the oracle method, which uses the ground-truth data.

Figure 8 exemplifies segmentation results of the proposed algorithm in OSS. It is observable that the proposed algorithm yields spatially accurate and temporally coherent segments. Especially, the proposed algorithm robustly identifies newly appearing segments on the "Arctic Kayak" and "Gokart" sequences. Also, the proposed algorithm provides successful results on "VW Commercial" and "Bicycle Race," even though there are fast camera motions. Due to the page limitation, we provide more segmentation results as supplementary materials.

Next, in Table 2, we analyze the efficacy of each energy term in the MRF optimization. We test three configurations: 'STHS + Pairwise,' 'Temporal + Pairwise,' and 'STHS + Temporal.' First, in 'STHS + Pairwise,' we omit the temporal consistency function $\theta_{\mathrm{t}}$ in (11). Second, 'Temporal + Pairwise' does not perform STHS and ignores the STHS similarity function $\theta_{\mathrm{s}}$ in (11). Third, in 'STHS + Temporal,' we omit the pairwise cost in the MRF optimization. Note that the omission of the temporal consistency function in 'STHS + Pairwise' leads to worse VPR scores. Since the proposed STHS plays an essential role in handling newly appearing segments and alleviating the propagation of erroneous segmentation labels, 'Temporal + Pairwise' presents the worst scores. Also, the omission of the pairwise term in 'STHS + Temporal' leads to the performance degradation. However, since we consider spatial affinities in STHS, the degradation is relatively small.

**Table 2.** The segmentation performance of the proposed algorithm using various experimental configurations

| | BPR | | | VPR | | | Length | NCL |
|---|---|---|---|---|---|---|---|---|
| Experimental setting | ODS | OSS | AP | ODS | OSS | AP | $\mu(\delta)$ | $\mu$ |
| Proposed algorithm | 0.63 | 0.66 | 0.53 | 0.66 | 0.68 | 0.62 | 36.61(31.19) | 133.22 |
| A. Combination of energy functions | | | | | | | | |
| STHS + Pairwise | 0.62 | 0.66 | 0.55 | 0.64 | 0.66 | 0.59 | 34.02(28.69) | 119.45 |
| Temporal + Pairwise | 0.57 | 0.61 | 0.45 | 0.61 | 0.64 | 0.55 | 65.64(40.38) | 39.02 |
| STHS + Temporal | 0.62 | 0.65 | 0.52 | 0.65 | 0.67 | 0.62 | 35.43(30.26) | 125.68 |
| B. Post processing (short-label elimination) | | | | | | | | |
| Without post processing | 0.63 | 0.66 | 0.52 | 0.66 | 0.68 | 0.62 | 13.54(23.49) | 345.83 |

To analyze the effectiveness of the short-label elimination, we measure the performance of the proposed algorithm without the post processing. As reported in Table 2, the short-label elimination extends the average duration of segments and decreases the number of segments, by eliminating noisy segments. It does not increase BPR and VPR significantly, since the noisy segments are too small to affect on the quantitative results.

## 5 Conclusions

We proposed a novel online video segmentation algorithm. To identify newly appearing segments effectively, we introduced the STHS technique, which generates initial segments by sliding a window of frames. We first employed the spatial agglomerative clustering for each frame, and then performed the temporal bipartite graph matching across frames. Moreover, we defined the MRF energy function, which consists of the unary and pairwise costs. We computed the unary cost to exploit the initial STHS result and the previous segmentation result, and the pairwise cost to encourage similar superpixels to have the same label. Experimental results on the VSB100 dataset [14] showed that the proposed algorithm outperforms the state-of-the-art online video segmentation algorithms [11–13] significantly.

**Acknowledgement.** This work was supported partly by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. NRF-2015R1A2A1A10055037), and partly by the MSIP, Korea, under the ITRC support program supervised by the Institute for Information & communications Technology Promotion (No. IITP-2016-R2720-16-0007).

## References


1. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **33**(5), 898–916 (2011)
2. Yu, Y., Fang, C., Liao, Z.: Piecewise flat embedding for image segmentation. In: ICCV, pp. 1368–1376 (2015)
3. Xie, S., Tu, Z.: Holistically-nested edge detection. In: ICCV, pp. 1395–1403 (2015)
4. Corso, J.J., Sharon, E., Dube, S., El-Saden, S., Sinha, U., Yuille, A.: Efficient multilevel brain tumor segmentation with integrated bayesian model classification. IEEE Trans. Med. Imaging **27**(5), 629–640 (2008)
5. Grundmann, M., Kwatra, V., Han, M., Essa, I.: Efficient hierarchical graph-based video segmentation. In: CVPR, pp. 2141–2148 (2010)
6. Ochs, P., Brox, T.: Object segmentation in video: a hierarchical variational approach for turning point trajectories into dense regions. In: ICCV, pp. 1583–1590 (2011)
7. Galasso, F., Cipolla, R., Schiele, B.: Video segmentation with superpixels. In: Lee, K.M., Matsushita, Y., Rehg, J.M., Hu, Z. (eds.) ACCV 2012, Part I. LNCS, vol. 7724, pp. 760–774. Springer, Heidelberg (2013)
8. Khoreva, A., Galasso, F., Hein, M., Schiele, B.: Learning must-link constraints for video segmentation based on spectral clustering. In: Jiang, X., Hornegger, J., Koch, R. (eds.) CPR 2014. LNCS, vol. 8753, pp. 701–712. Springer, Heidelberg (2014)
9. Khoreva, A., Galasso, F., Hein, M., Schiele, B.: Classifier based graph construction for video segmentation. In: CVPR, pp. 951–960 (2015)
10. Yi, S., Pavlovic, V.: Multi-cue structure preserving MRF for unconstrained video segmentation. In: ICCV, pp. 3262–3270 (2015)
11. Xu, C., Xiong, C., Corso, J.J.: Streaming hierarchical video segmentation. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part VI. LNCS, vol. 7577, pp. 626–639. Springer, Heidelberg (2012)
12. Galasso, F., Keuper, M., Brox, T., Schiele, B.: Spectral graph reduction for efficient image and streaming video segmentation. In: CVPR, pp. 49–56 (2014)
13. Li, C., Lin, L., Zuo, W., Yan, S., Tang, J.: SOLD: sub-optimal low-rank decomposition for efficient video segmentation. In: CVPR, pp. 5519–5527 (2015)
14. Galasso, F., Nagaraja, N., Cardenas, T., Brox, T., Schiele, B.: A unified video segmentation benchmark: annotation, metrics and analysis. In: CVPR, pp. 3527–3534 (2013)
15. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: analysis and an algorithm. Adv. Neural Inf. Process. Syst. **2**, 849–856 (2002)
16. Yu, C.P., Le, H., Zelinsky, G., Samaras, D.: Efficient video segmentation using parametric graph partitioning. In: ICCV, pp. 3155–3163 (2015)
17. Vazquez-Reina, A., Avidan, S., Pfister, H., Miller, E.: Multiple hypothesis video segmentation from superpixel flows. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part V. LNCS, vol. 6315, pp. 268–281. Springer, Heidelberg (2010)
18. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. Int. J. Comput. Vis. **59**(2), 167–181 (2004)
19. Chang, J., Wei, D., Fisher, J.: A video representation using temporal superpixels. In: CVPR, pp. 2051–2058 (2013)
20. Reso, M., Jachalsky, J., Rosenhahn, B., Ostermann, J.: Temporally consistent superpixels. In: ICCV, pp. 385–392 (2013)

21. Shi, J., Malik, J.: Motion segmentation and tracking using normalized cuts. In: ICCV, pp. 1154–1160 (1998)
22. Brox, T., Malik, J.: Object segmentation by long term analysis of point trajectories. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part V. LNCS, vol. 6315, pp. 282–295. Springer, Heidelberg (2010)
23. Ochs, P., Brox, T.: Higher order motion models and spectral clustering. In: CVPR, pp. 614–621 (2012)
24. Lee, Y.J., Kim, J., Grauman, K.: Key-segments for video object segmentation. In: ICCV, pp. 1995–2002 (2011)
25. Ma, T., Latecki, L.J.: Maximum weight cliques with mutex constraints for video object segmentation. In: CVPR, pp. 670–677 (2012)
26. Zhang, D., Javed, O., Shah, M.: Video object segmentation through spatially accurate and temporally dense extraction of primary object regions. In: CVPR, pp. 628–635 (2013)
27. Oneata, D., Revaud, J., Verbeek, J., Schmid, C.: Spatio-temporal object detection proposals. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014, Part III. LNCS, vol. 8691, pp. 737–752. Springer, Heidelberg (2014)
28. Wang, W., Shen, J., Porikli, F.: Saliency-aware geodesic video object segmentation. In: CVPR, pp. 3395–3402 (2015)
29. Giordano, D., Murabito, F., Palazzo, S., Spampinato, C.: Superpixel-based video object segmentation using perceptual organization and location prior. In: CVPR, pp. 4814–4822 (2015)
30. Taylor, B., Karasev, V., Soatto, S.: Causal video object segmentation from persistence of occlusions. In: CVPR, pp. 4268–4276 (2015)
31. Jang, W.D., Lee, C., Kim, C.S.: Primary object segmentation in videos via alternate convex optimization of foreground and background distributions. In: CVPR, pp. 696–704 (2016)
32. Weinzaepfel, P., Revaud, J., Harchaoui, Z., Schmid, C.: DeepFlow: large displacement optical flow with deep matching. In: ICCV, pp. 1385–1392 (2013)
33. Comaniciu, D., Meer, P.: Mean shift: a robust approach toward feature space analysis. IEEE Trans. Pattern Anal. Mach. Intell. **24**(5), 603–619 (2002)
34. Dollár, P., Zitnick, C.L.: Fast edge detection using structured forests. IEEE Trans. Pattern Anal. Mach. Intell. **37**(8), 1558–1570 (2015)
35. Li, F., Kim, T., Humayun, A., Tsai, D., Rehg, J.M.: Video segmentation by tracking many figure-ground segments. In: ICCV, pp. 2192–2199 (2013)
36. Theodoridis, S., Koutroumbas, K.: Pattern Recognition, 3rd edn. Academic Press, Orlando (2006)
37. Boykov, Y., Veksler, O., Zabih, R.: Efficient approximate energy minimization via graph cuts. IEEE Trans. Pattern Anal. Mach. Intell. **20**(12), 1222–1239 (2001)