Semi-Supervised Video Object Segmentation with Super-Trajectories

Wenguan Wang, Jianbing Shen, *Senior Member, IEEE*, Fatih Porikli, *Fellow, IEEE*, and Ruigang Yang, *Senior Member, IEEE*

Abstract—We introduce a semi-supervised video segmentation approach based on an efficient video representation, called as "super-trajectory". A super-trajectory corresponds to a group of compact point trajectories that exhibit consistent motion patterns, similar appearances, and close spatiotemporal relationships. We generate the compact trajectories using a probabilistic model, which enables handling of occlusions and drifts effectively. To reliably group point trajectories, we adopt a modified version of the density peaks based clustering algorithm that allows capturing rich spatiotemporal relations among trajectories in the clustering process. We incorporate two intuitive mechanisms for segmentation, called as *reverse-tracking* and *object re-occurrence*, for robustness and boosting the performance. Building on the proposed video representation, our segmentation method is discriminative enough to accurately propagate the initial annotations in the first frame onto the remaining frames. Our extensive experimental analyses on three challenging benchmarks demonstrate that, given the annotation in the first frame, our method is capable of extracting the target objects from complex backgrounds, and even reidentifying them after prolonged occlusions, producing high-quality video object segments.

Index Terms—Video segmentation, trajectory extraction, density peaks clustering.

1 Introduction

Semi-supervised video segmentation refers to the partitioning of objects in a given video sequence with available annotations in the first frame. A pixel-accurate, spatiotemporal bipartition of the video is an essential building block for a wide spectrum of applications, such as action recognition [1], object tracking [2], semantic labeling [3], to name a few. Semi-supervised techniques also provide proper initializations for further video editing and analysis tasks (*e.g.*, interactive video cutout, dataset annotation) since they allow a tradeoff between accuracy and human interaction.

Aiming for this task, we incorporate a comprehensive video representation, *super-trajectory*, to capture the underlying spatiotemporal structure information that is intrinsic to real-world scenes. Each super-trajectory is composed of a group of trajectories that are similar in nature and have common characteristics. A point trajectory, *e.g.*, the tracked positions of an individual point across multiple frames, is a constituent of the super-trajectory. This representation portrays several properties of a video:

- *Long-term motion information* is explicitly modeled as it consists of trajectories over extended periods;
- *Spatiotemporal location information* is implicitly interpreted by clustering nearby trajectories; and
- *Compact features*, such as color and motion pattern, are described in a conveniently compact form.

- *W. Wang and J. Shen are with Beijing Laboratory of Intelligent Information Technology, School of Computer Science, Beijing Institute of Technology, (email: wenguanwang@bit.edu.cn, shenjianbing@ucla.edu)*
- *F. Porikli is with the Research School of Engineering, the Australian National University. (email: fatih.porikli@anu.edu.au)*
- *R. Yang is with the University of Kentucky, Lexington, KY 40507. (email: ryang@cs.uky.edu)*

Fig. 1. Our video segmentation method takes the first frame annotation as initialization (left). Leveraging on super-trajectories, the segmentation process achieves superior results even for challenging scenarios including heavy occlusions, complex appearance variations, and large shape deformations (middle, right).

With above convenient qualities, super-trajectory simplifies and reduces the complexity of propagating labels in the segmentation process. We first generate point trajectories based on a probabilistic model, which handles occlusions and drifts naturally. Then, we apply the density peaks based clustering (DPC) algorithm [4] that is modified to attain a proper split of videos in space and time by grouping these trajectories.

Our approach to the design of the super-trajectory is inspired by the following two motivations. Firstly, for the task of video segmentation, it is desirable to have a powerful abstraction of videos that is robust to spatiotemporal structure variations and deformations. As demonstrated in the recently released DAVIS dataset [5], most of the previous heuristic approaches exhibit serious limitations for the cases

with occlusions, motion blur and appearance changes. The proposed super-trajectory, on the other hand, is able to encode video efficiently handling these challenges (see Fig. 1).

Secondly, from the perspective of feature generation, merging and splitting video segments (through the corresponding point trajectories) into atomic spatiotemporal components is essential for handling occlusions and temporal discontinuities. However, it is known that the classical clustering methods, *e.g.*, k-means and spectral clustering, which are widely adopted by the existing trajectory methods, cannot reach a consensus on the definition of a cluster. To address this, we modify the DPC algorithm for grouping the point trajectories, leveraging on its traits of choosing cluster centers based on a more suitable criterion.

We also introduce a *reverse-tracking* strategy by excluding objects that originate outside the frame to eliminate the adverse effects of camera motion. To reidentify objects after occlusions, we exploit *object re-occurrence* information, which reflects the spatiotemporal relations between objects across the entire video sequence.

To summarize, our method has the following contributions:

- A semi-supervised video segmentation algorithm based on super-trajectories that capture spatiotemporal relations among point trajectories (Sections 3 and 4).
- A novel super-trajectory generation method based on a modified version of the DPC algorithm to reliably determine cluster centers that represent spatiotemporal structure variations (Section 3.2).
- A reverse-tracking strategy for identifying objects with long durations of propagation, and an object re-occurrence scheme for recovering objects after occlusion (Sections 4.2 and 4.3).

We evaluate our method on three publicly acceptable datasets, namely DAVIS [5], Youtube-Object [6] and SegTrack-V2 [7] benchmarks and compare with the state-of-the-art both qualitatively and quantitatively. Furthermore, to gain a comprehensive understanding of its various aspects, we implement three variants of our method and conduct multiple ablation studies. We also run two groups of experiments to assess the impacts of its different components. We observe that our method compares favorably with the previous state-of-the-art heuristic methods.

This paper builds upon our conference paper [8] and significantly extends it with in-depth discussions on the algorithm providing more details of the formulation, its implementation, and its multiple variants. It dives deeper into the two important assumptions for video segmentation, *reverse-tracking* and *object re-occurrence*, and quantitatively demonstrates their effectiveness. It also offers a more inclusive and insightful overview of the recent work of video segmentation and trajectory extraction. Last but not least, it reports extensive experimental results with an additional large-scale dataset, Youtube-Object [6] for further validation.

The remainder of the paper is organized as follows. An overview of the related work is presented in Section 2. The proposed super-trajectory is described in detail in Section 3. The novel video segmentation method is explained in Section 4 followed by the experimental analyses on robustness, effectiveness, and efficiency in Section 5.

2 Related Work

We provide a brief overview of recent works in two relevant fields: video segmentation and point trajectory extraction.

2.1 Video Segmentation

According to the level of supervision required, we first broadly categorize the conventional video segmentation techniques into unsupervised, semi-supervised and supervised methods. Then, we specifically discuss the recently proposed deep-learning based approaches, due to their astonishing performance improvement.

Unsupervised algorithms ordinarily aim for over-segmentation, including solutions for hierarchical segmentation [9], [10], temporal superpixel [11], and super-voxels [12], [13]. The key assumption behind these methods is to group pixels that have consistent appearance and motion properties since other types of prior knowledge on image content and object type are absent. Similarly, motion segmentation approaches [14], [15], [16] extract moving object regions from the scene background presuming the motion information is a reliable indicator of the objects. For instance, [14], [15] are specifically based on the analysis of long-term motion information, represented as trajectories, posing the segmentation task as a trajectory clustering problem. More recent works, *e.g.* [17], [18], propose automatic motion segmentation for foreground-background separation. While they do not require manual annotations, they rely on restrictive constraints on the application scenario. For identifying objects, many techniques employ saliency and objectness cues, which are bootstrapped from research efforts in salient object detection and generic object proposals. To this end, [19], [20] introduce saliency information as prior knowledge to infer the object, and [21], [22], [23], [24], [25], [26] generate object segments via ranking hundreds of object candidates [27]. Object proposal based approaches are usually time-consuming due to the high computational load of generating object hypotheses and complicated ranking processes. As stated in [5], unsupervised methods are well-suited for parsing large-scale databases, but they fail in case of their underlying assumptions do not hold.

Semi-supervised methods propagate the given labels in one or more key-frames to the entire video sequence [28], [29], [30], [31], [32]. They are also referred as *label propagation*. As argued earlier, unsupervised approaches are bound by their underlying assumptions, therefore incorporating human provided annotations is considered a reliable solution for object segmentation. Semi-supervised video segmentation methods often rely on optical flow [33], [34] and share similar spirit with *video tracking* [35], [7]. Among many variants, these methods use flow-based random field propagation models [36], patch-seam based propagation strategies [37], energy optimizations over graph models [38], joint segmentation and detection frameworks [39], and pixel segmentation on bilateral spaces [40]. Semi-supervised approaches, compared to their

unsupervised competitors, are more practical and would provide more accurate partitions. At the same time, they may suffer from drift issues during the propagation process.

Supervised methods [41], [42], [43], [44] require tedious user interaction and iterative human corrections. They are often referred as *interactive video segmentation* or *video cut-out* in computer graphics. In general, supervised approaches can attain high-quality boundaries even though they demand extensive and time-consuming human supervision. Compared with unsupervised or semi-supervised methods, supervised approaches are capable of producing more accurate partitions. However, the labor-intensive process is unfeasible at large scale. Thus, supervised methods are more suitable for specific scenarios such as video post-production.

Deep Learning based video segmentation methods have become popular in recent years following the success of deep learning in many computer vision applications. The work of Fragkiadaki *et al.* [24] can be viewed as an early attempt to introduce deep learning into video segmentation. In this work, a CNN based Moving Objectness Detector is trained on image and motion fields for detecting moving objects. Another option to the use deep learning is the shallow combination with the features of the pretrained neural networks, such as [34]. Very recently, efforts have been devoted to explore an integrated use of neural networks in video segmentation. These deep learning based approaches can be classified as either unsupervised [45], [46], [47], [48] or semi-supervised [49], [50], [51], [52], [53], [54], [55] architectures. Thanks to the strong learning capability of neural networks, deep learning based video segmentation methods achieved higher performance over the traditional heuristic methods. However, there are still two remaining difficulties. The first one, mainly due to the absence of sufficiently large and pixel-wise annotated video datasets [52], it is a challenge to train a video segmentation network in an end-to-end fashion. To address this, several deep learning methods attempted to leverage static training samples from the existing large-scale image segmentation datasets (*e.g.*, COCO [56]), and use optical flow as an extra input. The second one comes from the inherent computational requirements as the training step of deep models for video understanding often require large hardware memory and intensive computations. As a result, most of the current deep learning based video segmentation models were built upon pre-trained networks (*e.g.*, VGGnet [57]) and fine-tuned with smaller scale datasets.

In this paper, we introduce a model for semi-supervised video segmentation based on the super-trajectory concept, which is a compact and convenient abstraction of trajectories. The success of the proposed segmentation method demonstrates the potential of super-trajectories for this task.

2.2 Point Trajectory

Point trajectories are generated through tracking points over multiple frames. They have the advantage of representing long-term motion information. Historically, Kanade-Lucas-Tomasi (KLT) [58] is one of the earlier attempts to track a small set of feature points. Inspiring several follow-up studies in video segmentation and action recognition, optical flow based dense trajectories [59] improve over sparse interest point tracking. Some representative studies [14], [60], [61], [62], [15], [63], [64], [65] address the problem of motion segmentation of all moving objects in video, in contrast to traditional unsupervised methods that aim to extract a single primary object. These trajectory segmentation methods usually track points via dense optical flow and perform segmentation via clustering trajectories. The trajectories are directly grouped into a few clusters as object segments, using spectral clustering [14], [60], [63], [64], energy-based clustering [61], or minimum cost multi-cut [65]. The final partitioning is obtained via employing graph-cuts on the rough clustering results, usually assuming the number of objects with different motion patterns is known.

Trajectories are also used for action recognition [66], [67], [68], where the trajectory aligned descriptor acts as a powerful representation for describing actions in video sequences. Wang *et al.* [66] first used trajectory-aligned descriptors for action classification and demonstrated the state-of-the-art performance. Later, an improved version [67] was developed for removing noisy trajectories from the camera motion. With the success of deep learning in computer vision tasks, Wang *et al.* [68] encoded deep-learned features into trajectories, reporting improved performance. Generally, trajectory based action recognition methods [66], [67], [68] put more focus on the combination of trajectory and other descriptors (*e.g.*, HOG) due to the pursuit of describing whole video sequences and capturing most informative parts for classifying the whole action sequence. Thus, the computation of trajectories is relatively straightforward and fast, and the trajectories have some uniform and specified lengths. However, motion segmentation models place more emphasis on the point tracking accuracy and clustering performance as they aim for generating per-frame pixel-wise segmentation. Therefore, the trajectories are usually computed using more accurate but more time-consuming optical flow, inferred via more complicated strategies and with varying lengths.

Existing approaches mostly handle trajectories in pairs or individually, and directly group all trajectories into a few clusters (as segments), ignoring the inner coherence in a group of similar trajectories. Instead, we go one step beyond the conventional trajectory methods by putting trajectories in operation as united super-trajectory groups instead of individual entities. The proposed super-trajectory idea inherits the representability of trajectory for modeling long-term motion information, while further exploiting spatiotemporal relations among trajectories, thus offering compact and atomic video representation.

3 Super-Trajectory

We first introduce the super-trajectory in this section and then describe our segmentation approach in Section 4. In Section 3.1, we present our trajectory generation method based on a probabilistic model. Then, in Section 3.2, we introduce our super-trajectory generation method using density peaks based clustering (DPC) algorithm [4].

3.1 Trajectory Generation

Given a sequence of video frames $I_{1:T} = \{I_1, \cdots, I_T\}$ within time range $[1, T]$, each pixel point can be tracked to the next frame using optical flow. This tracking process can be executed frame-by-frame until some termination conditions (*e.g.*, occlusion, incorrect motion estimates, *etc.*) are reached. The tracked points are composed into a trajectory and a new tracker is initialized where prior tracker finished. Contrast to previous trajectory methods that design many hard thresholds [14], [66], [15] for detecting occlusion, or unreliable motion estimates, we build our trajectory generation on a more interpretable and reasonable probabilistic model.

Let $\mathbf{w}$ denote a flow field indexed by pixel positions that returns a 2D flow vector at a given point. Using LDOF [69], we compute forward-flow field $\mathbf{w}_t$ from frame I_t to I_{t+1}, and the backward-flow field $\hat{\mathbf{w}}_t$ from I_t to I_{t-1}. We track pixel potion $\mathbf{x} = (x, y, t)$ to the consecutive frames in both directions. The tracked points of consecutive frames are concatenated to form a trajectory τ:

$$\tau = \{\mathbf{x}_n\}_{n=1}^{L} = \{(x_n, y_n, t_n)\}_{n=1}^{L}, \qquad t_n \in [1, T], \tag{1}$$

where L indicates the length of trajectory τ and point $\mathbf{x}_n = (x_n, y_n, t_n)$ is tracked via:

$$(x_n, y_n) = (x_{n-1}, y_{n-1}) + \mathbf{w}_{t_{n-1}}(x_{n-1}, y_{n-1}). \tag{2}$$

As the optical flow is subpixel accurate, x and y will usually end up between grid points. We use bilinear interpolation to infer the flow at these points.

We model point tracking process as a first order Markovian process, and denote the probability that n-th point $\mathbf{x}_n$ of trajectory τ is correctly tracked from frame I_{t_1} as $p(\mathbf{x}_n|I_{t_1:t_n})$. The prediction model is defined by:

$$p(\mathbf{x}_n|I_{t_1:t_n}) = p(\mathbf{x}_n|\mathbf{x}_{n-1}, I_{t_n}) p(\mathbf{x}_{n-1}|I_{t_1:t_{n-1}}), \tag{3}$$

where $p(\mathbf{x}_1|I_{t_1}) = 1$ and $p(\mathbf{x}_n|\mathbf{x}_{n-1}, I_{t_n})$ is formulated as:

$$p(\mathbf{x}_n|\mathbf{x}_{n-1}, I_{t_n}) \propto exp\{-(\mathcal{E}_{app} + \mathcal{E}_{occ})\}. \tag{4}$$

The energy functions $\mathcal{E}$ penalize various potential tracking error. The former energy $\mathcal{E}_{app}$ is expressed as:

$$\mathcal{E}_{app}(\mathbf{x}_n, \mathbf{x}_{n-1}) = ||I_{t_n}(x_n, y_n) - I_{t_{n-1}}(x_{n-1}, y_{n-1})||, \tag{5}$$

which penalizes the appearance variations between corresponding points. Obvious, a tracked point would be consistent in appearance over time.

The latter energy $\mathcal{E}_{occ}$ is included to penalize occlusions. It uses the consistency of the forward and backward flows:

$$\mathcal{E}_{occ}(\mathbf{x}_n, \mathbf{x}_{n-1}) = \frac{||\hat{\mathbf{w}}_{t_n}(x_n, y_n) + \mathbf{w}_{t_{n-1}}(x_{n-1}, y_{n-1})||}{||\hat{\mathbf{w}}_{t_n}(x_n, y_n)|| + ||\mathbf{w}_{t_{n-1}}(x_{n-1}, y_{n-1})||}. \tag{6}$$

Ideally, when a point is successfully tracked without occlusion, we expect an one-to-one correlation between corresponding points: $\mathbf{x}_n$ and $\mathbf{x}_{n-1}$. Thus the backward flow vector $\hat{\mathbf{w}}_{t_n}(x_n, y_n)$ should be opposite in direction of the forward flow vector $\mathbf{w}_{t_n}(x_{n-1}, y_{n-1})$: $\hat{\mathbf{w}}_{t_n}(x_n, y_n) = -\mathbf{w}_{t_n}(x_{n-1}, y_{n-1})$, which makes the numerator close to 0. When this consistency constraint is violated, occlusions or unreliable optical flow estimates might occur.

It is important to notice that the proposed tracking model performs accurately yet our model is not limited to the above constraints. We terminate the tracking process when $p(\mathbf{x}_n|I_{t_1:t_n}) < 0.5$, and then we start a new tracker at $\mathbf{x}_n$. In our implementation, we discard the trajectories shorter than four frames.

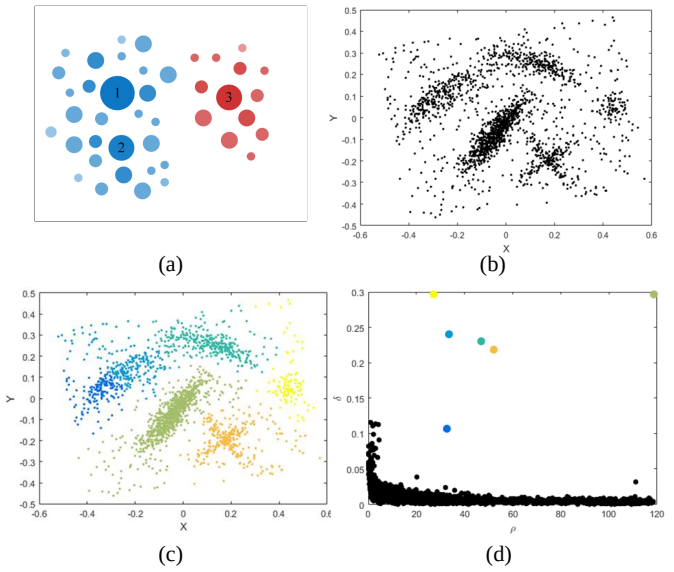


Fig. 2. Illustration of the density peaks based clustering (DPC) algorithm. (a) A schematic diagram where the bigger circles indicate higher local densities ρ. (b) Sample point distributions in two dimensions. (c) Clustering results with DPC, where different colors represent different clusters. (d) Local density ρ and distance δ distributions for the data points of (b). See Section 3.2.1 for detailed explanations.

3.2 Super-Trajectory Generation

Previous studies indicate the value of trajectory based representations for long-term motion information. Here, our intuition is that neighboring trajectories exhibit compact spatiotemporal relationships and they have similar characteristics in appearance and motion patterns. This motives us operating on trajectories as united groups.

We generate super-trajectory by clustering trajectories adopting the recently proposed density peaks based clustering (DPC) [4]. Before introducing our super-trajectory generation method, we first describe DPC.

3.2.1 *Density Peaks based Clustering (DPC)*

DPC clusters the data by finding its density peaks. It provides a unique solution of fast clustering based on the idea that cluster centers are characterized by a higher density than their neighbors and by a relatively large distance from points with higher densities.

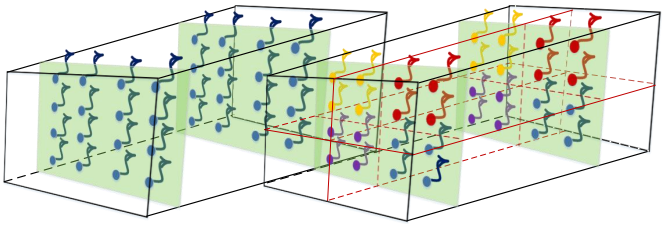
Given the distances d_{ij} between data points, for each data point i, DPC calculates two quantities: local density ρ_i and its distance δ_i from points of higher density. The local density ρ_i of data point i is defined as [1]:

$$\rho_i = \sum\nolimits_j d_{ij}. \tag{7}$$

Here, δ_i is measured by computing the minimum distance between the point i and any other point with higher density:

$$\delta_i = \min_{j:\rho_j > \rho_i} (d_{ij}). \tag{8}$$

1. We do not use a cut-off kernel or Gaussian kernel adopted in [4] due to the small amount of data.



(a) (b)

Fig. 3. Illustration of initial super-trajectory generation process, described in Section 3.2.2. (a) The arrows indicate trajectories and the dots indicate the initial location of the trajectories. (b) We cluster trajectories into K groups on the spatial grid (in this case, $K = 4$).

For the point with highest density, it takes $\delta_i = max_j(d_{ij})$. Since δ_i is much larger than the typical nearest neighbor distance only for points that are local or global maxima in the density, cluster centers are recognized as points for which the value of δ_i is anomalously large.

Cluster centers are the points with high local density ($\rho\uparrow$) and large distance ($\delta\uparrow$) from other points with higher local density. This core observation is illustrated by the simple example in Fig. 2 (a), where the bigger circle indicates higher local density ρ and two clusters, centered as $point1$ and $point3$, are colored blue and red, respectively. We can find that cluster centers are surrounded by neighbors with lower local density and they are at a relatively large distance from any points with a higher local density. $point2$ also have a relatively large local density. However, since it is closer to $point1$ of higher density, it has less distance δ and $point3$ is more favored to be a cluster center due to both high local density and large distance. The data points can be ranked via:

$$\gamma_i = \rho_i \delta_i, \tag{9}$$

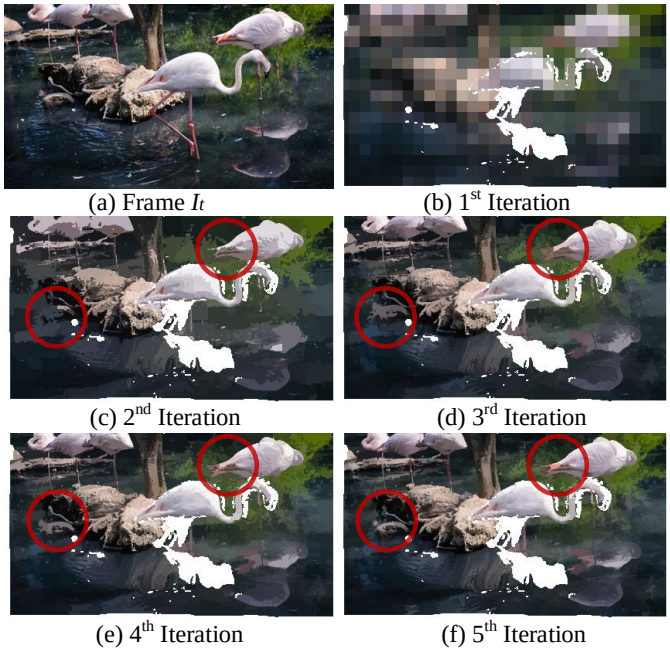
where the cluster centers are recognized as points for which the values of ρ_i and δ_i are both large. Then the top ranking points are selected as centers. After successfully declaring cluster centers, each remaining data points is assigned to the cluster center as its nearest neighbor of higher density. This cluster assignment is performed in a single step, in contrast with other clustering algorithms (*e.g.,* k-means) which iteratively update cluster centers with certain objective function. Fig. 2 (b)-(d) gives another case for DPC with two-dimensional data. As seen, in Fig. 2 (c), the cluster centers correspond to the points (represented as large solid circles) in Fig. 2 (d) with large values of δ and sizeable densities ρ.

3.2.2 Grouping Trajectories via DPC

Given a trajectory $\tau : \{(x_n, y_n, t_n)\}_n$ spans L frames, we define three features: spatial location (l_τ), color (c_τ), and velocity (v_τ), for describing τ:

$$\begin{aligned} l_\tau &= \frac{1}{L}\sum\nolimits_{n=1}^{L}(x_n, y_n), \quad c_\tau = \frac{1}{L}\sum\nolimits_{n=1}^{L} I_{t_n}(x_n, y_n), \\ v_\tau &= \frac{1}{L}\sum\nolimits_{n=1}^{L}\big(\frac{1}{\Delta t}(x_{n+\Delta t} - x_n, y_{n+\Delta t} - y_n)\big), \end{aligned} \tag{10}$$

where we set $\Delta t = 3$. We tested $\Delta t = \{5, 7, 9\}$ and did not observe obvious effect on the results.



(a) Frame I_t (b) 1st Iteration (c) 2nd Iteration (d) 3rd Iteration (e) 4th Iteration (f) 5th Iteration

Fig. 4. Super-trajectory generation via iterative trajectory clustering. (a) Frame I_t. (b)-(f) Visualization of super-trajectory in the time slice I_t at different iterations. Each pixel is assigned to the average color of all points within its super-trajectory. The blank areas are the discarded trajectories, which are shorter than four frames. The areas with obvious changes are highlighted in red circles. For clarity, we set the number of initial spatial grid to K=500. See Section 3.2.2 for more details.

Between each pair of trajectories τ_i and τ_j that share some frames, we define their distance d_{ij} via measuring descriptor similarity:

$$d_{ij} = \sum\nolimits_{f\in\{l,c,v\}} ||f_{\tau_i} - f_{\tau_j}||. \tag{11}$$

We normalize color distance on max intensity, location distance on sampling step R (detailed below), motion distance on the mean motion magnitude of all the trajectories, which makes above distance measures to have similar scales. In case there is no temporal overlap, we set $d_{ij} = H$, where H has a very large value.

It is clear that trajectories are usually asynchronous, *i.e.,* they cover different frames, mainly due to occlusion and disocclusion. For this, we modify the DPC algorithm for clustering the compact trajectories into super-trajectories. We first roughly partition trajectories into several non-overlap clusters, and then iteratively updates each partition to get the optimized trajectory clusters. The only parameter of our super-trajectory algorithm is number of spatial grids K, as the degree of spatial subdivision. The spatial sampling step becomes $R = \sqrt{S/K}$, where S refers to the product of the height and width of image frame. The clustering procedure begins with an initialization step where we divide the input video $I_{1:T}$ into several non-overlap spatiotemporal volumes of size $R \times R \times T$. As shown in Fig. 3, all the trajectories $\mathcal{T} = \{\tau_i\}_i$ are divided into K spatially regular volumes. A trajectory τ falls into the volume where it starts. Then we need to find a proper cluster number of each trajectory group, thereby further offering a reasonable temporal split of video, and further generate more accurate spatiotemporal clustering.

Algorithm 1 DPC for Generating Super-Trajectory Centers

Input: A sub-group of trajectories $\mathcal{T}' = \{\tau'_i\}_i$ ($\mathcal{T}' \subset \mathcal{T}$), distance matrix $\{d_{ij}\}$ via Eq. 11 and cluster number C;
Output: Organized trajectory clusters;
1: Compute local densities $\{\rho_i\}_i$ via Eq. 7;
2: Compute distance $\{\delta_i\}_i$ via Eq. 8;
3: Find $\{\tau'_{i'}\}_{i'}$ with $\delta_{i'} = H$, where $|\{\tau'_{i'}\}_{i'}| = n'$;
4: **if** $C < n'$ **then**
5: Select $\{\tau'_{i'}\}_{i'}$ as cluster centers;
6: **else**
7: Compute $\{\gamma_i\}_i$ via $\gamma_i = \rho_i \delta_i$;
8: Select the trajectories with C highest γ values as cluster centers;
9: **end if**
10: Assign each remaining trajectories to cluster center as its nearest neighbor of higher density ρ.

For each trajectory group, we initially estimate the cluster number as $C = T/\overline{L}$, where $\overline{L}$ indicates the average length of all trajectories. Then we apply a modified DPC algorithm for generating trajectory clusters, as described in Alg. 1. In Alg. 1-3, in case $\delta_i = H$ the trajectory τ_i does not have any temporal overlap with those trajectories that have higher local densities. This means the trajectory τ_i is the center of an isolated group. If $C < n'$ in Alg. 1-4, there exist more than C unconnected trajectory groups. Accordingly, we select the trajectories with the highest densities of those unconnected trajectory groups as the centers (Alg. 1-5). Otherwise, as in Alg. 1-7 and 8, the trajectories with the C highest γ values are selected as the cluster centers. The whole initialization process is described in Alg. 2-1,2,3.

With above initialization process, we group trajectories into super-trajectories according to their spatiotemporal relationships and similarities (see Fig. 4(b)). Next, we iteratively refine our super-trajectory assignments. In this process, each trajectory is classified into the nearest cluster center.

For reducing the searching space, we only search the trajectories fall into a $2R \times 2R \times T$ space-time volume around the cluster center $\tau_{i'}$ (Alg. 2-7). This results in a significant speed advantage by limiting the size of search space to reduce the number of distance calculations. Once each trajectory has been associated to the nearest cluster center, an update step adjusts the center of each trajectory cluster via Alg. 1 with $C = 1$ (Alg. 2-14,15). We drop very small trajectory clusters and combine those trajectories to other nearest trajectory clusters. In practice, we find $4 \sim 5$ iterations for above refining process are enough for obtaining satisfactory performance. Visualization results of super-trajectory generation with different iterations are presented in Fig. 4.

Using DPC in Alg. 1, we group all trajectories $\mathcal{T} = \{\tau_i\}_i$ into m non-overlapping clusters, represented as super-trajectories $\mathcal{X} = \{\chi_j\}_{j=1}^m$, where $\chi_j = \{\tau_i \mid \tau_i$ is classified into j-th cluster via Alg. 2$\}$. Note that, m (the number of super-trajectories) is varied at each iteration in Alg. 2 since we merge small clusters into other clusters. Additionally, m for different videos is different even with same input parameter K. That is important, since different videos have different temporal characteristics, thus we only constrain their spatial shape via K.

Algorithm 2 Super-Trajectory Generation

Input: All the trajectories $\{\tau_i\}_i$, spatial sampling step R;
Output: Super-trajectory assignments;
/ Initialization */*
1: Obtain K trajectory groups via spatial sampling step R;
2: Set initial cluster number $C = T/\overline{L}$ for each group;
3: Obtain initial cluster centers $\{\tau_{i'}\}_{i'}$ from each trajectory group via Alg. 1, where $|\{\tau_{i'}\}_{i'}| = m$;
4: **loop**
/ Iterative Assignment */*
5: Set label $l_i = -1$ and distance $\kappa_i = H$ for each trajectory τ_i;
6: **for** each trajectory cluster center $\tau_{i'}$ **do**
7: **for** each trajectory τ_j falls in a $2R \times 2R \times T$ space-time volume around $\tau_{i'}$ **do**
8: Compute distance $d_{ji'}$ between τ_j and $\tau_{i'}$ via Eq. 11;
9: **if** $d_{ji'} < \kappa_j$ **then**
10: Set $\kappa_j = d_{ji'}$, $l_j = i'$;
11: **end if**
12: **end for**
13: **end for**
/ Update Assignment */*
14: Set cluster number $C = 1$ for each group;
15: Update $\{\tau_{i'}\}_{i'}$ for each cluster via Alg. 1.
16: **end loop**

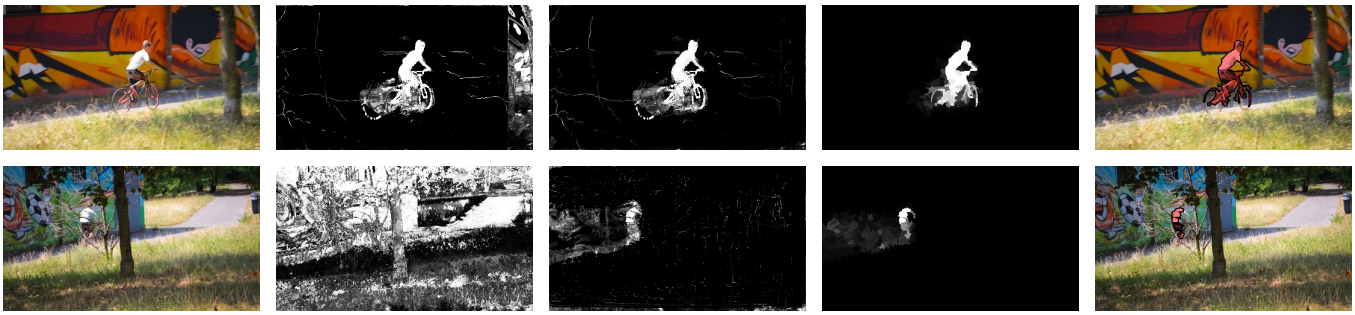
4 Super-Trajectory for Segmentation

In Section 3, we cluster a set of compact trajectories into super-trajectory. In this section, we describe our video segmentation approach that leverages on super-trajectories.

4.1 Super-Trajectory based Propagation

Given the mask $\mathcal{M}$ of the first frame, we seek a binary partitioning of pixels into foreground and background classes. Clearly, the annotation can be propagated to the rest of the video, using the trajectories that start at the first frame. However, only a few of points can be successfully tracked across the whole scene, due to occlusion, drift or unreliable motion estimation. Benefiting from our efficient trajectory clustering approach, super-trajectories are able to spread more annotation information over longer periods. This inspires us to base our label propagation process on super-trajectory.

For inferring the foreground probability of super-trajectories $\mathcal{X}$, we first divide all the trajectories $\mathcal{T}$ into three categories: foreground trajectories $\mathcal{T}^f$, background trajectories $\mathcal{T}^b$ and unlabeled trajectories $\mathcal{T}^u$, where $\mathcal{T} = \mathcal{T}^f \cup \mathcal{T}^b \cup \mathcal{T}^u$. The $\mathcal{T}^f$ and $\mathcal{T}^b$ are the trajectories which start at the first frame and are labeled by the annotation mask $\mathcal{M}$, while the $\mathcal{T}^u$ are the trajectories start at any frames except the first frame, thus cannot be labeled via $\mathcal{M}$. Accordingly, super-trajectories $\mathcal{X}$ are classified into two categories: labeled ones $\mathcal{X}^l$ and unlabeled ones $\mathcal{X}^u$. A labeled super-trajectory $\chi_j^l \in \mathcal{X}^l$ contains at least one labeled



(a) (b) (c) (d) (e)

Fig. 5. (a) Input frames. (b) Estimated foregrounds via Eq. 12 and the appearance model in Section 4.1. (c) Estimated foregrounds via our reverse tracking strategy (Eq. 14) and the updated appearance model in Section 4.2. (d) Estimated foregrounds via backward re-occurrence based optimization (Eq. 16, Section 4.3). (e) Final segmentation results.

trajectory from $\mathcal{T}^f$ or $\mathcal{T}^b$, and its foreground probability can be computed as the ratio between the included foreground trajectories and the labeled ones it contains:

$$p_f(\chi_j^l) = \frac{|\chi_j^l \cap \mathcal{T}^f|}{|\chi_j^l \cap \mathcal{T}^f| + |\chi_j^l \cap \mathcal{T}^b|}. \tag{12}$$

For the points belonging to the labeled super-trajectory χ_j^l, their foreground probabilities are set as $p_f(\chi_j^l)$.

Then we build an appearance model for estimating the foreground probabilities of unlabeled pixels. The appearance model is built upon the labeled super-trajectories $\mathcal{X}^l$, consists of two weighted Gaussian Mixture Models over RGB color values, one for the foreground and one for the background. The foreground GMM is estimated form all labeled super-trajectories $\mathcal{X}^l$, weighted by their foreground probabilities $\{p_f(\chi_j^l)\}_j$. The estimation of background GMM is analogous, with the weight replaced by the background probabilities $\{1-p_f(\chi_j^l)\}_j$. The foreground GMM is initialized with 3 components, while the background GMM has 5 components, following general settings. The appearance models leverage the foreground and background super-trajectories over many frames, instead of using only the first frame or labeled trajectories, therefore they can robustly estimate appearance information.

4.2 Reverse Tracking

Although above model successfully propagates more annotation information across the whole video sequence, it still suffers from some difficulties: the model will be confused when a new object comes into view (see Fig. 5 (b)). To this, we propose to *reversely track points* for excluding new incoming objects. We compute the 'source' of unlabeled trajectory $\tau_i^u \in \mathcal{T}^u$:

$$(x_0, y_0) = (x_1, y_1) - v_{\tau_i^u}, \tag{13}$$

where (x_1, y_1) indicates starting position and $v_{\tau_i^u}$ refers to velocity via Eq. 10. It is clear that, if the virtual position (x_0, y_0) is out of image frame domain, trajectory τ_i^u is a latecomer. For those trajectories $\mathcal{T}^o \subset \mathcal{T}^u$ start outside view, we treat them as background. Labeled super-trajectory $\chi_j^l \in \mathcal{X}^l$ is redefined as the one contains at least one trajectory from $\mathcal{T}^f$, $\mathcal{T}^b$ or $\mathcal{T}^o$, and Eq. 12 is updated as

$$p_f(\chi_j^l) = \frac{|\chi_j^l \cap \mathcal{T}^f|}{|\chi_j^l \cap \mathcal{T}^f| + |\chi_j^l \cap \mathcal{T}^b| + |\chi_j^l \cap \mathcal{T}^o|}. \tag{14}$$

Those outside trajectories $\mathcal{T}^o$ are also adopted for training appearance model in prior step. According to our experiment in Section 5.3, this assumption offers about 6% performance improvement. Foreground estimation results via our reverse tracking strategy are presented in Fig. 5 (c).

4.3 Backward Re-occurrence

Most video segmentation methods assume objects are consistent between successive frames. However, it is also very common that objects move into or leave view, which poses great challenge for existing approaches. Instead of constraining local object consistency, we pursue global consistency via exploring *re-occurrence* of objects across the whole scene. As suggested by [19], objects, or regions, often re-occur both in space and in time. Here, we build correspondences among re-occurring regions across distant frames and transport foreground estimates globally. This process is based on super-pixel level, since super-trajectories cannot cover all of pixels.

Let $\{r_i\}_i$ be the superpixel set of input video. For each region, we search its N Nearest Neighbors (NNs) as its re-occurring regions using KD-tree search. For region r_i of frame I_t, we only search its NNs in previous frames $\{I_1, \cdots, I_t\}$. Such *backward search* strategy is for biasing the segmentation results of prior frames as the propagation accuracy degrades over time. Following [19], each region r_i is represented as a concatenation of several descriptors f_{r_i}: RGB and LAB color histograms (6 channels$\times$20 bins), HOG descriptor (9 cells$\times$6 orientation bins) computed over a 15×15 patch around superpixel center, and spatial coordinate of superpixel center. The spatial coordinate is with respect to image center and normalized into $[0, 1]$, which implicitly incorporates spatial consistency in NN-search.

After NN-search in the feature space, we construct a weight matrix W for all the regions $\{r_i\}_i$:

$$W_{ij} = \begin{cases} e^{-||f_{r_i} - f_{r_j}||} & \text{if } r_j \text{ is one of NNs of } r_i \\ 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

Then a probability transition matrix P is built via row-wise normalization of W. We define a column vector v that gathers all the foreground probabilities of $\{r_i\}_i$. The foreground probability of a superpixel is assigned as the average foreground probabilities of its pixels.

Then we iteratively update v via the probability transition matrix P. In each iteration k, we update $v^{(k)}$ via:

$$v^{(k)} = Pv^{(k-1)}, \tag{16}$$

which equivalents to updating foreground probability of a region with the weighted average of its NNs. In each iteration, we keep the foreground probabilities of those points belonging to labeled trajectories unchanged. Then we recompute $v^{(k)}$ and update it in next iteration. In this way, the relatively accurate annotation information of the labeled trajectories is preserved. Additionally, the annotation information is progressively propagated in a forward way and the super-trajectories based foreground estimates are consistent even across many distant frames (see Fig. 5 (d)).

After 10 iterations, the pixels (regions) with foreground probabilities lager than 0.5 are classified as foreground, thus obtaining final binary segments. In Section 5.3, we test $N = \{4, 6, \cdots, 20\}$ and only observe $\pm 0.3\%$ performance variation. We set $N = 8$ for obtaining best performance.

5 Experimental Results

The performance evaluation and analysis of the proposed approach are reported in this section. Two groups of experiments are conducted. First, our approach is compared with some of the state-of-the-art video segmentation approaches on three universally acceptable benchmarks. Second, a few important issues regarding our approach are discussed, such as dissecting various components and variants, and the running time.

5.1 Experimental Setup

Parameter Settings In Section 3.2, we set number of spatial grids $K = 1200$. In Section 4.3, we over-segment each frame into about 2000 superpixels via SLIC [70] for good boundary adherence. For each superpixel, we set the number of NNs $N = 8$. In our experiments, all the parameters of our algorithm are fixed to unity.

Datasets We evaluate our method on three public video segmentation benchmarks, *e.g.*, DAVIS [5], YouTube-Objects [6], and Segtrack-V2 [7]. Some statistics and features of these datasets are summarized in Table 1.

The newly released DAVIS [5] contains 50 video sequences (3455 frames in total) and pixel-accurate manual ground-truth for the foreground object in every frame. These videos span a wide range of typical challenges encountered in video object segmentation such as occlusions, fast-motion, appearance changes and motion blur. The videos in DAVIS are split into train set (30) and validation set (20).

Youtube-Objects [6] is a large dataset of 1, 407 videos collected from 155 web videos. This dataset includes videos with 10 object categories. With the setting and ground-truth masks of [13], we consider totally 126 videos with more than 20, 000 frames. The annotations are roughly pixel-level and provided on every 10^{th} frame of downsampled videos.

TABLE 1
Characteristics of three video segmentation datasets used in evaluation.

Dataset	Ref	Year	#Videos	#Frames	#Objects
DAVIS	[5]	2016	50 *(train:30,val:20)*	3, 455	50
YouTube-Objects	[6]	2012	126	20, 000	126
Segtrack-V2	[7]	2013	14	947	24

SegTrack-V2 [7] extends the SegTrack dataset [29] to contain 8 additional videos, in which totally 14 low-resolution video sequences with 24 instance objects and 947 frames. SegTrack-V2 is a relatively small but is the most widely adopted dataset for video segmentation. It is originally proposed for joint segmentation and tracking and is designed to be challenging with respect to background-foreground color similarity, fast motion and complex shape deformation. Pixel-level annotation on the objects is offered for every frame. Since instance-level masks are provided for sequences with multiple objects, in our experiments, we treat each specific instance segmentation as separate problem.

5.2 Performance Comparison

To evaluate the quality of the proposed Super-Trajectory based Video segmentation (STV), we provide in this section both qualitative as well as quantitative comparison on DAVIS dataset [5], Youtube-Object [6] and SegTrack-V2 [7] datasets. We compare the proposed STV against nine classical state-of-the-art alternatives: BVS [40], FCP [38], JMP [44], SEA [37], TSP [11], HVS [9], VSF [33], SCF [13], and OFL [34]. BVS, FCP, JMP, SEA, VSF, SCF, and OFL are *semi-supervised* video segmentation approaches. TSP and HVS are for unsupervised over-segmentation, but they can also accept initial annotation in the first frame, following the settings in [5]. For completeness, we also report six deep learning based semi-supervised video segmentations models: VPN [49], CTN [52], SFL [53], MSK [50], OSVOS [51], and OnAVOS [54]. The results are obtained via running their publicly available codes with default settings or borrowed from their papers.

5.2.1 Evaluation on DAVIS Dataset

Evaluation Metrics We evaluate the effectiveness of our approach on DAVIS dataset with three accompanied evaluation tools: intersection-over-union metric ($\mathcal{J}$) for measuring the region-based segmentation similarity, F-measure ($\mathcal{F}$) for measuring the contour accuracy, temporal stability ($\mathcal{T}$) for measuring the temporal consistency of segments.

Intersection-over-union metric is one of the most widely adopted metric to evaluate the performance of image/video segmentation methods. Given a segmentation mask M and ground-truth G, intersection-over-union score is defined as

$$\mathcal{J} = \frac{M \bigcap G}{M \bigcup G}.$$

TABLE 2
IoU score ($\mathcal{J}$), contour accuracy ($\mathcal{F}$) and temporal stability ($\mathcal{T}$) averaged on the validation set of DAVIS [5]. For IoU score and contour accuracy, higher values are better. For temporal stability, lower values are better. The best results of non-deep learning and deep learning models are **boldfaced**, respetively.

Dataset	Metric	Non-Deep Learning Model							Deep Learning Model					
		BVS	FCP	JMP	SEA	TSP	HVS	**STV**	VPN	CTN	SFL	MSK	OSVOS	OnAVOS
DAVIS	IoU Score $\mathcal{J}\uparrow$	0.600	0.584	0.570	0.504	0.319	0.546	**0.689**	0.702	0.735	0.761	0.797	0.798	**0.861**
	Contour Accuracy $\mathcal{F}\uparrow$	0.588	0.492	0.531	0.480	0.297	0.529	**0.670**	0.655	0.693	0.760	0.754	0.806	**0.849**
	Temporal stability $\mathcal{T}\downarrow$	0.347	0.306	0.159	**0.154**	0.561	0.360	0.185	0.324	0.220	0.189	0.218	0.378	**0.190**

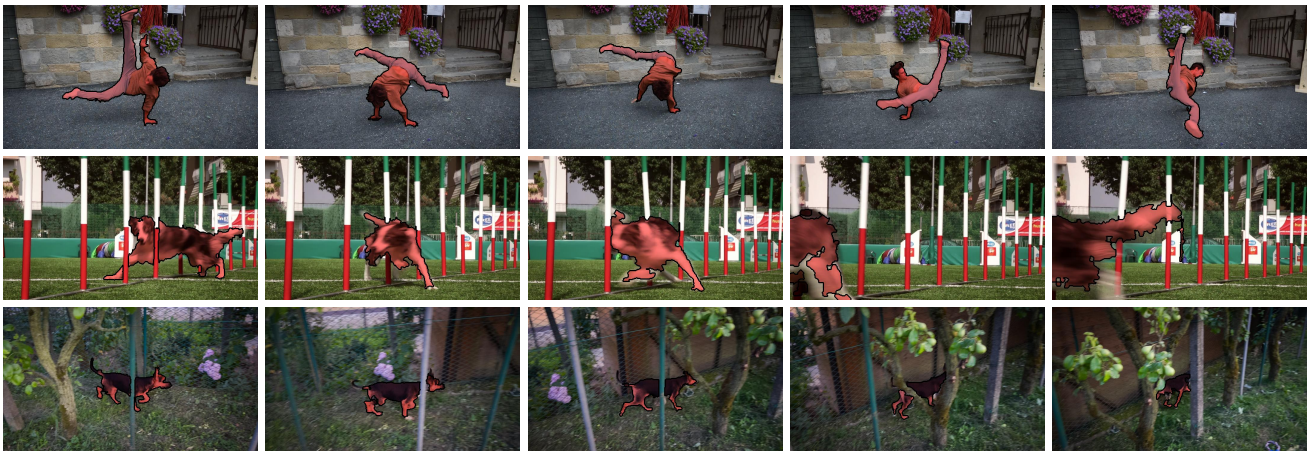


Fig. 6. Qualitative segmentation results on four video sequences from DAVIS [5] (from top to bottom: *bmx-bumps*, *breakdance-flare*, *dog-agility* and *libby*). It can be observed that the proposed algorithm is applicable to a large set of scenarios and robust to scale changes, motion blur, occlusions and multiple splits, and background appearance similarities.

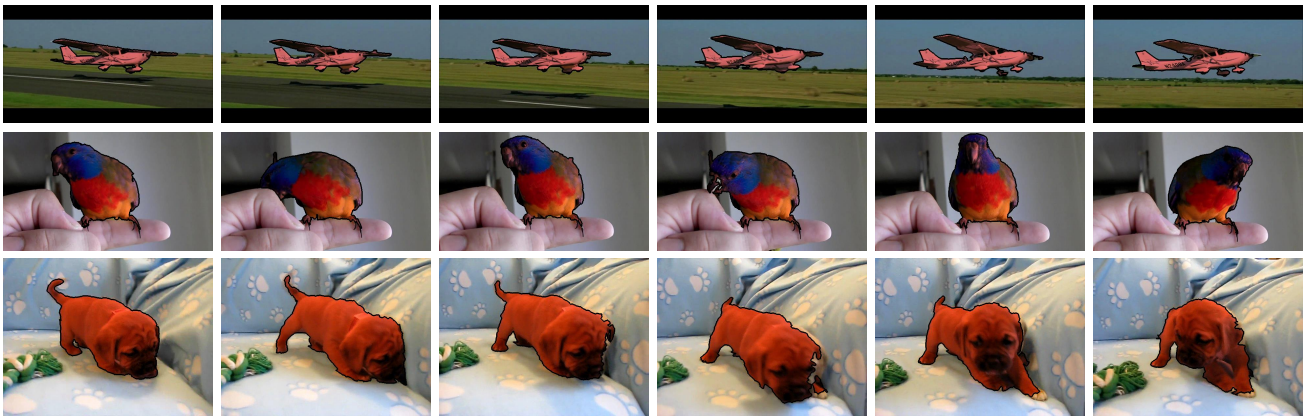


Fig. 7. Qualitative segmentation results on representative video sequences from Youtube-Objects dataset [6] (from top to bottom: *aero02*, *bird12*, and *dog10*). The initial masks are presented in the first row.

Contour accuracy ($\mathcal{F}$) is for measuring how well the segment contours $c(M)$ match the ground-truth contour $c(G)$. Contour-based precision P_c and recall R_c between $c(M)$ and $c(G)$ can be computed via bipartite graph matching. Given P_c and R_c , contour accuracy $\mathcal{F}$ is defined as

$$\mathcal{F} = \frac{2P_cR_c}{P_c + R_c}.$$

Temporal stability ($\mathcal{T}$) is used for penalizing inconsistent segments. It is computed as the per-pixel cost of matching two successive segmentation contours. The match is achieved by minimizing the Shape Context Descriptor (SCD) [71] distances between the matched points.

Quantitative Results In Table 2, IoU score, contour accuracy, and temporal stability (averaged on the validation set of the DAVIS dataset) are reported. It can be observed that, among the heuristic video segmentation models, our method (STV) achieves the highest average IoU score (**0.689**) over the 20 video sequences from the validation set of the DAVIS dataset. Our results also achieve a significant improvement over the second best algorithm BVS (0.600) and the third best algorithm FCP (0.584). In addition, STV achieves the best overall contour accuracy (**0.670**) over other non-deep learning based algorithms and gains a competitive temporal stability score (0.185). This demonstrates that our segments align better with the ground-truth object boundaries and

TABLE 3
IoU score ($\mathcal{J}$) on the Youtube-Objects dataset [6]. The average is computed over all 126 video sequences. Higher values are better. The best results are **boldfaced**.

Dataset	Category	Method				
		BVS	OFL	VSF	SCF	**STV**
Yotube -Object	aeroplane	0.808	0.853	0.890	0.862	0.811
	bird	0.764	0.831	0.816	0.810	0.813
	boat	0.601	0.706	0.742	0.685	0.791
	car	0.567	0.688	0.709	0.693	0.754
	cat	0.527	0.606	0.677	0.588	0.780
	cow	0.648	0.715	0.791	0.685	0.722
	dog	0.616	0.716	0.703	0.617	0.739
	horse	0.531	0.623	0.678	0.539	0.716
	motorbike	0.416	0.599	0.615	0.608	0.680
	train	0.621	0.747	0.782	0.663	0.761
	Avg.	0.597	0.701	0.740	0.649	**0.756**

they are temporally consistent. We also observe that, compared with the heuristic methods, deep learning based semi-supervised video segmentation algorithms generate more accurate segmentation results.

Qualitative Results Qualitative video segmentation results for four video sequences from the DAVIS dataset [5] are presented in Fig. 6. With the first frame as initialization, the proposed algorithm has the ability to segment the objects with fast motion patterns (*breakdance-flare*) or large shape deformation (*dog-agility*). It also produces accurate segmentation maps even when the foreground suffers occlusions (*libby*). In Section 5.3.3, we provide a more detailed analysis of the performance of our method for typical video object segmentation challenges.

5.2.2 Evaluation on Youtube-Object and SegTrack-V2

Quantitative Results We report our performance on the Youtube-Object [6] and SegTrack-V2 [7] datasets, which are widely used for semi-supervised segmentation. The IoU scores of our and various state-of-the-art methods on Youtube-Object and SegTrack-V2 are presented in Table 3 and Table 4, respectively. As can be seen, our method outperforms other methods overall, achieving the best IoU score on most of the videos with the average score up to **0.753** (Youtube-Object) and **0.781** (SegTrack-V2).

Qualitative Results Representative pixel labeling results on Youtube-Object and SegTrack-V2 datasets are shown in Fig. 7 and Fig. 8. We can observe that the target foreground objects in challenging scenarios (such as existence of similar objects, deformations, color changes, motion and image blur, scale variations, etc.) can be accurately segmented out by our algorithm. Our quantitative and qualitative results demonstrate the power of our method.

5.3 In-Depth Validation Experiments

In this section, we offer more detailed exploration for the proposed approach in several aspects with DAVIS dataset [5]. We test the values of important parameters, verify basic assumptions of the proposed algorithm, evaluate the contributions from each part of our approach, perform attribute-based study and conduct runtime comparison.

TABLE 4
IoU score ($\mathcal{J}$) on the SegTrack-V2 dataset [7]. The average is computed over all 24 object instances. Higher values are better. The best results are **boldfaced**.

Dataset	Object	Method				
		BVS	OFL	SEA	HVS	**STV**
SegTrack -V2	bird of paradise	0.897	0.871	0.823	0.868	0.901
	birdfall	0.653	0.529	0.093	0.574	0.461
	bmx1	0.671	0.879	0.445	0.392	0.922
	bmx2	0.032	0.040	0.000	0.325	0.401
	cheetah1	0.054	0.259	0.177	0.188	0.666
	cheetah2	0.092	0.372	0.006	0.244	0.467
	drift1	0.685	0.779	0.429	0.552	0.934
	drift2	0.327	0.274	0.111	0.272	0.509
	frog	0.761	0.784	0.634	0.671	0.812
	girl	0.865	0.842	0.624	0.319	0.916
	hummingbird1	0.532	0.672	0.140	0.137	0.762
	hummingbird2	0.287	0.685	0.368	0.252	0.675
	monkey	0.875	0.878	0.761	0.619	0.925
	monkeydog1	0.405	0.471	0.049	0.683	0.432
	monkeydog2	0.171	0.210	0.090	0.188	0.874
	parachute	0.937	0.933	0.925	0.691	0.942
	penguin1	0.816	0.804	0.802	0.720	0.970
	penguin2	0.820	0.835	0.731	0.807	0.928
	penguin3	0.785	0.839	0.463	0.752	0.951
	penguin4	0.764	0.862	0.516	0.806	0.920
	penguin5	0.478	0.823	0.537	0.627	0.863
	penguin6	0.843	0.873	0.701	0.755	0.952
	solider	0.553	0.868	0.719	0.665	0.908
	worm	0.654	0.832	0.724	0.347	0.650
	Avg.	0.584	0.675	0.453	0.518	**0.781**

5.3.1 Parameter Verification

With the train set of the DAVIS dataset, we first study the influence of the needed input parameter: number of spatial grids K, of our super-trajectory algorithm in Section 3.2. We report the performance by plotting the IoU value of the segmentation results as functions of a variety of Ks, where we vary $K = \{800, 900, \cdots, 1500\}$. As shown in Fig. 9 (a), the performance increases with finer super-trajectory clustering in spatial domain ($K\uparrow$). However, when we further increase K, the final performance does not change obviously. In our experiments, we set $K = 1200$ where the maximum performance is obtained over the train set of DAVIS.

Later, we investigate the influence of parameter N, which indicates the number of the NNs of a region in Section 4.3. We plot IoU score with varying $N = \{2, 4, \cdots, 20\}$ in Fig. 9 (b), and set $N = 8$ for achieving best performance.

5.3.2 Ablation Study

To quantify the improvement obtained with our proposed trajectories in Section 3.1, we compare to two baseline trajectories: LTM [15] and DAD [66] in our experimental results. LTM is widely used for motion segmentation and DAD shows promising performance for action detection. To be fair, we only replace our trajectory generation part with above two methods, estimate optical flow via LDOF [69] and keep all other parameters fixed.

To further dissect various parts of our method, we offer five variants of the proposed algorithm STV, list as follows:

- STV-s: For demonstrating the effectiveness of the our super-trajectory based label propagation in Section 4.1, we offer baseline STV-s by performing label

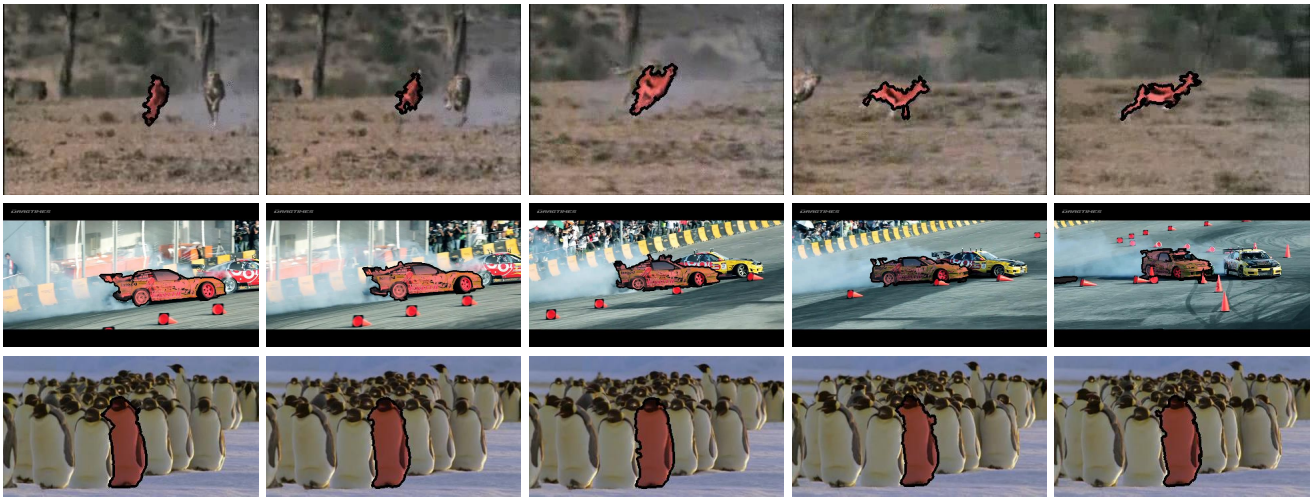


Fig. 8. Qualitative segmentation results on representative video sequences from SegTrack-V2 [7] (from top to bottom: *cheetah1*, *drift1*, and *penguin3*). The initial masks are presented in the first row.

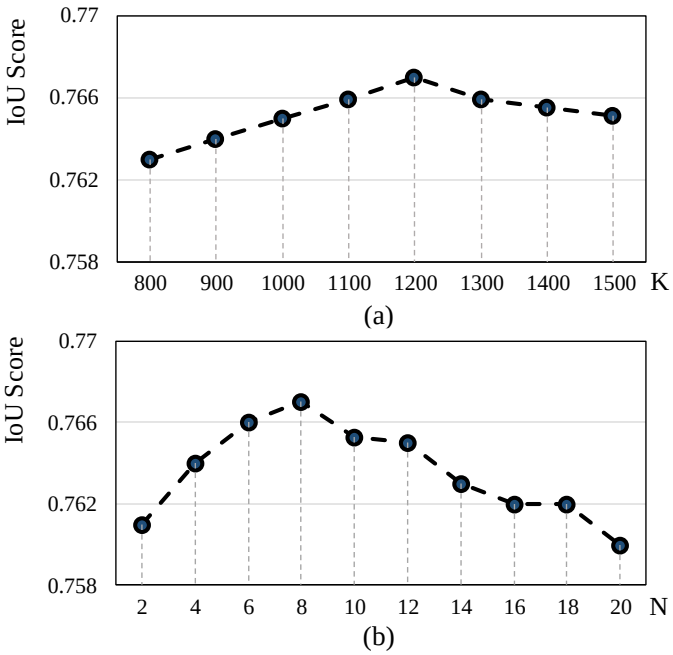


Fig. 9. The IoU scores for parameter selection for number of spatial grids K (a) and the number of the NNs N (b). See Section 5.3.1 for more detailed discussion.

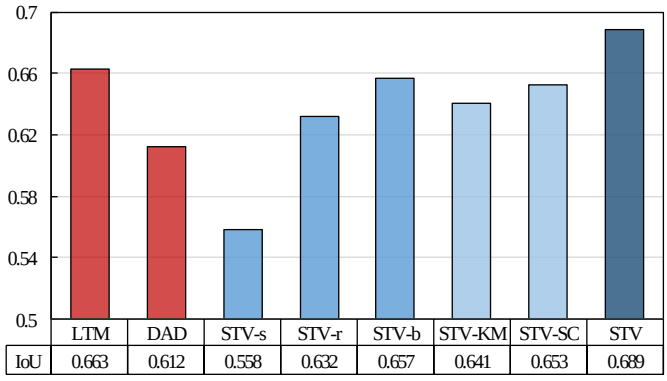


Fig. 10. Average IoU score over the validation set of DAVIS dataset. We compare our method (STV) with two trajectory based methods: LTM [15] and DAD [66], and five variations of our algorithm: STV-s, STV-r, STV-b, STV-KM and STV-SC. See Section 5.3.2 for more details.

propagation on trajectory level and only use labeled-trajectories for establishing appearance model.

- STV-r: For evaluating the effectiveness of the proposed reverse tracking strategy in Section 4.2, we offer baseline STV-r by performing segmentation without considering outside trajectories $\mathcal{T}^o$.
- STV-b: For evaluating the effectiveness of the backward re-occurrence assumption, we offer baseline STV-b via performing segmentation without Eq.16.
- STV-KM: For accessing the influence of the DPC algorithm, we offer baseline STV-KM via replacing DPC with K-means clustering algorithm.
- STV-SC: For accessing the influence of the DPC algorithm, we offer baseline STV-KM via replacing DPC with spectral clustering algorithm.

The comparison results with above baselines are summarized in Fig. 10. Four important conclusions can be drawn: (1) compared with classical trajectory methods [15], [66], the proposed trajectory generation approach is preferable; (2) significant improvement over STV-s (0.689 vs 0.558) clearly demonstrates the advantage of super-trajectory for capturing rich structure information of video; (3) the improvement over STV-r and STV-b verifies the effectiveness of our reverse tracking strategy and global optimization via backward re-occurrence; (4) the DPC algorithm is more favored compared with K-means (0.689 vs 0.641) or spectral clustering algorithm (0.689 vs 0.653).

5.3.3 Attribute-Based Analysis

The videos in the DAVIS dataset are also categorized according to their various attributes such as appearance change (AC), background clutter (BC), camera shake (CS), dynamic background (DB), deformation (DEF), edge ambiguity (EA), fast motion (FM), heterogeneous objects (HO), interesting objects (IO), low resolution (LR), motion blur (MB), occlusion (OCC), out-of-view (OV), shape complexity (SC), scale-

TABLE 5
Attribute-based aggregate performance on the DAVIS dataset with IoU score ($\mathcal{J}$). Higher values are better. The best performing method of each category is highlighted in **boldfaced**.

Dataset	Method	Attribute														
		AC	BC	CS	DB	DEF	EA	FM	HO	IO	LR	MB	OCC	OV	SC	SV
DAVIS	BVS	0.459	0.627	0.621	0.604	0.704	0.575	0.534	0.628	0.628	0.594	0.579	0.681	0.430	0.671	0.490
	FCP	0.509	0.587	0.611	0.622	0.612	0.577	0.554	0.597	0.595	0.589	0.530	0.589	0.526	0.592	0.524
	JMP	0.580	0.607	0.609	0.595	0.593	0.557	0.503	0.561	0.588	0.511	0.509	0.470	0.611	0.525	0.581
	SEA	0.461	0.586	0.424	0.577	0.498	0.516	0.395	0.494	0.543	0.472	0.394	0.470	0.441	0.505	0.491
	TSP	0.174	0.414	0.359	0.400	0.313	0.326	0.177	0.281	0.347	0.308	0.147	0.269	0.215	0.334	0.239
	HVS	0.418	0.622	0.561	0.603	0.590	0.546	0.418	0.541	0.566	0.485	0.435	0.527	0.415	0.572	0.460
	STV-s	0.465	0.571	0.553	0.568	0.537	0.525	0.510	0.547	0.569	0.517	0.493	0.532	0.486	0.527	0.498
	STV-r	0.524	0.653	0.621	0.647	0.704	0.610	0.584	0.647	0.654	0.582	0.550	0.638	0.540	0.657	0.561
	STV-b	0.547	0.674	0.653	0.679	0.730	0.631	0.602	0.663	0.684	0.617	0.583	0.654	0.581	0.692	0.598
	STV	**0.588**	**0.706**	**0.699**	**0.713**	**0.753**	**0.662**	**0.631**	**0.703**	**0.710**	**0.641**	**0.628**	**0.712**	**0.623**	**0.711**	**0.624**

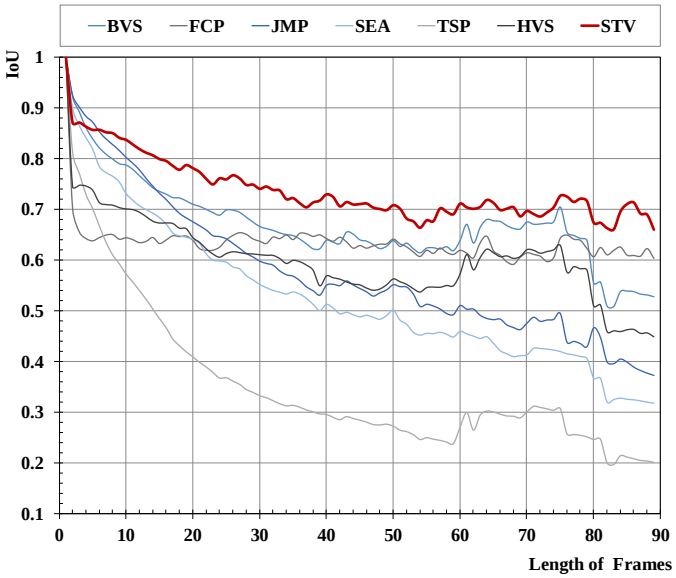


Fig. 11. Segmentation performance with IoU score over time, reported on DAVIS dataset. It can be observed that the IoU decreases during propagation of the initial mask over the consecutive video frames. Our method STV has consistently better performance compared to others.

variation (SV). With these attribute annotations, we present a more detailed evaluation in Table 5. Three variations of the proposed STV method: STV-s, STV-r, and STV-b, described in Section 5.3.2, are also included for a thorough analysis of the effectiveness of our super-trajectory representation, the influence of the proposed reverse tracking scheme and the backward re-occurrence strategies.

The attribute based analysis shows that the proposed video segmentation model, STC, is robust to various challenges presented in the DAVIS dataset. Specifically, it compares favorably on any subset of videos sharing the same attribute. Due to the representation power of the super-trajectory and the efficient DPC algorithm, our segmentation model handles the dynamic background (DB) and motion blur (MB) well. With the reverse tracking strategy, STC is able to discriminate the cases involving the background clutter (BC), and occlusion (OCC). By leveraging on the backward re-occurrence, STV recovers from the out-of-view scenarios (OV) and attains an increased robustness to the deformation (DEF), shape complexity (SC), and scale-variation (SV), which usually degrade methods that strongly rely on propagation of segmentations on a per-frame basis. We also observe significant improvement in cases with scale-variations and low-resolution objects, which are typically failure cases for methods relying on spatiotemporal connections.

5.3.4 *Performance over Time*

For semi-supervised video segmentation, as the number of frames increases, the performance will decrease since errors accumulate over time. In Fig. 11, we plot the IoU scores of different approaches BVS [40], FCP [38], JMP [44], SEA [37], TSP [11] and HVS [9], with the initial annotation propagated over time. It can be observed that, the IoU score of our method drops more slowly and consistently gains better performance over different numbers of propagation frames, compared with other methods. This demonstrates the results of our method experience less drift of the object regions over time.

5.3.5 *Runtime Comparison*

TABLE 6
Runtime comparison (seconds/frame) on the DAVIS [5].

Method	BVS [40]	FCP [38]	HVS [9]	OFL [34]
Time (s)	16 (0.37*)	>50	16	137
Code Type	Matlab&C++	Python	C++	Matlab&C++
Method	SEA [37]	TSP [11]	JMP [44]	STV
Time (s)	6	63	14	9
Code Type	Matlab&C++	Matlab&C++	C++	Matlab&C++

* The runtime reported in [40] is offered for reference, since the code released in https://github.com/owang/BilateralVideoSegmentation is much slower.

We conduct running-time comparisons on DAVIS dataset [5] with 480p image frames. We include six semi-supervised video segmentation methods BVS [40], FCP [38], HVS [9], JMP [44], SEA [37], TSP [11] and OFL [34] for providing a comprehensive view of execution times of existing approaches. The time comparison results (excluding optical computation time) are listed in Table 6. Although we do not have the code of FCP [38], it's computation time must large than 50 seconds per frame. Since FCP is based on object proposal, which takes more than 50 seconds for processing each frame. As seen, our method achieves a better tradeoff

between performance and computation efficiency. All the tests are performed on a Dell T5610 workstation with an Intel Xeon E5 CPU of 2.50 GHz. The proposed method is implemented with MATLAB and C++.

6 Conclusions and Future Work

We introduced a super-trajectory representation based semi-supervised video segmentation approach. We demonstrated that, based on the density peaks based clustering, compact trajectories can be efficiently grouped into super-trajectories. Super trajectory possesses various desired properties and capable of capturing: i) long-term motion information, ii) local spatiotemporal information, and iii) diverse and compact features of video. We showed that, in our context, occlusion and drift are naturally handled by our trajectory generation method using the probabilistic model. Our solution for reverse tracking points and our approach to leverage the property of region re-occurrence both lead an improved robustness for many segmentation challenges such as occlusions and move-in/-out.

By extensive experimental evaluations on three large video segmentation datasets [5], [6], [7], we verified that our approach outperforms the current non-deep learning based and heuristic methods. We also analyzed several variants and components for a comprehensive assessment of various aspects of our method.

One potential direction for future work is to combine our super-trajectory with deep learning descriptors, as the work of [68], for a more powerful representation of video sequences. Additionally, our work provides valuable evidence toward combining compact spatiotemporal representation with certain priors (*e.g.*, saliency, objectness) for other computer vision applications such as action recognition.

References

[1] M. Hoai, Z.-Z. Lan, and F. De la Torre, "Joint segmentation and classification of human actions in video," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 3265–3272.

[2] J. Son, I. Jung, K. Park, and B. Han, "Tracking-by-segmentation with online gradient boosting decision tree," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3056–3064.

[3] X. Liu, D. Tao, M. Song, Y. Ruan, C. Chen, and J. Bu, "Weakly supervised multiclass video segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 57–64.

[4] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.

[5] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. V. Gool, M. Gross, and A. Sorkine-Hornung, "A benchmark dataset and evaluation methodology for video object segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 724–732.

[6] A. Prest, C. Leistner, J. Civera, C. Schmid, and V. Ferrari, "Learning object class detectors from weakly annotated video," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3282–3289.

[7] F. Li, T. Kim, A. Humayun, D. Tsai, and J. M. Rehg, "Video segmentation by tracking many figure-ground segments," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2192–2199.

[8] W. Wang, J. Shen, J. Xie, and P. Fatih, "Super-trajectory for video segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[9] M. Grundmann, V. Kwatra, M. Han, and I. Essa, "Efficient hierarchical graph-based video segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2141–2148.

[10] C. Xu, C. Xiong, and J. J. Corso, "Streaming hierarchical video segmentation," in *European Conference on Computer Vision*, 2012, pp. 626–639.

[11] J. Chang, D. Wei, and J. W. Fisher, "A video representation using temporal superpixels," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2051–2058.

[12] C. Xu and J. J. Corso, "Evaluation of super-voxel methods for early video processing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1202–1209.

[13] S. D. Jain and K. Grauman, "Supervoxel-consistent foreground propagation in video," in *European Conference on Computer Vision*, 2014, pp. 656–671.

[14] T. Brox and J. Malik, "Object segmentation by long term analysis of point trajectories," in *European Conference on Computer Vision*, 2010, pp. 282–295.

[15] K. Fragkiadaki, G. Zhang, and J. Shi, "Video segmentation by tracing discontinuities in a trajectory embedding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1846–1853.

[16] A. Papazoglou and V. Ferrari, "Fast object segmentation in unconstrained video," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 1777–1784.

[17] A. Vazquez-Reina, S. Avidan, H. Pfister, and E. Miller, "Multiple hypothesis video segmentation from superpixel flows," in *European Conference on Computer Vision*, 2010, pp. 268–281.

[18] D. Oneata, J. Revaud, J. Verbeek, and C. Schmid, "Spatio-temporal object detection proposals," in *European Conference on Computer Vision*, 2014, pp. 737–752.

[19] A. Faktor and M. Irani, "Video segmentation by non-local consensus voting." in *Proceedings of the British Machine Vision Conference*, vol. 2, no. 7, 2014, p. 8.

[20] W. Wang, J. Shen, and F. Porikli, "Saliency-aware geodesic video object segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3395–3402.

[21] Y. J. Lee, J. Kim, and K. Grauman, "Key-segments for video object segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2011, pp. 1995–2002.

[22] T. Ma and L. J. Latecki, "Maximum weight cliques with mutex constraints for video object segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 670–677.

[23] D. Zhang, O. Javed, and M. Shah, "Video object segmentation through spatially accurate and temporally dense extraction of primary object regions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 628–635.

[24] K. Fragkiadaki, P. Arbelaez, P. Felsen, and J. Malik, "Learning to segment moving objects in videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4083–4090.

[25] F. Xiao and Y. Jae Lee, "Track and segment: An iterative unsupervised approach for video object proposals," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 933–942.

[26] Y. J. Koh and C.-S. Kim, "Primary object segmentation in videos based on region augmentation and reduction," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7417–7425.

[27] I. Endres and D. Hoiem, "Category independent object proposals," in *European Conference on Computer Vision*, 2010, pp. 575–588.

[28] V. Badrinarayanan, F. Galasso, and R. Cipolla, "Label propagation in video sequences," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 3265–3272.

[29] D. Tsai, M. Flagg, and J. M. Rehg, "Motion coherent tracking using multi-label MRF optimization," *Proceedings of the British Machine Vision Conference*, 2010.

[30] I. Budvytis, V. Badrinarayanan, and R. Cipolla, "Semi-supervised video segmentation using tree structured graphical models," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 2257–2264.

[31] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Hypercolumns for object segmentation and fine-grained localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 447–456.

[32] W. Wang, J. Shen, and F. Porikli, "Selective video object cutout," *IEEE Transactions on Image Processing*, vol. 26, no. 12, pp. 5645–5655, 2017.
[33] N. Shankar Nagaraja, F. R. Schmidt, and T. Brox, "Video segmentation with just a few strokes," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3235–3243.
[34] Y.-H. Tsai, M.-H. Yang, and M. J. Black, "Video segmentation via object flow," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3899–3908.
[35] W. Brendel and S. Todorovic, "Video object segmentation by tracking regions," in *Proceedings of the IEEE International Conference on Computer Vision*, 2009, pp. 833–840.
[36] S. Vijayanarasimhan and K. Grauman, "Active frame selection for label propagation in videos," in *European Conference on Computer Vision*, 2012, pp. 496–509.
[37] S. Avinash Ramakanth and R. Venkatesh Babu, "SeamSeg: Video object segmentation using patch seams," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 376–383.
[38] F. Perazzi, O. Wang, M. Gross, and A. Sorkine-Hornung, "Fully connected object proposals for video segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3227–3234.
[39] L. Wen, D. Du, Z. Lei, S. Z. Li, and M.-H. Yang, "JOTS: Joint online tracking and segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2226–2234.
[40] N. Märki, F. Perazzi, O. Wang, and A. Sorkine-Hornung, "Bilateral space video segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 743–751.
[41] X. Bai, J. Wang, D. Simons, and G. Sapiro, "Video SnapCut: robust video object cutout using localized classifiers," *ACM Transactions on Graphics)*, vol. 28, no. 3, p. 70, 2009.
[42] A. Criminisi, T. Sharp, C. Rother, and P. Pérez, "Geodesic image and video editing," *ACM Transactions on Graphics*, vol. 29, no. 5, pp. 134–1, 2010.
[43] F. Zhong, X. Qin, Q. Peng, and X. Meng, "Discontinuity-aware video object cutout," *ACM Transactions on Graphics*, vol. 31, no. 6, p. 175, 2012.
[44] Q. Fan, F. Zhong, D. Lischinski, D. Cohen-Or, and B. Chen, "JumpCut: non-successive mask transfer and interpolation for video cutout." *ACM Transactions on Graphics*, vol. 34, no. 6, pp. 195–1, 2015.
[45] P. Tokmakov, K. Alahari, and C. Schmid, "Learning motion patterns in videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 531–539.
[46] D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan, "Learning features by watching objects move," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
[47] P. Tokmakov, K. Alahari, and C. Schmid, "Learning video object segmentation with visual memory," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017.
[48] S. Jain, B. Xiong, and K. Grauman, "Fusionseg: Learning to combine motion and appearance for fully automatic segmention of generic objects in videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
[49] V. Jampani, R. Gadde, and P. V. Gehler, "Video propagation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
[50] F. Perazzi, A. Khoreva, R. Benenson, B. Schiele, and A. Sorkine-Hornung, "Learning video object segmentation from static images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
[51] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool, "One-shot video object segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
[52] W.-D. Jang and C.-S. Kim, "Online video object segmentation via convolutional trident network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5849–5858.
[53] J. Cheng, Y.-H. Tsai, S. Wang, and M.-H. Yang, "Segflow: Joint learning for video object segmentation and optical flow," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 686–695.
[54] P. Voigtlaender and B. Leibe, "Online adaptation of convolutional neural networks for video object segmentation," in *Proceedings of the British Machine Vision Conference*, 2017.
[55] X. Li, Y. Qi, Z. Wang, K. Chen, Z. Liu, J. Shi, P. Luo, C. C. Loy, X. Tang, A. Khoreva *et al.*, "Video object segmentation with re-identification," in *The 2017 DAVIS Challenge on Video Object Segmentation-CVPR Workshops*, 2017.
[56] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European Conference on Computer Vision*, 2014, pp. 740–755.
[57] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.
[58] J. Shi and C. Tomasi, "Good features to track," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1994, pp. 593–600.
[59] N. Sundaram, T. Brox, and K. Keutzer, "Dense point trajectories by GPU-accelerated large displacement optical flow," in *European Conference on Computer Vision*, 2010, pp. 438–451.
[60] K. Fragkiadaki and J. Shi, "Detection free tracking: Exploiting motion and topology for segmenting and tracking under entanglement," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 2073–2080.
[61] J. Lezama, K. Alahari, J. Sivic, and I. Laptev, "Track to the future: Spatio-temporal video segmentation with long-range motion cues," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 3369–3376.
[62] K. Fragkiadaki, W. Zhang, G. Zhang, and J. Shi, "Two-granularity tracking: Mediating trajectory and detection graphs for tracking under occlusions," in *European Conference on Computer Vision*, 2012, pp. 552–565.
[63] P. Ochs and T. Brox, "Higher order motion models and spectral clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 614–621.
[64] P. Ochs, J. Malik, and T. Brox, "Segmentation of moving objects by long term video analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 6, pp. 1187–1200, 2014.
[65] M. Keuper, B. Andres, and T. Brox, "Motion trajectory segmentation via minimum cost multicuts," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3271–3279.
[66] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Action recognition by dense trajectories," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 3169–3176.
[67] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 3551–3558.
[68] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4305–4314.
[69] T. Brox and J. Malik, "Large displacement optical flow: descriptor matching in variational motion estimation," *IEEE transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 500–513, 2011.
[70] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012.
[71] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 509–522, 2002.

Wenguan Wang received the B.S. degree in computer science and technology from the Beijing Institute of Technology in 2013. He is currently working toward the Ph.D. degree in the School of Computer Science, Beijing Institute of Technology, Beijing, China. His current research interests include computer vision and deep learning. He received the Baidu Scholarship in 2016.

Jianbing Shen (M'11-SM'12) is a Professor with the School of Computer Science, Beijing Institute of Technology, Beijing, China. He has published about 70 journal and conference papers such as *IEEE Trans. on Pattern Analysis and Machine Intelligence*, *IEEE CVPR*, and *IEEE ICCV*. He has also obtained many flagship honors including the Fok Ying Tung Education Foundation from Ministry of Education, the Program for Beijing Excellent Youth Talents from Beijing Municipal Education Commission, and the Program for New Century Excellent Talents from Ministry of Education. His research interests include computer vision and graphics. He is on the editorial boards of *Neurocomputing*.

Fatih Porikli (M'96,SM'04,F'14) is an IEEE Fellow and a Professor in the Research School of Engineering, Australian National University (ANU). He is also acting as the Chief Scientist at Huawei, Santa Clara. He has received his Ph.D. from New York University in 2002. Previously he served Distinguished Research Scientist at Mitsubishi Electric Research Laboratories. His research interests include computer vision, pattern recognition, manifold learning, image enhancement, robust and sparse optimization and online learning with commercial applications in video surveillance, car navigation, intelligent transportation, satellite, and medical systems.

Ruigang Yang received the MS degree from Columbia University in 1998 and the PhD degree from the University of North Carolina, Chapel Hill in 2003. He is currently a full professor of Computer Science at the University of Kentucky. His research interests span over computer vision and computer graphics, in particular in 3D reconstruction and 3D data analysis. He has published more than 100 papers, which, according to Google Scholar, has received close to 6,000 citations with an h-index of 37 (as of 2014). He is currently an associate editor of the *IEEE Transactions on Pattern Analysis and Machine Intelligence* and a senior member of IEEE.