Polygonal Point Set Tracking

Gunhee Nam[1]* Miran Heo[2] Seoung Wug Oh[3] Joon-Young Lee[3] Seon Joo Kim[2]

[1]Lunit Inc. [2]Yonsei University [3]Adobe Research

Abstract

In this paper, we propose a novel learning-based polygonal point set tracking method. Compared to existing video object segmentation (VOS) methods that propagate pixel-wise object mask information, we propagate a polygonal point set over frames. Specifically, the set is defined as a subset of points in the target contour, and our goal is to track corresponding points on the target contour. Those outputs enable us to apply various visual effects such as motion tracking, part deformation, and texture mapping. To this end, we propose a new method to track the corresponding points between frames by the global-local alignment with delicately designed losses and regularization terms. We also introduce a novel learning strategy using synthetic and VOS datasets that makes it possible to tackle the problem without developing the point correspondence dataset. Since the existing datasets are not suitable to validate our method, we build a new polygonal point set tracking dataset and demonstrate the superior performance of our method over the baselines and existing contour-based VOS methods. In addition, we present visual-effects applications of our method on part distortion and text mapping.

1. Introduction

Object mask tracking in a video is one of the most frequently required tasks in visual effects (VFX). However, the task (*i.e.* rotoscoping) is so painstaking and time-consuming that even a highly-skilled designer processes only a dozen frames on average per day [28]. Therefore, propagating object mask information through subsequent frames becomes a critical problem to reduce human labor for rotoscoping. Propagation methods are categorized into four groups based on object representations: point, region, contour, and polygonal point set (Figure 2). Each representation carries different amount of information.

Patch tracking [14, 33, 13, 7] denotes a target object as *point* representation and tracks a target point over frames by matching the patch around the point. The tracking enables visual effects that require positional information such as motion tracking and texture mapping. These applications often require multiple patch tracking to compute point-to-point information. However, conducting each patch tracking independently ignores strong correlations between target points, thus multiple patch tracking are susceptible to a drift problem and are not suitable for mask propagation.

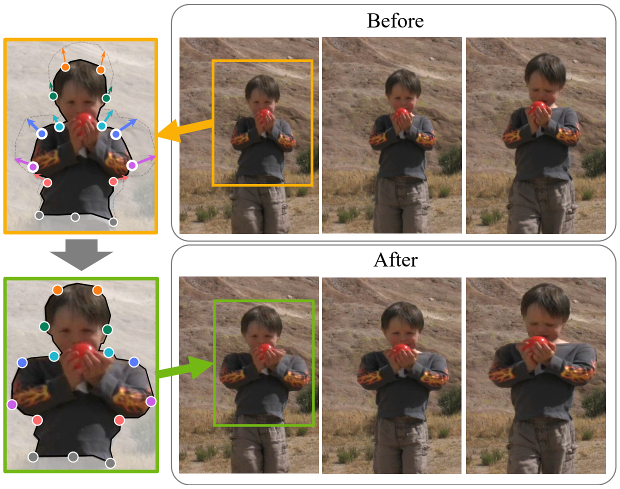


Figure 1: Our method tracks a set of points in a polygon over frames. The output represents mask contour with point correspondences across frames. It allows multiple applications, *e.g.*, a non-rigid transformation of a specific part of an object over time as shown here.

Meanwhile, video object segmentation (VOS) [44, 8, 22, 47, 35, 34] and contour tracking [20, 49, 11, 39] propagate target object information over subsequent frames by representing the target as a *region* (*i.e.* mask) and a *contour* respectively. These representations can describe only the target area without any pixel correspondences, therefore they are not applicable to complex VFX scenarios that require point-to-point relation information (*e.g.*, Figure 1).

On the other hand, *polygonal point set* tracking combines the positional information with the target region. The polygonal point set is defined as a subset of contour points that represent an object in a polygonal shape. By tracking the point set, we can get both the object contour and the point-to-point matching information. Previous works in this category [2, 28, 32, 37] focus on making the user inter-

*Work mostly done during a M.S. student at Yonsei University

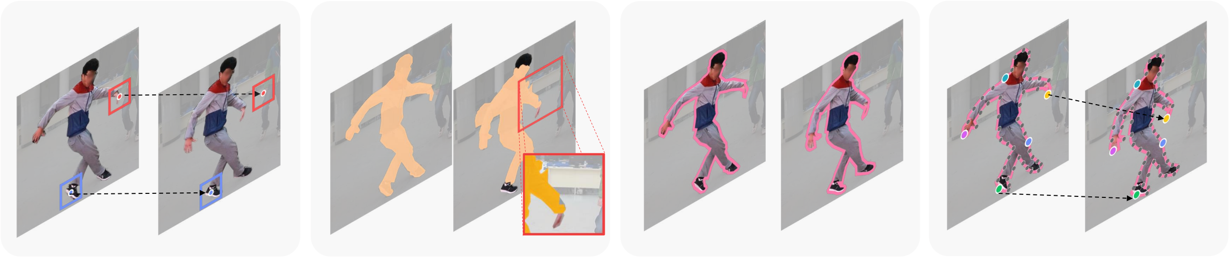

(a) Patch Tracking (b) Video Object Segmentation (c) Contour Tracking (d) Point Set Tracking (Ours)

Figure 2: Illustration of different approaches for object mask propagation. (a) Independent multi-patch tracking represents an object coarsely and drifts easily. (b) Region-based video object segmentation achieves high accuracy in pixel-level dense prediction, but it usually may yield cattery boundaries due to a high degree of freedom and does not provide point correspondences. (c) In contour tracking, the constrained contour representation can give us clean boundary, but point correspondence information is still missing. On the other hand, (d) Polygonal point set tracking provides both clean polygonal object mask and point correspondences across frames.

action easier for highly customizable results rather than taking the point-to-point matching (or tracking) into account. Therefore, they assume heuristic shape priors of an object and often exhibit propagation failures for challenging object motions.

In this paper, we aim to track all points directly through a learning-based approach without assuming a heuristic shape prior and propose a novel point set tracking method. According to the hypothesis that the target object state in adjacent frames is highly correlated, we train a network to learn progressive alignment of a point set between frames. We first match the point set globally using a simple rigid transformation. Then, we further tune each point position in a coarse to fine manner using a local alignment module. Our local alignment module (LAM) adopts recurrent neural networks (RNN) to take into account the temporal history of each point and also uses multi-head attention (MHA) modules for non-local communication among the points. In addition, we regularize the alignments to avoid drifting and honor the original topology of a target polygon in challenging situations.

We introduce a new learning strategy to train our model without fully annotated data. Currently available VOS datasets [9, 48] only contain region information (*i.e.* masks), thus it cannot be directly applied to our point correspondence learning. To overcome the data issue, we propose an unsupervised learning method based on cyclic consistency between predicted frames [45, 46]. We also obtain the supervision for point set tracking by synthesizing data from image instance segmentation data. To the best of our knowledge, this is the first work on learning-based point set tracking that considers the point correspondence.

Figure 1 shows an example application that utilizes our network results. In this example, to exaggerate the upper body, the same effect is applied to the target across the entire frame, even if the user edits the points in the set individually only in the first frame.

Popular evaluation datasets for video object segmentation are not suitable for evaluating point-set tracking as they only provide mask annotations [9, 16, 27, 32]. While CPC [32] provides annotations of object contours, it is also not sufficient for the evaluation as there is no point correspondence annotation. To this end, we introduce an evaluation dataset for polygonal point set tracking, consisting of 30 sequences. To build the dataset, named PoST, we augment video clips from existing VOS datasets with additional point set annotations with correspondence. We evaluate our method on PoST and the existing VOS datasets [9, 32], and we show that our method outperforms competing methods with a large margin.

Our contributions are summarized as follows:

- We propose a novel learning-based method for polygonal point set tracking with point correspondence for the first time.
- We design a local alignment module for point tracking with taking temporal history and communication with other points into account.
- We present a learning strategy to train a deep network for point set tracking using unsupervised learning and synthesized data.
- We introduce a new dataset for evaluating performance of point set tracking.

2. Related Work

Object Mask Propagation Patch tracking is a naive approach of the object mask propagation at the point level. By tracking a given patch through all frames, the patch location provides the positional information of the tracked part of an object across frames. However, recent research based on a deep learning approach has focused on object-level tracking rather than patch-level because it is hard to annotate corresponding patches over frames [4, 5, 12]. Although optimization-based methods [14, 33, 13, 7] bypass this data issue, it is not robust enough since they rely on hand-crafted features. Moreover, while multiple patch tracking is performed on the different parts of the same object in the case of VFX, such as texture mapping, they drift easily because each patch is tracked independently.

On the other hand, region-based video object segmentation estimates pixel-wise masks. In this approach, the user-supplied mask is temporally propagated to other frames to aid the time-consuming per frame segmentation. Recently, thanks to the representational power of deep learning, the region-based methods have reached a milestone in its object mask propagation performance [44, 8, 22, 47, 35, 34]. Despite the success of the aforementioned methods, its shape representation inherently limits many editing applications as it cannot provide point-to-point information.

Contour tracking methods [20, 49, 11, 39] also propagate the object masks but use a more constrained representation (*i.e*. object boundary). Contour tracking has been performed by probabilistic [20, 49] and hidden Markov models [11]. More recently, Saboo *et al*. [39] propose a learning-based framework that solves the problem by adopting an attention mechanism. By employing the contour representation, these methods reduce prediction noise and yield a clean object boundary but still lack temporal point-to-point correspondence similar to the region-based object segmentation methods.

In polygonal point set tracking, different from region-based segmentation and contour tracking, each point can be tracked. In this problem setting, various cues for propagating an object mask are previously explored. In [2], an interpolation between two key frames is performed. Some methods try to find sharp object edges using snakes [6, 23] after a global tracking through either using shape manifold [28] or shape prior [32, 37]. However, the previous point set tracking methods rather focus on its convenience for user interaction for controlling object shapes than establishing an accurate point-to-point correspondence. Thus, many of them are limited to applications that require matching points over time.

Point Set Representation in Deep Learning While point set representation is not popular in modern deep learning architectures in computer vision, there are some efforts to employ it. To find an efficient way against manual annotation for segmentation, the shape representation is defined as polygon structure in [10, 1, 31]. In these methods, several architectures of recurrent neural networks (RNN) [10], graph neural network (GNN) [1] and graph convolution network (GCN) [31] are suggested to deal with point set in deep approach. Point set representation is also employed for image instance segmentation as well [36, 29]. Other options to handle the representation are proposed in those methods such as circular convolution [36] and transformer [29]. These previous methods show considerable potential for the shape representation of point set but only focus on a single image level. Point set tracking is inherently impossible for these image-level approaches.

Global-local alignment Since adjacent frames in a video sequence are highly correlated, global-local alignment is a common approach in many video-related tasks such as optical flow [41], video inpainting [26], and object tracking [15]. Although not a learning-based nor point-to-point tracking method, SnapCut [3] is also one of the most popular rotoscoping approaches that shares a similar philosophy with ours, where patches are tracked first and then the local classifier refines the contour. Following this concept, we adopt the global-local alignment in our framework.

3. Method

Our method tracks a set of polygonal points representing a subset of points on the contour of the target object. As shown in Figure 3, our framework is divided into two steps: global and local alignments. For global alignment, we compute an affine transformation matrix between the previous and the current frames. We globally align the previous frame and its point set using the computed matrix. In the second step, we forward the transformed frame and point set to a local feature encoder, and extract point-wise features for each point in the set. We forward the extracted point features to Local Alignment Module (LAM) and update the point correspondences locally. LAM computes the offset of each point to update the point's location progressively in a coarse to fine manner.

3.1. Global Alignment

Since the deformation of an object or viewpoint change is small between adjacent frames, a simple geometric transformation like affine transform can align the two frames to some extent. Inspired by [21, 26], the global alignment network predicts an affine transformation matrix to align the previous frame $\mathcal{I}_{t-1}$ toward the current frame $\mathcal{I}_t$. To estimate the affine transformation matrix $\mathbf{A}_{t-1\to t}$, the binary target mask $\hat{\mathcal{M}}_{t-1}$ obtained from the polygonal point set $\hat{\mathbf{P}}_{t-1}$ is also given as an additional input into the global alignment network, allowing it to focus on the target while disregarding the background. For computational efficiency, the input is resized into a half and a lightweight backbone is used for the global alignment network as in [26]. In short, $\mathbf{A}_{t-1\to t}$ is obtained by the global alignment network $f_{\text{glob}}(\cdot)$ as follows:

$$\mathbf{A}_{t-1\to t} = f_{\text{glob}}(\mathcal{I}^{\downarrow}_t, \mathcal{I}^{\downarrow}_{t-1}, \hat{\mathcal{M}}^{\downarrow}_{t-1}), \tag{1}$$

where $\mathcal{I}^{\downarrow}$ and $\hat{\mathcal{M}}^{\downarrow}$ denote downsized image and mask respectively. The output of the network is a 6-dimensional vector that represents an affine matrix. The previous frame

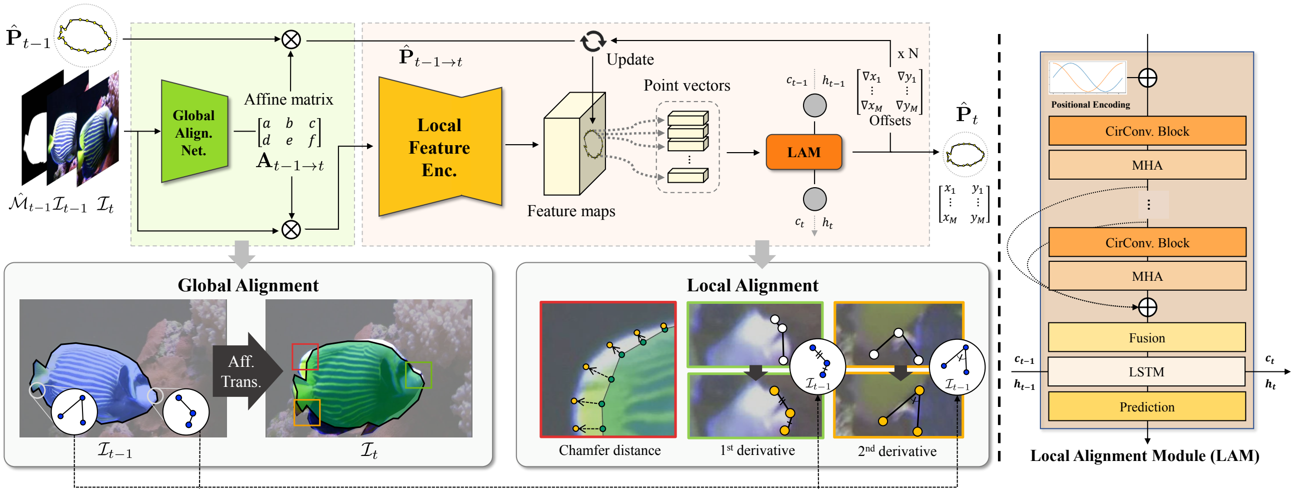


Figure 3: Overview of our framework. In our framework, the alignment steps are divided into global and local alignment. First, the previous frame and point set are globally aligned by the affine transform matrix from the global alignment networks. Then, the features encoded from the current and aligned inputs are used for local alignment after sampled as point vectors. Local alignment module (LAM) yields displacement offsets of points in the set from point vectors and updates the point set. The local alignment performs iteratively in a coarse to fine manner.

$\mathcal{I}_{t-1}$ and its point set $\hat{\mathbf{P}}_{t-1}$ are then warped by the computed $\mathbf{A}_{t-1\to t}$ into $\mathcal{I}_{t-1\to t}$ and $\hat{\mathbf{P}}_{t-1\to t}$.

3.2. Local Alignment

After the global alignment, each point in the point set is further aligned locally. We extract point features using a local feature encoder. Specifically, We use `resnet50` [18] as an encoder backbone and take FPN feature maps [30] for accurate localization. We feed the current frame $\mathcal{I}_t$, the warped previous frame $\mathcal{I}_{t-1\to t}$ and the warped target mask $\hat{\mathcal{M}}_{t-1\to t}$ to the encoder after concatenating them along the channel axis.

From the encoded feature maps, we sample point feature vectors according to the location of each point in the point set. Given the point vectors, our *Local Alignment Module* (LAM) calculates the displacement offset of each point. The locations of the points are updated using the offsets, and we repeat the point feature sampling and the offset update. In each iteration, we use the feature maps at a different scale from coarse to the finest levels, thus the point set is aligned in a coarse-to-fine manner.

Local Alignment Module (LAM). Local Alignment Module (LAM) takes a set of point feature vectors and yields 2-channeled offsets (*i.e.* horizontal and vertical), where each 2-channel offset corresponds to each point. Figure 3 describes the detailed architecture of LAM. The module can be divided into four parts: *positional encoding*, *backbone*, *temporal information transfer*, and *prediction*.

Positional encoding allows the network to identify the order of each point in the set. We use the sinusoidal positional embedding as in [43]. However, in our positional embedding, the first point should meet the last point reflecting the cyclic characteristic of a polygon. Therefore, we adjust the period of the sinusoidal function according to the number of points N in the point set as follows:

$$\mathbf{E}^i_{\text{order}} = [\sin(2\pi i/N), \cos(2\pi i/N)], \tag{2}$$

where $\mathbf{E}^i_{\text{order}}$ denotes the positional embedding of i^{th} vertex.

Backbone is constructed by stacking 8 base blocks where each block consists of two circular convolutions [36] followed by multi-head attention (MHA) [43]. In the last layer, features from each level are fused by concatenation and an 1x1 convolution followed by max pooling. *Temporal information transfer* improves the temporal consistency of the estimation by taking the history of each point. We use Long Short-Term Memory (LSTM) [19] for this purpose. Finally, *prediction* layer outputs the offset of each point.

4. Training

4.1. Objective Function

Global Alignment. To train the global alignment network, we use two losses: point set matching loss and pixel matching loss. The point set matching loss [31] considers a group of points by measuring a global distance, where each point of two different point sets is matched one-to-one with each other along the polygonal path. As there are multiple global distances between two point sets depending on the starting index, we minimize the minimum of all possible distances. With a point set with starting index k, $\mathbf{P^k} = [\mathbf{P}^{k\%N}, \mathbf{P}^{(k+1)\%N}, ..., \mathbf{P}^{(k+N-1)\%N}]$, the point set matching loss $\mathcal{L}_g$ is defined as follows:

$$\mathcal{L}_g = \min_{k=[0,\ldots,N-1]} \sum_{i=0}^{N-1} \left\| \mathbf{P}^{\mathbf{k}i} - \hat{\mathbf{P}}^i \right\|_1, \tag{3}$$

where $\mathbf{P}^i$ and $\hat{\mathbf{P}}^i$ denote the points each from the ground-truth and the globally-aligned point sets with index i ($0 \leq i < N$) respectively, and $||\cdot||$ denotes smooth L1 distance [17].

Pixel matching loss is based on the hypothesis that the brightness (or color) of the aligned pixel from the previous frame should be similar to that of the corresponding pixel in the current frame [26]. The pixel matching loss $\mathcal{L}_\text{p}$ is calculated as follows:

$$\mathcal{L}_\text{p} = \frac{1}{K}\sum_{x=0}^{W-1}\sum_{y=0}^{H-1} \mathbb{1}_\text{obj}^{xy} \left\| \mathcal{I}_t^{xy} - \mathcal{I}_{t-1\to t}^{xy} \right\|_2, \tag{4}$$

where $\mathbb{1}_\text{obj}^{xy}$ is an indicator function to check if pixel (x, y) belongs to the target mask in the current frame $\mathcal{I}_t$ whose width and height is W and H respectively, and $K = \sum_{x=0}^{W-1}\sum_{y=0}^{H-1} \mathbb{1}_\text{obj}^{xy}$.

Local Alignment. We have two different scenarios in training and use different objective functions for the local alignment in each scenario. When we have ground-truth point correspondences (*e.g.*, synthetic data), we use the smooth L1 loss between two matching points as $\mathcal{L}_c = \sum_{i=0}^{N-1} ||\mathbf{P}^i - \hat{\mathbf{P}}^i||_1$.

Another scenario is with existing VOS datasets, having only ground-truth masks without point correspondences. In this case, we sample a polygonal point set along the ground-truth mask boundary at each frame. Then, we adopt Chamfer distance between the predicted point set and the target point set as our objective for local alignment. The objective encourages each predicted point to be mapped to a point on the object boundary and is formally defined as follows:

$$\begin{aligned}\mathcal{L}_c = \frac{1}{N}\sum_{i=0}^{N-1} \min_{j=[0,\ldots,N-1]} ||\mathbf{P}^i - \hat{\mathbf{P}}^j||_2 \\ + \frac{1}{N}\sum_{j=0}^{N-1} \min_{i=[0,\ldots,N-1]} ||\mathbf{P}^i - \hat{\mathbf{P}}^j||_2.\end{aligned} \tag{5}$$

In our implementation, $M = N$ as we sample the same number of points at each frame.

In addition to the correspondence objective, we include regularization terms into our objective function to avoid drifting in challenging situations, *e.g.*, a corresponding point is occluded. We want to honor the previous shape topology in the case, therefore we make use of the first and second derivative regularization ($\mathcal{R}_1$ and $\mathcal{R}_2$) on the predicted point set and prevent dramatic changes in the length and the angle of the point set. The regularization terms are defined as follows:

$$\mathcal{R}_1 = \sum_{i=0}^{N-2} (||\hat{\mathbf{P}}_t^i - \hat{\mathbf{P}}_t^{i-1}||_2 - ||\hat{\mathbf{P}}_{t-1}^i - \hat{\mathbf{P}}_{t-1}^{i-1}||_2)^2, \tag{6}$$

$$\begin{aligned}\mathcal{R}_2 = \sum_{i=0}^{N-3} ||(\hat{\mathbf{P}}_t^{i+1} - 2\hat{\mathbf{P}}_t^i + \hat{\mathbf{P}}_t^{i-1}) \\ - (\hat{\mathbf{P}}_{t-1}^{i+1} - 2\hat{\mathbf{P}}_{t-1}^i + \hat{\mathbf{P}}_{t-1}^{i-1})||_2.\end{aligned} \tag{7}$$

We observe that these regularization terms greatly improve the stability of our model outputs.

Unsupervised Learning. To further improve the performance of our model when training on a dataset without point matching annotations, we employ an unsupervised approach for the correspondence learning [46, 45]. By exploiting the cycle-consistency in time, we can assume that a point set tracked forward-then-backward should be matched to the point set at the same location. To make a cycle, we run the network forward K frames as usual, and then we run the network backward from the outputs of the forward pass to the initial frame. By doing so, we can derive matching points between the predictions during the forward and the backward pass. From the point set collected during the forward and backward pass ($\hat{\mathbf{P}}$ and $\hat{\mathbf{Q}}$), an unsupervised loss is defined as $\mathcal{L}_u = \frac{1}{KN}\sum_{k=0}^{K-1}\sum_{i=0}^{N-1} ||\hat{\mathbf{P}}_k^i - \hat{\mathbf{Q}}_k^i||_1$.

4.2. Data Augmentation by Synthetic Data

Supervision for point-to-point matching is crucial for our model to learn polygonal point set tracking with accurate correspondences. However, this information is not available in existing datasets. To complement it, we take an image instance segmentation dataset [25] and synthesize video data with full supervision signals. We first crop one or two objects from randomly sampled images in the dataset. For each cropped object, we extract a polygonal point set and deform it using the moving least squares method [40] with randomly chosen control points within the point set. These deformed objects are randomly pasted using the linear blending to a background image that is also randomly sampled. Then, we generate a sequence with synthetic movement by applying random affine transforms to each object and background. This procedure allows us to generate data with full supervisory signals, and we used it to augment our training data.

5. Experiments

5.1. Implementation Details

To train our model, we use video object segmentation datasets, including the training sets from YouTube-VOS [48] and DAVIS [9]. We randomly choose a short clip from each sequence and extract contours from ground-truth masks using the method proposed in [42]. We filter out samples containing partial occlusion, where a contour is divided into several pieces as they cannot be tracked over frames. From a contour, a polygonal point set is sampled for the network input. In addition to the real data, we also make use of the synthetic dataset as described in Section 4.2.

We train our model using Adam [24] optimizer with the initial learning rate of 0.0001, decayed by a factor of 10 after 70k iterations. The backbone network of the local alignment module is initialized with pretrained weights on ImageNet [38]. Our model is trained for 100k iterations with a batch size of 8 on four NVIDIA RTX 2080 Ti GPUs for 4 days. The number of sampled frames increases over epochs from 2 to 7.

We uniformly sample 128 vertices from the contour for the polygonal point set during both training and test time. An input image is cropped into a patch based on the bounding box of the predicted contour with margin. The cropped area is defined as the method proposed in [4]. For the local alignment, the point set is updated with 5 iterations, and we use the FPN layer with stride from 1/32 to 1/4 as our feature map for each iteration except the last iteration. For the last iteration, the layer with stride 1/4 is used again.

5.2. Evaluation Dataset

Video Object Segmentation Datasets. Because our method is closely related to the video object segmentation task, we evaluate our model on DAVIS2016 [9], one of the most popular benchmarks for the task. The dataset consists of 20 sequences under challenging scenarios. Because only mask annotation is given for each target object in the dataset, we sample a point set as described in Section 5.1.

For the evaluation of contour-based video segmentation, Lu *et al*. [32] proposed CPC dataset, where target object motions are mostly rigid. The dataset consists of 9 sequences of 34 frames in average without any training data. These sequences are annotated by professional designers using a standard editing tool. The annotation is given as a parametric line of Bézier curves.

PoST. To evaluate our polygonal point set tracking properly, we need annotations not only object masks but also point correspondences across frames. However, the existing VOS benchmark datasets aim to evaluate the quality of object masks only. CPC [32] has parametric contour annotations, but its control points do not correspond with each other across frames. Furthermore, CPC evaluation is not reliable because it only contains nine sequences, and each sequence has mostly too small motions resulting in saturated performance.

To this end, we propose a new challenging dataset for point set tracking, named as PoST (**Po**int **S**et **T**racking). We take a few sequences from the existing datasets of DAVIS [9], CPC [32], SegTrack v2 [27] and JumpCut[16] in order to cover various target object classes in different video characteristics. To ensure that the point set tracking is possible, we avoid sequences where a target object exhibits extreme occlusions. For each sequence, we annotate point set correspondences every 10 frames throughout the sequence. If there are no accurate corresponding points in a specific frame, we marked the points and excluded them from the evaluation. In the end, we annotated 30 sequences and use this dataset as our main benchmark.

5.3. Metrics

For the evaluation on video object segmentation datasets (CPC and DAVIS2016), we use the region similarity $\mathcal{J}$ and boundary accuracy $\mathcal{F}$. In addition, we measure *average accuracy* of pixel-level mask prediction introduced in [37].

To evaluate the point tracking, we modify the metrics, spatial accuracy (SA) and temporal accuracy (TA), introduced in [32]. These metrics measure the contour tracking accuracy by computing the distance from the ground truth points to their closest points in the predicted contour. Different from the original metrics, we measure the distance between the exact corresponding points as follows:

$$\begin{aligned}
\mathrm{SA}_\tau(\mathbf{P},\mathbf{Q}) &= \lambda \sum_{t=0}^{T-1} \sum_{i=0}^{N-1} ||\mathbf{P}_t^i - \mathbf{Q}_t^i||_2 < \tau, \\
\mathrm{TA}_\tau(\mathbf{P},\mathbf{Q}) &= \lambda \sum_{t=1}^{T-1} \sum_{i=0}^{N-1} ||(\mathbf{P}_t^i - \mathbf{Q}_t^i) \\
&\qquad - (\mathbf{P}_{t-1}^i - \mathbf{Q}_{t-1}^i)||_2 < \tau,
\end{aligned} \tag{8}$$

where $\mathbf{P}_t^i$ and $\mathbf{Q}_t^i$ are the corresponding points with index i each in the prediction and ground-truth sets at time t, and τ and λ denote a relative spatial threshold and an averaging scale factor of $\frac{1}{TN}$.

5.4. Ablation Study

To verify the importance of our proposals, we conduct an ablation study on three main components: local alignment module (LAM), unsupervised loss (UL), and synthetic data (SD). For a baseline, we use our local alignment network only with circular convolution blocks after the global alignment.

We perform the ablation study on PoST and summarize the results in Table 1. For better analysis, spatial accuracy (SA) and temporal accuracy (TA) are measured with multiple threshold settings. Throughout all experiments, when we use the synthetic data, point tracking performance increases dramatically with an absolute gain of more than 10 points on average in terms of $\mathrm{SA}_{.04}$. Under the condition of the absence of point matching loss (without UL and SD), positional encoding and temporal information transfer in LAM does not improve performance due to no matching point supervision (see row 1 and 3). However, this additional information enhances point tracking performance when given point matching supervision by an absolute gain of 3 points on average in terms of $\mathrm{SA}_{.04}$ (see row 2 and 5). Without the point supervision, the cycle consistency of unsupervised loss only increases SA since it helps to recover the point correspondence when tracking is failed (see row 3 and 4). The unsupervised loss, however, further improves the performance by exploiting the synthetic data for the point matching supervision (see row 5 and 6).

LAM	UL	SD	SA$_{.16}$	SA$_{.08}$	SA$_{.04}$	TA$_{.16}$	TA$_{.08}$	TA$_{.04}$
			0.906	0.776	0.615	0.976	0.943	0.846
		✓	0.907	0.820	0.701	0.974	0.943	0.881
✓			0.909	0.774	0.599	0.969	0.924	0.819
✓	✓		0.909	0.808	0.672	0.961	0.920	0.829
✓		✓	0.950	0.865	0.736	**0.977**	0.943	0.884
✓	✓	✓	**0.964**	**0.902**	**0.803**	**0.977**	**0.956**	**0.896**

Table 1: Ablation studies on PoST. Three different components of our framework are validated: local alignment module (LAM), unsupervised learning by temporal cycle consistency (UL) and synthetic data (SD).

Method	SA$_{.16}$	SA$_{.08}$	SA$_{.04}$	TA$_{.16}$	TA$_{.08}$	TA$_{.04}$
CSRT [33]	0.925	0.878	**0.807**	0.973	0.927	0.842
MaskFlownet [50]	0.664	0.497	0.345	0.941	0.831	0.640
STM [35] + CSRT [33]	0.856	0.715	0.550	0.965	0.910	0.815
Roto++ [28]	0.769	0.530	0.366	0.841	0.630	0.403
ROAM [37]	0.871	0.717	0.512	0.965	0.873	0.697
Ours	**0.964**	**0.902**	0.803	**0.977**	**0.956**	**0.896**

Table 2: Comparison with other methods on PoST.

	Avg. Acc.	$\mathcal{J}$	$\mathcal{F}$
STM [35]	0.997	0.957	0.982
MaskFlownet [50]	0.948	0.625	0.627
CPC* [32]	**0.998***	**0.963***	**0.997***
Roto++ (1 kf) [28]	0.976	0.640	0.527
Roto++ (2 kf) [28]	0.989	0.840	0.810
ROAM [37]	0.995	0.951	-
ROAM† [37]	0.995†	0.859†	0.893†
Ours	0.997	0.948	0.995

* partial evaluation.
† reproduced with default setting.

Table 3: Quantitative Results on CPC.

5.5. Comparison with Other Methods

PoST. We found only few works [28, 37] that perform a point tracking mechanism in their framework. For completeness, we also compare our method against alternative methods such as optical flow-based tracking and patch tracking combined with a video object segmentation technique. In the case of optical flow, the given point set is tracked through all frames by propagating each point following the flow map from a current state-of-the-art optical flow method [50]. Patch tracking can also be used to track each point by extracting patches centered on given points. For the patch tracking method, we can additionally guide the result to stick to the object boundary using object masks from the-state-of-the-art video object segmentation method, STM [35]. We use CSRT [33] for the patch tracking here.

	Avg. Acc.	$\mathcal{J}$	$\mathcal{F}$
STM [35]	**0.992**	**0.887**	**0.899**
MaskFlownet [50]	0.873	0.300	0.289
Roto++ (1 kf) [28]	0.908	0.217	0.195
Roto++ (2 kf) [28]	0.926	0.329	0.290
ROAM [37]	0.952	0.583	-
ROAM† [37]	0.929†	0.378†	0.335†
Ours	0.971	0.642	0.637

† reproduced with default setting.

Table 4: Quantitative Results on DAVIS2016 Val.

Results of each method on PoST is reported in Table 2. As the results show, our model yields superior performance compared with existing competitors in point set tracking in terms of SA and TA with entire thresholds. Note that patch tracking shows impressive results in terms of SA, but the outlines of the object were not preserved well in this setting.

Video Object Segmentation Datasets. Although video object segmentation is not the main goal in this paper, our method can represent object contour using a polygonal point set. For evaluation, we report performance on video object segmentation dataset, CPC [32] and DAVIS2016 [9].

Quantitative results on CPC are summarized in Table 3. In CPC, even though the performance is saturated due to the rigidity and the static motion of the targets in the dataset, our method shows a comparable accuracy with other methods. Note that STM [35] aims to derive a region-based object mask thus does not track a point set. We report the performance of CPC [32] only for a subset of the dataset as reported in the original paper, because their codes are not available. In Roto++, we also test with two ground-truth keyframes of the first and last frames because interpolation between keyframes plays an important role in the mechanism. The performance of ROAM [37] is reported in two cases: the number reported in the original paper and our reproduced results with an official code in the default setting.

For DAVIS2016 validation set, the performance of various methods is shown in Table 4. Since the dataset targets non-rigid object motion and occlusions, contour-based object segmentation methods inherently have many challenges on this dataset. Despite the limitation, our method outperforms other point set tracking approaches with significant gaps in all metrics.

Runtime Performance. Table 5 shows runtime performance on DAVIS2016. We use GeForce TITAN X as in [37]. Our method is the fastest one among other point set tracking approaches.

5.6. Qualitative Results

Figure 4 shows some qualitative results of our method. Given an initial polygonal point set of the target object, our model propagates the set over frames. Each corresponding

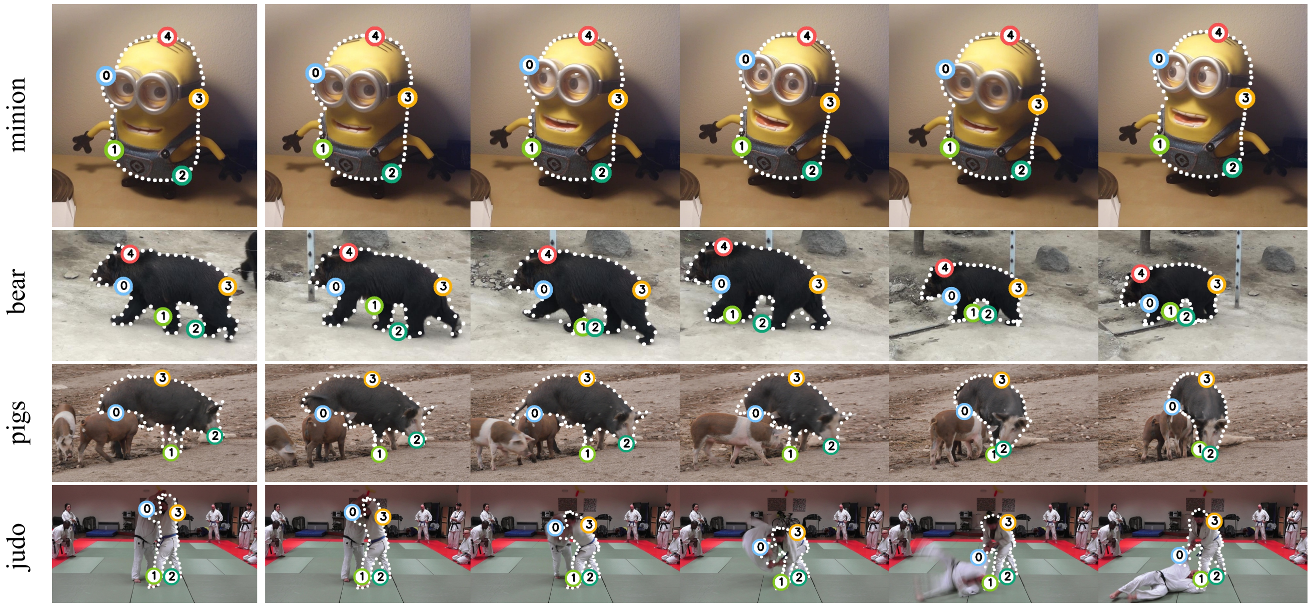
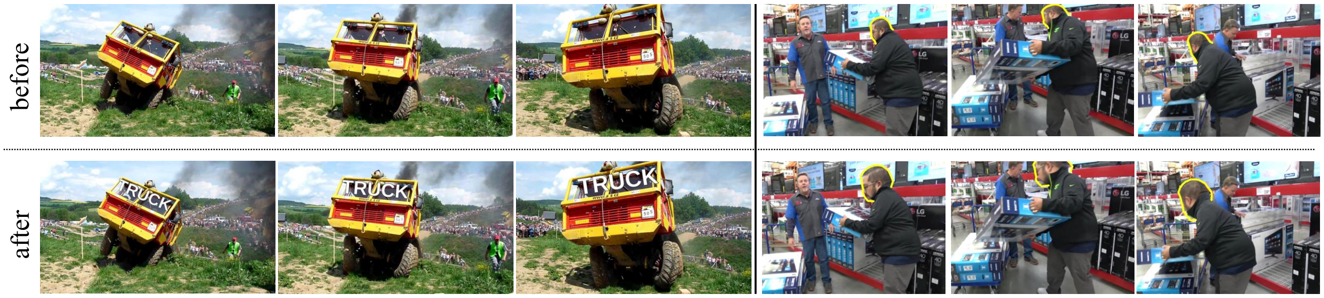


Figure 4: Qualitative results on various datasets. The images in the first column are the initial frames of each clip. Points in a predicted point set are colored as white and several identical points with the same index are visualized in the same color. For better analysis, we select samples in different categories and scenarios.



(a) (b)

Figure 5: Applications of our point set tracking method. (a) Text is mapped by motion tracking in front of the truck. (b) The head of a man is exaggerated by part distortion.

	Roto++ [28]	ROAM [37]	STM+CSRT [35, 33]	**Ours**
Time (ms) / Frame	118*	5639*	2158	84

∗ reported in [37].

Table 5: Runtime Performances on DAVIS2016 Val.

point is marked with unique indices in different colors. Our model yields successful results in terms of both regional segmentation and point set propagation for the target object through all sequences.

5.7. Applications

Figure 5 showcases some applications of the point set tracking. We can apply text mapping by motion tracking for a rigid object in Figure 5 (a) and part-level deformation for non-rigid object in Figure 5 (b). Our method makes it easy to apply these effects, whereas patch tracking and region-based segmentation require additional information.

6. Conclusion

In this paper, we proposed a learning-based method for polygonal point set tracking. We designed global and local alignment networks for polygonal point set tracking. To train the network, we introduced our learning scheme using synthetic data and the unsupervised cycle-consistency loss. We demonstrated that our model successfully propagates a polygonal point set over time with accurate point-wise correspondences even without any fully annotated ground-truth data. We achieved state-of-the-art performance on multiple benchmarks and showcased interesting applications.

Acknowledgements This work was supported in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) Grant funded by the Korean Government (MSIT), Artificial Intelligence Graduate School Program, Yonsei University, under Grant 2020-0-01361.

References

[1] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Efficient interactive annotation of segmentation datasets with polygon-rnn++. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[2] Aseem Agarwala, Aaron Hertzmann, David H Salesin, and Steven M Seitz. Keyframe-based tracking for rotoscoping and animation. *ACM Transactions on Graphics (ToG)*, 2004.

[3] Xue Bai, Jue Wang, David Simons, and Guillermo Sapiro. Video snapcut: robust video object cutout using localized classifiers. *ACM Transactions on Graphics (ToG)*, 2009.

[4] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[5] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[6] Andrew Blake and Michael Isard. *Active contours: the application of techniques from graphics, vision, control theory and statistics to visual tracking of shapes in motion*. 2012.

[7] David S Bolme, J Ross Beveridge, Bruce A Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[8] Sergi Caelles, Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. One-shot video object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[9] Sergi Caelles, Alberto Montes, Kevis-Kokitsi Maninis, Yuhua Chen, Luc Van Gool, Federico Perazzi, and Jordi Pont-Tuset. The 2018 davis challenge on video object segmentation. *arXiv preprint arXiv:1803.00557*, 2018.

[10] Lluis Castrejon, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler. Annotating object instances with a polygon-rnn. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[11] Yunqiang Chen, Yong Rui, and Thomas S Huang. Multicue hmm-ukf for real-time contour tracking. *IEEE transactions on pattern analysis and machine intelligence*, 2006.

[12] Martin Danelljan, Luc Van Gool, and Radu Timofte. Probabilistic regression for visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[13] Martin Danelljan, Andreas Robinson, Fahad Shahbaz Khan, and Michael Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[14] Martin Danelljan, Fahad Shahbaz Khan, Michael Felsberg, and Joost Van de Weijer. Adaptive color attributes for real-time visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[15] Heng Fan and Haibin Ling. Siamese cascaded region proposal networks for real-time visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[16] Qingnan Fan, Fan Zhong, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. Jumpcut: non-successive mask transfer and interpolation for video cutout. *ACM Transactions on Graphics (ToG)*, 2015.

[17] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997.

[20] Michael Isard and Andrew Blake. Contour tracking by stochastic propagation of conditional density. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 1996.

[21] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, 2015.

[22] Varun Jampani, Raghudeep Gadde, and Peter V Gehler. Video propagation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[23] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International journal of computer vision*, 1988.

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[25] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv preprint arXiv:1811.00982*, 2018.

[26] Sungho Lee, Seoung Wug Oh, DaeYeun Won, and Seon Joo Kim. Copy-and-paste networks for deep video inpainting. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[27] Fuxin Li, Taeyoung Kim, Ahmad Humayun, David Tsai, and James M Rehg. Video segmentation by tracking many figure-ground segments. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2192–2199, 2013.

[28] Wenbin Li, Fabio Viola, Jonathan Starck, Gabriel J Brostow, and Neill DF Campbell. Roto++ accelerating professional rotoscoping using shape manifolds. *ACM Transactions on Graphics (TOG)*, 2016.

[29] Justin Liang, Namdar Homayounfar, Wei-Chiu Ma, Yuwen Xiong, Rui Hu, and Raquel Urtasun. Polytransform: Deep polygon transformer for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[30] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE*

Conference on Computer Vision and Pattern Recognition, 2017.

[31] Huan Ling, Jun Gao, Amlan Kar, Wenzheng Chen, and Sanja Fidler. Fast interactive object annotation with curve-gcn. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[32] Yao Lu, Xue Bai, Linda Shapiro, and Jue Wang. Coherent parametric contours for interactive video object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[33] Alan Lukezic, Tomas Vojir, Luka Cehovin Zajc, Jiri Matas, and Matej Kristan. Discriminative correlation filter with channel and spatial reliability. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[34] Jiaxu Miao, Yunchao Wei, and Yi Yang. Memory aggregation networks for efficient interactive video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[35] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[36] Sida Peng, Wen Jiang, Huaijin Pi, Xiuli Li, Hujun Bao, and Xiaowei Zhou. Deep snake for real-time instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[37] Juan-Manuel Perez-Rua, Ondrej Miksik, Tomas Crivelli, Patrick Bouthemy, Philip HS Torr, and Patrick Perez. Roam: A rich object appearance model with application to rotoscoping. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[38] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 2015.

[39] Shivam Saboo, Frederic Lefebvre, and Vincent Demoulin. Deep learning and interactivity for video rotoscoping. In *2020 IEEE International Conference on Image Processing (ICIP)*, 2020.

[40] Scott Schaefer, Travis McPhail, and Joe Warren. Image deformation using moving least squares. In *ACM SIGGRAPH 2006 Papers*. 2006.

[41] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[42] Satoshi Suzuki et al. Topological structural analysis of digitized binary images by border following. *Computer vision, graphics, and image processing*, 1985.

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, 2017.

[44] Paul Voigtlaender and Bastian Leibe. Online adaptation of convolutional neural networks for video object segmentation. *arXiv preprint arXiv:1706.09364*, 2017.

[45] Ning Wang, Yibing Song, Chao Ma, Wengang Zhou, Wei Liu, and Houqiang Li. Unsupervised deep tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[46] Xiaolong Wang, Allan Jabri, and Alexei A Efros. Learning correspondence from the cycle-consistency of time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[47] Seoung Wug Oh, Joon-Young Lee, Kalyan Sunkavalli, and Seon Joo Kim. Fast video object segmentation by reference-guided mask propagation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[48] Ning Xu, Linjie Yang, Yuchen Fan, Jianchao Yang, Dingcheng Yue, Yuchen Liang, Brian Price, Scott Cohen, and Thomas Huang. Youtube-vos: Sequence-to-sequence video object segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[49] Alper Yilmaz, Xin Li, and Mubarak Shah. Object contour tracking using level sets. In *Asian conference on computer vision*, 2004.

[50] Shengyu Zhao, Yilun Sheng, Yue Dong, Eric I Chang, Yan Xu, et al. Maskflownet: Asymmetric feature matching with learnable occlusion mask. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.