Unsupervised Video Object Segmentation with Motion-Based Bilateral Networks

Siyang Li[1,2(✉)], Bryan Seybold[2], Alexey Vorobyov[2], Xuejing Lei[1], and C.-C. Jay Kuo[1]

[1] University of Southern California, Los Angeles, USA
{siyangl,xuejing}@usc.edu, cckuo@sipi.usc.edu
[2] Google AI Perception, Mountain View, USA
{seybold,vorobya}@google.com

Abstract. In this work, we study the unsupervised video object segmentation problem where moving objects are segmented without prior knowledge of these objects. First, we propose a motion-based bilateral network to estimate the background based on the motion pattern of non-object regions. The bilateral network reduces false positive regions by accurately identifying background objects. Then, we integrate the background estimate from the bilateral network with instance embeddings into a graph, which allows multiple frame reasoning with graph edges linking pixels from different frames. We classify graph nodes by defining and minimizing a cost function, and segment the video frames based on the node labels. The proposed method outperforms previous state-of-the-art unsupervised video object segmentation methods against the DAVIS 2016 and the FBMS-59 datasets.

Keywords: Video object segmentation · Bilateral networks
Instance embeddings

1 Introduction

The goal of video object segmentation (VOS) is tracking moving objects with accurate masks. It serves as an important pre-processing step in video understanding. The mask of the target object can assist many vision tasks, such as action recognition and visual effects. There are two VOS scenarios depending on whether the tracked target is indicated or not. The former is called *semi-supervised* VOS while the latter is called *unsupervised* VOS or primary object segmentation. The moving objects being tracked and the remaining regions are referred to as *foreground* and *background*, respectively.

We focus on the unsupervised scenario in this work. Since the target objects are unknown, many unsupervised VOS methods rely on motion cues, i.e., optical

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-01219-9_13) contains supplementary material, which is available to authorized users.

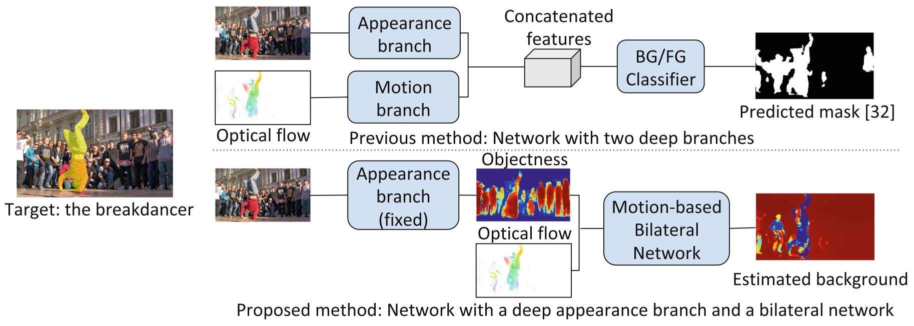
Appearance branch
Motion branch
Concatenated features
BG/FG Classifier
Predicted mask [32]
Optical flow
Previous method: Network with two deep branches
Target: the breakdancer
Appearance branch (fixed)
Objectness
Optical flow
Motion-based Bilateral Network
Estimated background
Proposed method: Network with a deep appearance branch and a bilateral network

Fig. 1. Compare the proposed method with the dual-branch network in [32]. Instead of training a deep motion branch to generate motion features, we propose a light bilateral network based on motion and objectness, which identifies the background by finding regions that have similar motion patterns with low-objectness regions

flow [2,15], to find the objects to be tracked. A commonly used architecture is a dual-branch convolutional neural network [5,16,32], consisting of an appearance branch and a motion branch, which take the RGB frames and optical flow as their input, respectively, as shown at the top of Fig. 1. Although the network is jointly trained with appearance and motion, it may not be able to build the correspondence between foreground and motion patterns. For example, if all cars move in the training data, cars will always be labeled as foreground. The network may map car's appearance to foreground directly while ignoring the motion pattern. Then, static cars may become false positives in inference.

In this work, we aim at building the correspondence between motion patterns and foreground/background directly. A motion-based bilateral network (BNN) for background estimation is proposed to integrate the appearance information (i.e., objectness) and optical flow, as shown at the bottom of Fig. 1. More specifically, the bilateral network is trained to model the motion patterns based on optical flow in the non-object region (inferred by the objectness map from the appearance branch) and identify regions with similar motion patterns. It is worthwhile to point out that bilateral networks have previously been used to propagate masks temporally in [17], but in the proposed method, bilateral networks are adopted to extend the background region spatially from non-object regions to static objects. Then, to leverage the inter-frame similarity, a graph composed of pixels from consecutive frames is constructed, with the similarity defined based on the instance embeddings [9]. Because pixel-wise graph inference can be time-consuming [34], we build the graph on a reduced set of pixels and later propagate the labels to remaining pixels (Fig. 2). According to the experimental results, incorporating information from multiple frames leads to better segmentation results.

We tested the proposed method on the DAVIS 2016 dataset [29] and the FBMS-59 dataset [27]. The experimental results demonstrate superior performance over previous state-of-the-art methods. The contribution includes: (1)

proposing a trainable bilateral network to estimate background based on motion cues and objectness; (2) efficient multi-frame reasoning via graph cut of a reduced set of points seeded across objects; and (3) achieving state-of-the-art results on the DAVIS 2016 and the FBMS-59 datasets with intersection-over-union (IoU) scores of 80.4% and 73.9%, and outperforms previous best results by 1.9% and 2.0%, respectively.

2 Related Work

Unsupervised Video Object Segmentation. Prior to the popularity of deep convolutional neural networks (CNN), some methods used a Gaussian Mixture Model (GMM) and optical flow to build an appearance model and a motion model, respectively [22]. Graph cut is a commonly used technique that integrates the appearance/motion model with temporal/spatial smoothness by optimizing a cost function [28,34]. It is important to formulate the problem in a way so that it can be solved efficiently, e.g. [26]. The proposed method also uses graph cut to jointly optimize a cost function, but we formulate the graph cut over a subset of diverse seed points and rely on deep embeddings from CNNs to better model the similarity between pixels. Later, CNNs, including fully convolutional networks (FCN) that were initially developed for image segmentation [25], were adopted to segment the moving objects in videos. A common approach is training an FCN to generate a binary mask for each frame [3]. A dual branch CNN is used in [5,16,32]. It takes in a frame and its optical flow (or two consecutive frames) to generate appearance and motion features. These features are combined in a later stage and finally the network produces a binary mask. MP [31] relies on optical flow solely to produce the motion-salient regions. Due to the lack of guidance from object masks, it therefore also identifies "moving stuff" such as waves. A common challenge for VOS is the lack of densely annotated videos. FSEG [16] proposes to generate masks from weakly labeled data (bounding boxes). IET [24] transfers the instance embeddings learned from static images and uses heuristics to identify representative embeddings of the moving objects. SfMNet [35] learns object models and motion patterns without mask annotations by differentiable rendering.

Semi-supervised Video Object Segmentation. As mentioned above, for semi-supervised VOS, the object masks for the first frame (or more) are provided, which allow building an accurate appearance model. One common framework for semi-supervised VOS is mask propagation [17,21], where the predicted mask of the previous frame and the new RGB frame are input to the network. The propagation process can be viewed as a modification of the previous frame mask. Another framework for semi-supervised VOS is binary mask prediction conditioned on the user-input frames [3,5,36], which usually requires online fine-tuning for optimal performance. That is, during inference, the network weights are updated by further training on the provided mask. Online fine-tuning is expensive in terms of both runtime and memory, as each query sequence has a unique model stored. Therefore, these approaches do not scale well. Graph cut is also used for semi-supervised VOS [19,26]. BVS [26] samples a regular grid in the bilateral space and conducts graph cut on the grid vertices. CTN [19] uses a CNN to identify confident foreground and background regions based on the provided masks and optical flow. Then a graph is built, taking the color and motion similarity into consideration. PLM [37] compares the deep features of pixels from later frames with the first-frame pixels and determines the foreground probability. It also demands online fine-tuning for optimal performance.

Image Segmentation. Semantic image segmentation and object instance segmentation are two important problems in computer vision and attract a lot attention recently [4,9–11,13,14,25,38]. One popular architecture for image segmentation is the fully convolutional neural network (FCN), which predict a label for each pixel [4,25]. In semantic segmentation, the goal is to assign a semantic category label to each pixel. Building VOS approaches upon the semantic segmentation network has a natural limitation: object instances from the same category cannot be distinguished. Since we may need to track one moving instance from multiple static instances of the same category (e.g. a moving car out from parked cars) in VOS, we rely on instance segmentation networks [6,9], where different instances are distinguishable. In [9], an embedding vector is generated for each pixel and the distance between embedding vectors encode the probability of two pixels belonging to the same instance. IET [24] transfers these embeddings without further fine-tuning on video datasets and choose representative embeddings for foreground and background heuristically for each frame. Then the segmentation is done on each frame individually, so the masks are not temporally smooth enough. In this proposed method, we build a graph of sampled pixels from multiple frames and rely on the embeddings to measure the similarity along the time dimension.

3 Method

The proposed method is explained in detail in this section. An overview is depicted in Fig. 2. It contains three modules: (1) background estimation with a motion-based bilateral network, (2) classification of sampled pixels with graph cut, and (3) frame segmentation by propagating labels of sampled pixels.

3.1 Motion-Based Bilateral Networks for Background Estimation

Motion cues are essential to the solution of the unsupervised VOS problem. In MP [31], a binary segmentation neural network uses the optical flow vector as the input and produces a motion saliency map. Due to the camera motion, flow patterns do not always correspond to moving objects. Consider the following two scenarios: (1) an object with the leftward motion and a static camera, and (2) an object with the rightward motion and a camera following the object. The latter is viewed as left-moving background with a static object. The flow patterns are flipped for the two scenarios yet both expect high motion saliency on objects. In

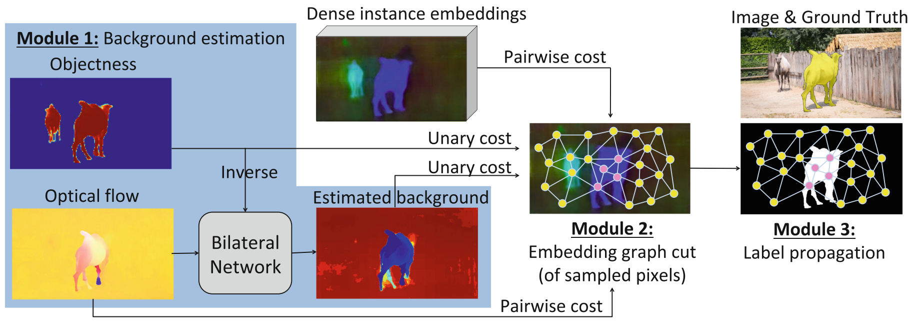

Fig. 2. An overview of the proposed method that consists of three modules. The background is estimated from the inverse objectness map and the optical flow through a bilateral network (BNN) in Module 1, which is enclosed in light blue. Then, an embedding graph that contains sampled pixels (marked by dots) from a set of consecutive frames as vertices is constructed. The unary cost is defined based on the objectness and the estimated background from the BNN. The pairwise cost is from instance embedding and optical flow similarity. All vertices are classified in Module 2 by optimizing the total cost, where magenta and yellow dots denote the foreground and background vertices, respectively. Finally, the graph vertex labels are propagated to all remaining pixels based on embedding similarity in Module 3. Best viewed in color

other words, it is confusing for the network to find motion-salient objects solely relying on optical flow.

To address this problem, we integrate optical flow with objectness scores to estimate the background, which includes static objects and non-objects (also known as *stuff* [12]). Given an objectness map output from a segmentation network [9], we can locate stuff regions by thresholding the objectness map. The optical flow in those regions can model the motion patterns of background (due to the camera motion). By identifying regions of similar motion, we can associate static objects with the background. In other words, background is expanded from stuff regions to include static objects. Inspired by temporal propagation in semi-supervised VOS with the bilateral space [17,26], we solve this background expansion problem with a bilateral network (BNN) [18], i.e. generalized bilateral filtering by replacing default Gaussian filters [1,33] with learnable ones.

Bilateral Filtering. We briefly review the bilateral filtering below and refer to [1,18] for more details. Bilateral filtering is implemented by four steps: (1) constructing a bilateral grid, (2) splatting the features of input samples to the high-dimensional bilateral grid, (3) convolution on the high-dimensional grid, and (4) slicing the filtered features on grid back to samples of interest. Let d and $\mathbf{f}_q \in \mathbb{R}^d$ denote the dimension of the bilateral space and the position vector of sample q in the bilateral space, respectively. In addition, let $s_I(i)$, $s_V(v)$ and $s_O(o)$ denote the feature of an input sample i, a vertex v of the bilateral grid, and an output sample o. We explain the case with the feature being a scalar below, but the process can be generalized to vector features (e.g., for image denoising, the feature is the 3-D color components) by repeating for each feature channel. A bilateral filtering process with $d = 2$ is illustrated in Fig. 3. The 2-D bilateral grid partitions the space into rectangles, and the position of each vertex v can be described by $\mathbf{f}_v \in \mathbb{R}^d$. The feature on the vertex is obtained by accumulating the features of input samples in its neighbor rectangles, in form of

$$s_V(v) = \sum_{i \in \Omega(v)} w(\mathbf{f}_v, \mathbf{f}_i) s_I(i), \tag{1}$$

where $w(\mathbf{f}_v, \mathbf{f}_i)$ is the weight function that defines the influence from sample i on vertex v and $\Omega(v)$ stands for the neighborhood of v. Commonly used weight functions $w(\cdot, \cdot)$ include multilinear interpolation and the nearest neighbor indicator. Afterwards, filters, $c(\cdot)$, are applied to the splatted feature on the grid, as shown in the center figure of Fig. 3:

$$s'_V(v) = \sum_m s_V(v - m) c(m), \tag{2}$$

where $s'_V(v)$ is the filtered feature on vertex v. Finally, the filtered features on vertices are *sliced* to an output sample, o, given its position $\mathbf{f}_o$, as shown in the right figure of Fig. 3. Mathematically, the feature of sample o is obtained by

$$s_O(o) = \sum_{v \in \Omega(o)} w(\mathbf{f}_o, \mathbf{f}_v) s'_V(v), \tag{3}$$

where $\mathit{\Omega}(o)$ represents the set of surrounding vertices, with a set size of 2^d. The weight function $w(\cdot, \cdot)$ is identical with the one in Eq. (1).

The filter $c(\cdot)$ in Eq. (2) is Gaussian in traditional bilateral filtering. It is generalized to learnable filters in [18], which can be trained by minimizing a loss defined between $s_O(o)$ and the target feature of o. The learnable filters compose the bilateral networks (BNN) [17].

Motion-Based Bilateral Networks. A commonly used bilateral space is composed by color components (e.g., RGB) and location indices (x, y), so the 5-D position vector can be written as $\mathbf{f} = (r, g, b, x, y)^T$. For videos, the timestep, t, is often taken as an additional dimension, yielding $\mathbf{f} = (r, g, b, x, y, t)^T$ [17,26].

In the proposed method, we expand static regions spatially based on motion. Therefore, we have $\mathbf{f} = (dx, dy, x, y)^T$, where $(dx, dy)^T$ denotes the optical flow vector. We do not expand the static region temporally because optical flows on consecutive frames are not necessarily aligned. We build a regular grid of size $(G_{flow}, G_{flow}, G_{loc}, G_{loc})$ in this 4-D bilateral space. To obtain a set of input samples for splatting, we locate the stuff pixels by thresholding the objectness map. This set of stuff pixels is the initial background, denoted by $\mathcal{B}^{init}$. We use the inverted objectness score as the feature to be splatted from an input pixel,

$$s_I(i) = 1 - p^{\mathrm{Obj}}(i), i \in \mathcal{B}^{init}. \tag{4}$$

The inverted objectness score can be viewed as a soft vote for the background and the splatting process by Eq. (1) is to accumulate the background votes on

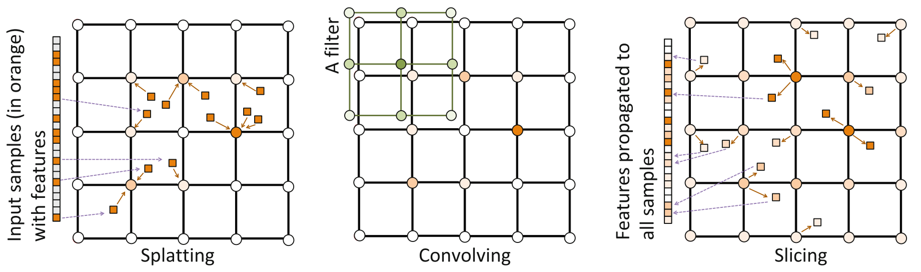
Input samples (in orange) with features
Splatting
A filter
Convolving
Features propagated to all samples
Slicing

Fig. 3. Illustration of a fast bilateral filtering pipeline with a bilateral space of dimension $d = 2$: (1) **splatting** (left): the available features from input samples (orange squares) are accumulated on the bilateral grid; (2) **convolving** (center): the accumulated features on vertices are filtered and propagated to neighbors; and (3) **slicing** (right): the feature of any sample with a known position in the bilateral space can be obtained by interpolation from its surrounding vertices

the vertices of the bilateral grid. After that, a 4-D filter is applied to propagate the votes to neighboring vertices. Finally, by slicing, the propagated votes are forwarded to the remaining pixels on the same frame, based on their optical flow and spatial locations.

To train the BNN, we clip the negative background votes by ReLU and apply the tanh function to convert the clipped votes to background probability,

$$p_{\mathrm{bnn}}^{\mathrm{BG}}(j) = \frac{1 - \exp\{-2 \times ReLU[s_O(j)]\}}{1 + \exp\{-2 \times ReLU[s_O(j)]\}}, \tag{5}$$

and the training loss is the cross entropy between $p_{\mathrm{bnn}}^{\mathrm{BG}}$ and the inverted ground truth mask,

$$\mathcal{L} = -\sum_j [1 - y(j)] \ln p_{\mathrm{bnn}}^{\mathrm{BG}}(j) + y(j) \ln[1 - p_{\mathrm{bnn}}^{\mathrm{BG}}(j)], \tag{6}$$

where $y(j) \in \{0, 1\}$ is the ground truth, with 0 and 1 representing background and foreground, respectively.

3.2 Embedding Graph Cut

Because optical flow computation is imperfect, the estimated background is not a suitable final segmentation. For example, the static parts of non-rigid objects will receive high background scores from the BNN, resulting in false negatives. To achieve more accurate masks, we integrate the predicted background region with the pixel-wise instance embeddings from [9]. The embedding is a representation vector of a pixel, and the Euclidean distance between two embeddings measures the (dis-)similarity between pixels. Here the similarity is defined as the probability of two pixels belonging to the same object instance. Mathematically, the similarity score of two pixels, i and j, is expressed as

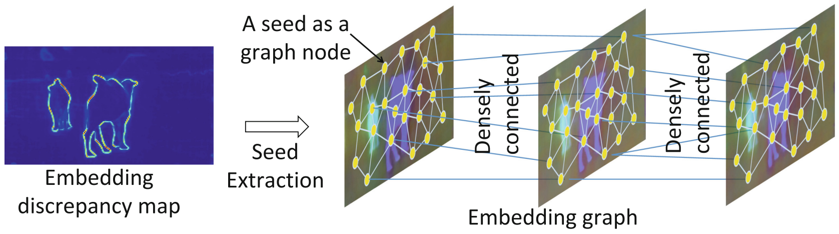

Fig. 4. An illustration of embedding graph construction. We extract seeds based on the embedding discrepancy map, which become graph nodes. Edges connect nodes that are spatial neighbors or from consecutive frames (see texts for the definition of spatial neighbors). Best viewed in color

$$R(i,j) = \frac{2}{1+\exp(||\mathbf{e}_i - \mathbf{e}_j||_2^2)}, \tag{7}$$

where $\mathbf{e}_i$ is the instance embedding vector for pixel i. The instance embeddings are explicitly trained to encode the similarity by minimizing the cross entropy between $R(i,j)$ and the ground truth identical instance indicator in [9].

Given that the instance embeddings describing pixel similarity, the objectness score and background score from the BNN, we adopt the graph cut method to classify pixels by minimizing a cost function. Graph cut has been used previously to solve VOS over either pixel-level graphs [34] or superpixel-level graphs [28]. The former is time consuming considering the number of pixels in a video while the latter is prone to superpixel error. In BVS [26], the graph cut was conducted on a 6-D bilateral grid (composed by color, spatial and temporal positions). However, it is not realistic to build a grid graph in the bilateral space with high dimensional instance embeddings and locations since the splatting/slicing process would be time-consuming and the resulting grid would be sparse.

In the proposed method, we still construct a graph with pixels as vertices. To save computation, the graph is built from a small subset of pixels that are approximately evenly distributed in the spatial space and located at stable points of the instance embedding map. These sampled pixels are called "seeds" and labeled via cost minimization. Then, their labels are propagated to all pixels, which will be explained later.

Building an embedding graph. As aforementioned, building a pixel-level graph leads to time-consuming graph cut. To reduce computation load, we follow the seed extraction process in [24] to find a set of pixels with representative embeddings on every frame. To classify the seeds on frame t, we build a graph based on seeds from frames $(t-1)$ to $(t+1)$, i.e., a temporal window of length 3 centered at t. Using a different temporal window that covers frame t is also possible, as studied in Sect. 4.4. We denote the seed set by $\mathcal{V}$. The next step is to link seeds with edges in the graph. Given the seeds on a frame, we identify the closest seed to every pixel, and we link the seeds with a graph edge if two neighboring pixels are closest to different seeds. These edges are called *spatial* *edges.* For seeds from consecutive frames, they are densely linked to yield *temporal edges.* Other seed pairs are not connected. The graph edges are denoted by $\mathcal{E}$. An illustration of the embedding graph of seeds is displayed in Fig. 4.

Graph Cut. A cut of the embedding graph is obtained by minimizing the following cost function:

$$L = \sum_{i \in \mathcal{V}} \phi(i) + \lambda \sum_{(i,j) \in \mathcal{E}} [l(i) \neq l(j)] \theta(i,j), \tag{8}$$

where $l(i) \in \{0, 1\}$ is the assigned label to pixel i, $\phi(\cdot)$ is the unary cost, and $\theta(\cdot, \cdot)$ is the pairwise cost. The unary cost is given by

$$\phi(i) = [1 - l(i)]\phi^{BG}(i) + l(i)\phi^{FG}(i), \tag{9}$$

where $\phi^{BG}(i)$ and $\phi^{FG}(i)$ are the costs for node i to be labeled as background and foreground, respectively. For background cost, we utilize the background probability from Eq. (5). For foreground cost, it is defined by the objectness score, $p^{\mathrm{Obj}}(i)$, obtained by the segmentation network [9]:

$$\phi^{BG}(i) = -\ln p^{\mathrm{BG}}_{\mathrm{bnn}}(i); \tag{10}$$

$$\phi^{FG}(i) = -\ln p^{\mathrm{Obj}}(i). \tag{11}$$

The pairwise cost encourages similar nodes being assigned to the same category, reducing errors from static parts of non-rigid objects. We consider both instance similarity via embeddings $\mathbf{e}_i$ and the motion similarity via optical flow $\mathbf{m}_i = [dx_i, dy_i]^T$. Specifically, $\theta(i,j)$ is given by

$$\theta(i,j) = \exp(-\frac{||\mathbf{e}_i - \mathbf{e}_j||_2^2}{\sigma_e^2}) + \delta(t_i, t_j)\alpha \exp(-\frac{||\mathbf{m}_i - \mathbf{m}_j||_2^2}{\sigma_m^2}), \tag{12}$$

where σ_e and σ_m are the standard deviation of the Gaussian kernels, α is the importance of the motion similarity relative to the embedding similarity and t_i is the frame index. If seeds i and j are from different frames, the motion similarity term is ignored as reflected by the dirac delta term, $\delta(t_i, t_j)$, since their optical flow may not be aligned. Although instance embeddings are trained on images, they are shown to be stable over consecutive frames in [24]. Thus, they are applicable to measure the similarity across frames. Our studies in Sect. 4.4 show that considering this inter-frame similarity is beneficial and ignoring the temporal edges leads to inferior performance.

3.3 Label Propagation

After graph cut, the final step is to propagate the label from seeds to remaining pixels. Given an arbitrary pixel, i, with its temporal-spatial location denoted by $(x_i, y_i, t_i)^T$, we can identify its neighboring seeds on frame t_i by finding its spatially closest seed and the spatial neighbors of that seed in the graph. Besides

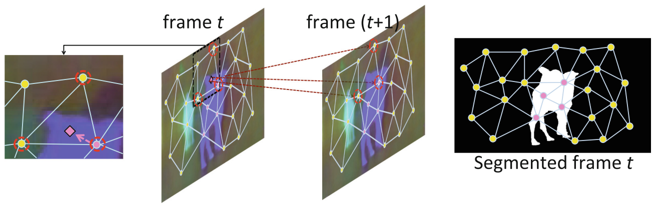

Fig. 5. Given an arbitrary pixel (marked by the diamond), its surrounding nodes (in red circles) are identified from the current frame, the previous frame (omitted here) and the following frame. The label of the node with the shortest embedding distance is assigned to the pixel. Best viewed in color

seeds on frame t_i, we also include the neighboring seeds for the pixels located at $(x_i, y_i, t_i - 1)^T$ and $(x_i, y_i, t_i + 1)^T$, i.e., pixels with the same spatial location in the previous frame and the following frame, as shown in Fig. 5. The neighboring seed set for pixel i is denoted by $\mathcal{N}(i)$. Among the seeds in $\mathcal{N}(i)$, the one with the shortest embedding distance to i is found via

$$n = \arg \min_{m \in \mathcal{N}(i)} ||\mathbf{e}_i - \mathbf{e}_m||_2^2. \tag{13}$$

The label of seed n is assigned to pixel i. We estimate the probability that pixel i is foreground from the shortest embedding distance to the nodes labeled as foreground and background in $\mathcal{N}(i)$, denoted by $d_{FG}(i)$ and $d_{BG}(i)$, respectively. The foreground probability is defined by

$$p^{FG}(i) = \frac{\exp[-d_{FG}^2(i)]}{\exp[-d_{FG}^2(i)] + \exp[-d_{BG}^2(i)]}. \tag{14}$$

Note that if the nodes in $\mathcal{N}(i)$ are all foreground (or background), then $p^{FG}(i)$ is defined to be 1 (or 0). Because the resolution of the dense embedding map is lower than the original video, we upsample the probability map using the multi-linear interpolation to the original resolution and further refine it with a dense conditional random field (CRF) [23].

4 Experiments

4.1 Datasets and Evaluation Metrics

The proposed method is evaluated on the DAVIS 2016 dataset [29] and the Freiburg-Berkeley Motion Segmentation 59 (FBMS-59) dataset [27]. The latter has multiple moving objects labeled separately. By following [31], [24], we convert the annotation to binary masks by grouping individual object masks.

DAVIS 2016. The DAVIS 2016 dataset [29] contains 50 densely annotated video sequences with high resolution. It is partitioned into two splits, *train* and *val*, with 30 and 20 sequences, respectively. Some videos from this dataset are challenging due to motion blur, occlusion and object deformation. The evaluation metrics include region similarity, boundary accuracy and temporal stability, proposed in [29]. The region similarity, denoted by $\mathcal{J}$, is defined as the intersection over union (IoU) between the annotation and the predicted mask. To measure the boundary accuracy, denoted by $\mathcal{F}$, the annotation boundary and the predicted mask boundary are compared and the F-score (the harmonic mean of precision and recall) is computed. The temporal stability, $\mathcal{T}$, measures the deformation needed to transform from one frame to its succeeding frame, and higher deformation means less smooth masks over time. This metric is applied to a subset of sequences in DAVIS 2016 as described in [29].

FBMS-59. 59 video sequences are collected in the FBMS-59 dataset [27]. In contrast to DAVIS, this dataset is sparsely annotated, with masks of 720 frames provided. We test the proposed method on the *test* split, containing 30 sequences. Apart from the aforementioned region IoU ($\mathcal{J}$), we also use the F-score protocol from [27] for this dataset, being consistent with previous methods.

4.2 Implementation

We train the bilateral network on the DAVIS 2016 *train* split. For each frame, the 4-D bilateral space position vector (dx, dy, x, y) of pixels is normalized within each frame and then input to the BNN. Theoretically, the learnable filters can be any 4-D tensor, but practically, to reduce the number of parameters of the network, the BNN is composed of four cascaded 1-D filters, one for each dimension. To train the BNN, we set the batch size to 64 and use a learning rate of 0.0001, with a total of $10k$ steps. Data augmentation is done by random sampling $M = 50k$ pixels in the low objectness region ($p^{\text{Obj}} < 0.001$) for splatting. During inference, we pick the M pixels with the lowest objectness score. The objectness score used for the BNN is from [9] and the optical flow is computed by a re-implementation of FlowNet2.0 [15]. For embedding graph cut, we use the instance embeddings from [9], where the training is conducted on the Pascal dataset [7]. We do not further fine-tune on any video dataset. The hyperparameters are determined by cross validation: the pairwise cost weight, λ is 0.1; the variance for instance embeddings and optical flow in Eq. (12), $\sigma_e^2 = 1$ and $\sigma_m^2 = 10$. The weight of the motion similarity relative to embedding similarity, α, is set to 1.

4.3 Performance Comparison

DAVIS 2016. The results on DAVIS 2016 are displayed in Table 1. The proposed method outperforms other methods under the unsupervised scenario in terms of $\mathcal{J}$ Mean and $\mathcal{F}$ Mean, with an improvement of 1.9% and 3.0% over the second best method, IET [24]. Note that for $\mathcal{J}$ Mean, our method even achieves slightly better results than some recent semi-supervised methods, OSVOS [3] and MSK [21]. In terms of temporal stability, our method is the second best

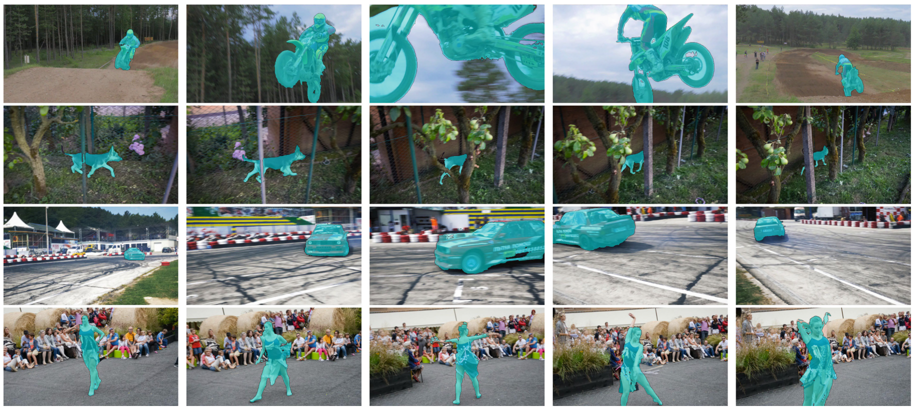

Fig. 6. Qualitative segmentation results from DAVIS 2016 *val* split. The four sequences feature motion blur, occlusion, large object appearance change, and static objects in background, respectively

Table 1. The results on the *val* split of DAVIS 2016 dataset [29]. The proposed method outperforms other unsupervised methods in terms of $\mathcal{J}/\mathcal{F}$ Mean, and is even better than some semi-supervised methods. For the temporal stability ($\mathcal{T}$), our method is the second best. The $\mathcal{J}$ value of each sequence is provided in the supplementary material

	Semi-supervised			Unsupervised						
	OAVOS [36]	OSVOS [3]	MSK [21]	SFL [5]	LVO [32]	MP [31]	FSEG [16]	ARP [22]	IET [24]	Ours
$\mathcal{J}$ Mean ↑	**86.1**	79.8	79.7	67.4	75.9	70.0	70.7	76.2	78.6	**80.4**
$\mathcal{J}$ Recall ↑	**96.1**	93.6	93.1	81.4	89.1	85.0	83.5	91.1	-	**93.2**
$\mathcal{J}$ Decay ↓	**5.2**	14.9	8.9	6.2	**0.0**	1.3	1.5	7.0	-	4.8
$\mathcal{F}$ Mean ↑	**84.9**	80.6	75.4	66.7	72.1	65.9	65.3	70.6	76.1	**78.5**
$\mathcal{F}$ Recall ↑	89.7	**92.6**	87.1	77.1	83.4	79.2	73.8	83.5	-	**88.6**
$\mathcal{F}$ Decay ↓	**5.8**	15.0	9.0	5.1	**1.3**	2.5	1.8	7.9	-	4.4
$\mathcal{T}$ Mean ↓	**19.0**	37.8	21.8	28.2	**26.5**	57.2	32.8	39.3	-	27.8

in the unsupervised category: 1.3% worse than the most stable method, LVO [32]. We provide some visualized results in Fig. 6 and more can be found in the supplementary material.

FBMS-59. The proposed method is evaluated on the *test* split of the FBMS-59 dataset, which has 30 sequences. The results are listed in Table 2. Our method outperforms the second best method, IET [24], in the $\mathcal{J}$ Mean and the F-score by 2% and 0.4%, respectively. We provide visualized segmentation results for the FBMS-59 dataset in the supplementary material.

4.4 Analysis of Module Contributions

Motion-Based BNNs. A video clip can be segmented by directly thresholding the background probability $p_{\text{bnn}}^{\text{BG}}$ in Eq. (5). That is, a pixel is foreground if

$p_{\text{bnn}}^{\text{BG}} < T_{\text{bnn}}$ and we set $T_{\text{bnn}} = 0.5$. This serves as the first baseline and is denoted by "BNN". Since optical flow is error-prone, raw results from the motion-based BNN are not satisfactory, especially when there are unstable stuff regions, e.g., waves. The second baseline is obtained by adaptively thresholding $p_{\text{bnn}}^{\text{BG}}$ by the objectness score p^{Obj}. Namely, a pixel belongs to foreground if $p_{\text{bnn}}^{\text{BG}} < p^{\text{Obj}}$, which effectively eliminates false positives in unstable stuff regions. This baseline is referred as "Obj-BNN". It combines the motion and objectness signals without utilizing the instance embedding or graph cut (also equivalent to assigning label to pixels based on the unary potentials only).

Table 2. Performance comparison on the *test* split of the FBMS-59 dataset [27]

Method	NLC [8]	CUT [20]	FST [28]	CVOS [30]	LVO [32]	MP [31]	ARP [22]	IET [24]	Ours
$\mathcal{J}$ Mean	44.5	-	55.5	-	-	-	59.8	71.9	**73.9**
F-score	-	76.8	69.2	74.9	77.8	77.5	-	82.8	**83.2**

Adding objectness boosts the performance of "BNN" by 20.9%, as shown in Table 3. The motion-based BNN with objectness achieves better results than previous methods using dual-branch CNNs [5,16,32][1], in terms of $\mathcal{J}$ Mean on the *val* split of DAVIS 2016.

The Embedding Graph. The embedding graph can be constructed in multiple ways depending on how the pairwise cost is defined and how graph nodes are linked in Table 4. Without the graph cut, the results match the "Obj-BNN" baseline in Table 3. We present $\mathcal{J}$ Mean results with (77.6) and without (74.7) the CRF refinement. We then constructed the embedding graph without temporal edges. Three options for pairwise cost in Eq. (12) were tested: considering the similarity in embedding space only (row 2), the similarity in motion only (row 3), and both (row 4). We then explored adding different temporal dependencies to the full intra-frame model. We connected seeds in consecutive frames sparsely (row 5): for a seed pair from consecutive frames, we check the seed regions formed by the pixels closest to a seed. If their corresponding seed regions spatially overlap by at least one pixel, they are connected by a temporal edge. We also connected seeds in consecutive frames densely (row 6). The variants of embedding graph cut are evaluated by the $\mathcal{J}$ Mean of the final segmentation with seed labels propagated to all pixels. Best performance is observed with both embedding and motion similarities considered and dense temporal edges.

Online Processing. The capability to process videos online, where results are generated for frames within a fixed latency, is a desirable feature. Using only preceding frames[2] produces the shortest latency and is causal. To process the t-th frame online, the embedding graph is built within a frame window, using

[1] Note that [5] is not as comparable as [16] and [32]. Its motion branch does not take in explicitly computed optical flow but two consecutive frames instead.

[2] We allow accessing frame $(t+1)$ for optical flow computation for frame t.

Table 3. Performance comparison between results of the motion-based BNN and other dual-branch methods on DAVIS 2016 *val* split (*without* CRF refinement)

Method	SFL [5]	LVO [32]	FSEG [16]	BNN	Obj-BNN
$\mathcal{J}$ Mean	67.4	70.1	70.7	53.8	**74.7**
$\mathcal{F}$ Mean	66.7	-	65.9	50.1	**70.9**

Table 4. Performance comparison of different pairwise costs and seed linking schemes. Motion similarity and dense temporal edges help to achieve better performance

	Similarity		Temporal edges		Metrics	
Variant	Embed.	Motion	Sparse	Dense	$\mathcal{J}$ Mean	$\mathcal{J}$ Mean (+CRF)
Obj-BNN					74.7	77.6
Similarity	✓				74.3	78.0
features for		✓			74.8	77.5
pairwise cost	✓	✓			75.7	78.9
Inter-frame	✓	✓	✓		76.2	79.8
seed linking	✓	✓		✓	77.3	80.4

Table 5. Building the embedding graph with different sets of consecutive frames for online and offline processing. Under the online scenario, we consider a temporal window of length $(W+1)$ ending at frame t. For offline processing, a window of length $(2W+1)$ centered at t is used. For label propagation, using seeds from the previous, the current and the following frames gives the optimal results. This group of variants is evaluated on DAVIS 2016 *val* set with $\mathcal{J}$ Mean (without CRF) as the metric. Causal variants (i.e., for online processing) are marked by a star ("*")

	Causal graph window				Acausal graph window		
Frames for label prop.	$W=0$	$W=1$	$W=5$	$W=10$	$W=1$	$W=5$	$W=10$
Current	75.3*	76.8*	76.9*	76.9*	77.0	76.9	76.8
+Previous	75.4*	77.0*	77.0*	77.1*	77.2	77.0	77.0
+Following	75.7	77.3	77.2	77.3	77.3	77.2	77.2

seeds from frames $(t-W)$ to t for the causal case and $(t-W)$ to $(t+W)$ for the acausal case. As shown in Table 5, building the embedding graph with only the current and previous frames does not affect the performance much. Note that $W=0$ eliminates temporal edges and gives the appropriately lower results matching Table 4. We also explore which frames are used for propagating labels from seeds to pixels in Eq. (13): in the top row, only the current frame is used to propagate labels; in the middle row, labels are propagated to pixels from seeds in the current and previous frames; in the bottom row, labels are propagated from the current, previous, and following frames. In the acausal case, we found that $W = 1$ gave the best performance, with seeds from the previous and the following frames included for label propagation.

5 Conclusions

A motion-based bilateral network (BNN) is proposed to reduce the false positives from static but semantically similar objects for the VOS problem in this paper. Based on optical flow and objectness scores, a BNN identifies regions with motion patterns similar to those of non-object regions, which help classify static objects as background. The estimated background obtained by the BNN is further integrated with instance embeddings for multi-frame reasoning. The integration is done by graph cut, and to improve its efficiency, we build a graph consisting of a set of sampled pixels called seeds. Finally, frames are segmented by propagating the label of seeds to remaining pixels. It is shown by experiments that the proposed method achieves the state-of-the-art performance in several benchmarking datasets.

References

1. Adams, A., Baek, J., Davis, M.A.: Fast high-dimensional filtering using the permutohedral lattice. In: Computer Graphics Forum, vol. 29, pp. 753–762. Wiley Online Library (2010)
2. Brox, T., Malik, J.: Large displacement optical flow: descriptor matching in variational motion estimation. IEEE Trans. Pattern Anal. Mach. Intell. **33**(3), 500–513 (2011)
3. Caelles, S., Maninis, K.K., Pont-Tuset, J., Leal-Taixé, L., Cremers, D., Van Gool, L.: One-shot video object segmentation. In: CVPR 2017. IEEE (2017)
4. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., L. Yuille, A.: DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. IEEE Trans. Pattern Anal. Mach. Intell. (2016)
5. Cheng, J., Tsai, Y.H., Wang, S., Yang, M.H.: SegFlow: joint learning for video object segmentation and optical flow. In: The IEEE International Conference on Computer Vision (ICCV) (2017)
6. De Brabandere, B., Neven, D., Van Gool, L.: Semantic instance segmentation with a discriminative loss function. In: IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPRW) (2017)
7. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The PASCAL visual object classes (VOC) challenge. Int. J. Comput. Vis. (IJCV) **88**(2), 303–338 (2010)
8. Faktor, A., Irani, M.: Video segmentation by non-local consensus voting. In: BMVC, vol. 2, p. 8 (2014)
9. Fathi, A., et al.: Semantic instance segmentation via deep metric learning. arXiv preprint arXiv:1703.10277 (2017)
10. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 2980–2988. IEEE (2017)
11. He, Y., Chiu, W.C., Keuper, M., Fritz, M.: STD2P: RGBD semantic segmentation using spatio-temporal data-driven pooling. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

12. Heitz, G., Koller, D.: Learning spatial context: using stuff to find things. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008. LNCS, vol. 5302, pp. 30–43. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88682-2_4
13. Huang, Q., Xia, C., Li, S., Wang, Y., Song, Y., Kuo, C.C.J.: Unsupervised clustering guided semantic segmentation. In: IEEE Winter Conference on Applications of Computer Vision (WACV) (2018)
14. Huang, Q., et al.: Semantic segmentation with reverse attention. In: British Machine Vision Conference (2017)
15. Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T.: FlowNet 2.0: evolution of optical flow estimation with deep networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 2 (2017)
16. Jain, S.D., Xiong, B., Grauman, K.: FusionSeg: learning to combine motion and appearance for fully automatic segmentation of generic objects in videos. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)
17. Jampani, V., Gadde, R., Gehler, P.V.: Video propagation networks. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2 (2017)
18. Jampani, V., Kiefel, M., Gehler, P.V.: Learning sparse high dimensional filters: Image filtering, dense CRFs and bilateral neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4452–4461 (2016)
19. Jang, W.D., Kim, C.S.: Online video object segmentation via convolutional trident network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5849–5858 (2017)
20. Keuper, M., Andres, B., Brox, T.: Motion trajectory segmentation via minimum cost multicuts. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3271–3279 (2015)
21. Khoreva, A., Perazzi, F., Benenson, R., Schiele, B., Sorkine-Hornung, A.: Learning video object segmentation from static images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)
22. Koh, Y.J., Kim, C.S.: Primary object segmentation in videos based on region augmentation and reduction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)
23. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected CRFs with Gaussian edge potentials. In: Advances in Neural Information Processing Systems, pp. 109–117 (2011)
24. Li, S., Seybold, B., Vorobyov, A., Fathi, A., Huang, Q., Kuo, C.C.J.: Instance embedding transfer to unsupervised video object segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2018)
25. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440 (2015)
26. Märki, N., Perazzi, F., Wang, O., Sorkine-Hornung, A.: Bilateral space video segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 743–751 (2016)
27. Ochs, P., Malik, J., Brox, T.: Segmentation of moving objects by long term video analysis. IEEE Trans. Pattern Anal. Mach. Intell. **36**(6), 1187–1200 (2014)
28. Papazoglou, A., Ferrari, V.: Fast object segmentation in unconstrained video. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1777–1784 (2013)

29. Perazzi, F., Pont-Tuset, J., McWilliams, B., Van Gool, L., Gross, M., Sorkine-Hornung, A.: A benchmark dataset and evaluation methodology for video object segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016)
30. Taylor, B., Karasev, V., Soatto, S.: Causal video object segmentation from persistence of occlusions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4268–4276 (2015)
31. Tokmakov, P., Alahari, K., Schmid, C.: Learning motion patterns in videos. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)
32. Tokmakov, P., Alahari, K., Schmid, C.: Learning video object segmentation with visual memory. In: Proceedings of the IEEE International Conference on Computer Vision (2017)
33. Tomasi, C., Manduchi, R.: Bilateral filtering for gray and color images. In: 1998 Sixth International Conference on Computer Vision, pp. 839–846. IEEE (1998)
34. Tsai, Y.H., Yang, M.H., Black, M.J.: Video segmentation via object flow. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3899–3908 (2016)
35. Vijayanarasimhan, S., Ricco, S., Schmid, C., Sukthankar, R., Fragkiadaki, K.: SfM-Net: learning of structure and motion from video. arXiv preprint arXiv:1704.07804 (2017)
36. Voigtlaender, P., Leibe, B.: Online adaptation of convolutional neural networks for video object segmentation. In: British Machine Vision Conference (2017)
37. Yoon, J.S., Rameau, F., Kim, J., Lee, S., Shin, S., Kweon, I.S.: Pixel-level matching for video object segmentation using convolutional neural networks. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 2186–2195. IEEE (2017)
38. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2881–2890 (2017)