Space-Time Memory Networks for Video Object Segmentation with User Guidance

Seoung Wug Oh, *Student Member, IEEE,* Joon-Young Lee, *Member, IEEE,*
Ning Xu, *Member, IEEE,* and Seon Joo Kim, *Member, IEEE*

Abstract—We propose a novel and unified solution for user-guided video object segmentation tasks. In this work, we consider two scenarios of user-guided segmentation: *semi-supervised* and *interactive* segmentation. Due to the nature of the problem, available cues – video frame(s) with object masks (or scribbles) – become richer with the intermediate predictions (or additional user inputs). However, the existing methods make it impossible to fully exploit this rich source of information. We resolve the issue by leveraging memory networks and learning to read relevant information from all available sources. In the semi-supervised scenario, the previous frames with object masks form an external memory, and the current frame as the query is segmented using the information in the memory. Similarly, to work with user interactions, the frames that are given user inputs form the memory that guides segmentation. Internally, the query and the memory are densely matched in the feature space, covering all the space-time pixel locations in a feed-forward fashion. The abundant use of the guidance information allows us to better handle challenges such as appearance changes and occlusions. We validate our method on the latest benchmark sets and achieve state-of-the-art performance along with a fast runtime.

Index Terms—Video Object Segmentation, User-guided Video Object Segmentation, Semi-supervised Video Object Segmentation, Interactive Video Object Segmentation, Memory Networks.

1 Introduction

Video object segmentation is a task that involves the separation of the foreground and the background pixels in all the frames of a given video. It is an essential step for many video editing tasks, which is receiving more attention as videos have become the most popular form of shared media content. It is a very challenging task as the appearance of the target object can change drastically over time and also due to occlusions and drifts. Despite many previous attempts, it still requires a significant amount of manual processing to achieve desirable results.

Video object segmentation usually begins with a user guidance to specify the target object. In this work, we consider two scenarios of user-guided segmentation: *semi-supervised* and *interactive* segmentation. In the semi-supervised setting, which is considered to be the standard method, the complete mask is given for the first frame and algorithms predict the object masks in the remaining frames using the first frame as the only source of evidence. Given an initial mask, the rest of the procedure is automatic thus easy-to-use, and this simplicity is also useful for comparing algorithms for research. However, from a practical viewpoint, there are researchers who hold the opinion that the semi-supervised workflow is not suitable for the actual use. The issue is that it is still difficult to get a complete mask even for one frame. In addition to this drawback, no error correction can be made.

- *S. W. Oh and S. J. Kim (Corresponding Author) are with Yonsei University, Seoul, Korea.*
  E-mail: sw.oh@yonsei.ac.kr and seonjookim@yonsei.ac.kr
- *J.-Y. Lee and N. Xu are with Adobe Research, San Jose, CA, USA.*
  E-mail: jolee@adobe.com and nxu@adobe.com

Manuscript received December, 2019; revised March 2020.

Alternatively, the interactive video object segmentation that involves human intervention in the loop can be considered. A round-based workflow, recently proposed in [1], seeks video-level segmentation for efficiency rather than frame-by-frame processing. In this workflow, a user repeatedly selects a frame and provides hints for segmentation in a user-friendly form (scribbles) An algorithm then processes the entire video to refine the current segmentation according to the new user inputs.

Both scenarios share a similar property, which means that available cues – video frame(s) with object masks or scribbles – become richer as the segmentation process progresses. Specifically, more frames that carry information for the segmentation become available by the intermediate mask predictions (in the semi-supervised scenario) or additional user inputs (in the interactive scenario). This observation is directly linked to the essential question for the learning-based approaches: from which frame(s) should the model learn the cues?

For the semi-supervised scenario, most of the previous learning-based methods seek clues only from a fixed number of sources, making it hard to fully exploit the rich source of information (Fig. 1). In some algorithms, the features are extracted and propagated from the previous frame (Fig. 1 (a)) [2], [3]. The main strength of this approach is that it can deal with changes in appearance better, while sacrificing robustness against occlusions and error drifts. Another direction for the deep learning-based approach is to use the first frame as a reference and independently detect the target object at each frame (Fig. 1 (b)) [4], [5], [6]. The pros and cons of this approach are the exact opposite of those of the previous approach. Some recent methods take a hybrid approach that uses both the first frame and the

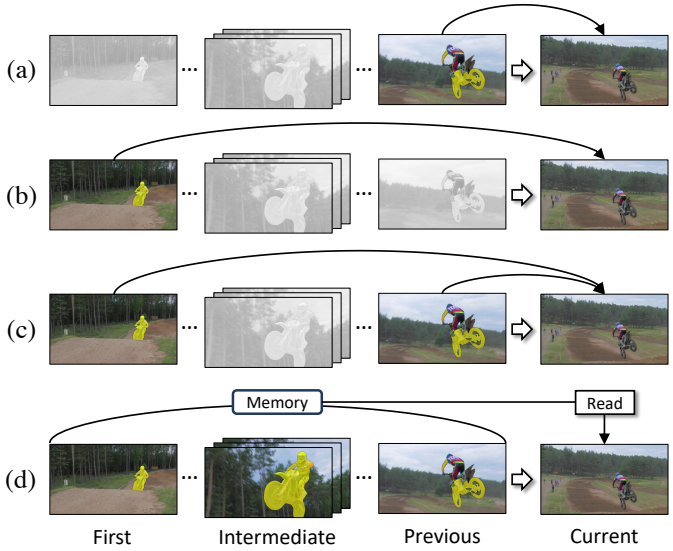


Fig. 1: Previous DNN-based algorithms extract features in different frames for semi-supervised video object segmentation (a-c). We propose an efficient algorithm that exploits multiple frames in the given video for more accurate segmentation (d).

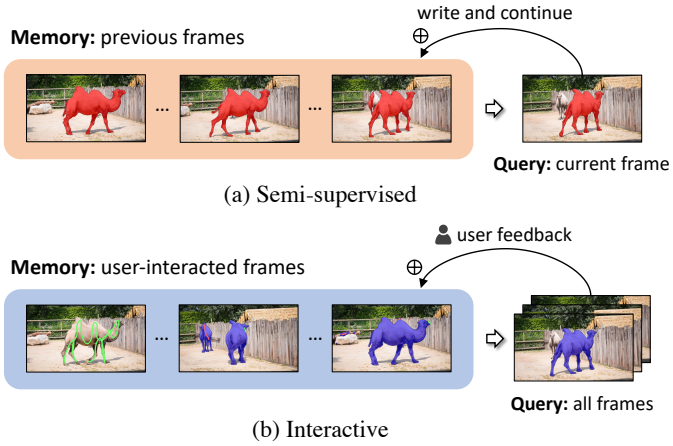


Fig. 2: The proposed memory networks work with two video object segmentation scenarios. In the semi-supervised scenario, the previous frames with the object mask are used as memory (a). In the interactive scenario, frames given user interactions are used as memory (b).

previous frame to take advantage of the two approaches (Fig. 1 (c)) [7], [8], [9].

As using two frames has been shown to be beneficial, a natural extension is to use more frames for the segmentation task. The question is how to design an efficient deep neural network (DNN) architecture to realize the idea. In the conference version of this paper [10], we proposed a novel DNN system for the semi-supervised video object segmentation based on the memory network [11], [12], [13]. Our memory network can memorize two or more frames and read the memory by purpose. By exploiting rich reference information, our approach can deal with appearance changes, occlusions, and drifts much better than the previous methods.

In this paper, we show that the proposed memory network is suitable not only for the semi-supervised segmentation, but also for the interactive task. In this case, the frames that are given user inputs are written onto the memory. Previously, a few methods were presented for the round-based interactive video object segmentation. In [1] and [14], online learning approaches that fine-tune the backbone networks using user scribbles have been proposed. However, online learning is computationally too expensive to be used within interactive tools, thus this limits practical uses. Oh et al. recently proposed joint learning of interactive object segmentation and mask propagation [15]. While this method shows better accuracy and a faster runtime than online learning methods, it still suffers from drifting issues oriented from the repeated mask propagation. On the other hand, our method performs memory-based object detection, therefore, it is free from propagation-driven problems and no online training is required.

Fig. 2 shows applications of our memory network for two video object segmentation scenarios. In the semi-supervised scenario, video frames are sequentially processed using the previous frames with masks as memory and the current frame as query. The newly obtained mask for the current frame is written into the memory to help predict the next frame. In the interactive scenario, all the video frames are segmented in parallel as query using frames with user inputs as memory. Once the user draws new scribbles for feedback, the information is added to the memory and the segmentation is updated accordingly.

Commonly for both applications, the spatio-temporal attention mechanism is exploited. Specifically, for each pixel in the query frame, attention on every pixel in the memory frames is computed, which makes it possible to determine whether the query pixel belongs to a foreground object or not. New information can be easily added by putting it onto the memory, and there is no restriction on the size of the memory as it is stored externally unlike network parameters. This memory update greatly helps us to address challenges such as appearance changes and occlusions at no additional cost. Aside from the benefits of saving more memory frames, our network includes a non-local spatial pixel matching mechanism that is suitable for pixel-level estimation problems. Note that the proposed network structure is compatible with both scenarios with a minor modification[1].

For both tasks, we validate our method on the latest benchmarks' validation and challenge sets. For the semi-supervised video object segmentation task, we use YouTube-VOS [16] and DAVIS-2016/2017 [17], [18] validation sets for evaluation. Our method achieve state-of-the-art performance outperforming all the existing methods by a large margin in terms of both speed and accuracy. For the interactive video object segmentation task, our method shows the best performance on the DAVIS-2017 validation set, and won first place in the DAVIS interactive challenge 2019, surpassing other participants with cutting-edge techniques [19].

1. The only difference is that our network for interactive segmentation can additionally take scribbles for input.

This journal paper extends our earlier work [10] by introducing a new application of the space-time memory networks to the interactive video object segmentation. Accordingly, a new training procedure, inference scheme, analysis, and further results are presented.

2 Related Work

2.1 Semi-supervised Video Object Segmentation

2.1.1 Previous Methods

Propagation-based methods [2], [3], [20], [21] learn an object mask propagator, a deep network that refines misaligned mask toward the target object (Fig. 1(a)). To make the network object-specific, online training data are generated from the first frame by deforming the object mask [3] or synthesizing images [20] for fine-tuning. Li et al. [21] integrate a re-identification module into the system to retrieve missing objects caused by drifts.

Detection-based methods [4], [5], [6], [22], [23], [24] work by learning an object detector using the object appearance on the first frame (Fig. 1(b)). In [4], [22], an object-specific detector learned by fine-tuning the deep networks at the test time is used to segment out the target object. In [6], [24], to avoid the online learning, pixels are embedded into feature space and classified by matching them to templates.

Some recent methods involved taking a hybrid approach [7], [8]. These methods are designed to take advantage of both detection and propagation approaches (Fig. 1(c)). In [7], [8], networks that exploit both the visual guidance from the first frame and the spatial priors from the previous frame were proposed. Yang et al. [8] proposed a meta modulator network that manipulates the intermediate layers given reference visual and spatial priors. Oh et al. [7] used a Siamese two-stream network that propagates the object mask while referring to the visual guidance from the first frame.

Furthermore, some methods tried to exploit all of the previous information [25], [26]. Xu et al. [25] proposed to learn a sequence-to-sequence network that learns the long-term information in videos. Voigtlaender and Leibe [26] employed the idea of online adaptation and continuously update the detector using the intermediate outputs. Our method can also make use of all the previous information. Different from the previous approaches, by maintaining the memory information externally, neither the network parameters nor the state of the recurrent network are needed to be updated. Our efficient memory mechanism brings not only a significant increase in speed but also results in state-of-the-art accuracy.

2.1.2 Online and Offline Learning

Many of the aforementioned methods fine-tune deep network models on the initial object mask in the first frame in order to remember the appearance of the target object during the test time [2], [3], [3], [4], [20], [21], [26]. While the online learning improves accuracy, it is computationally expensive, thus limiting its practical use. Offline learning methods attempted to bypass the online learning while retaining the same level of accuracy [6], [7], [8], [9], [24], [27], [28]. A common idea is to design deep networks capable of object-agnostic segmentation at the test time, based on the guidance information.

Our framework belongs to the offline learning method, and adopts a memory mechanism to make use of the guidance information. Our method maintains intermediate outputs in the external memory as the guidance, and adaptively selects necessary information in runtime. This flexible use of the guidance information enables our method to outperform the aforementioned methods by a large margin. Our memory network is also fast, as the memory reading is done as a part of the network forward pass, thus no online learning is required.

2.2 Interactive Video Object Segmentation

2.2.1 Frame-by-frame Methods

Traditional interactive video object segmentation methods usually follow the procedure of the rotoscoping, where an algorithm processes a video frame-by-frame given user annotations in various types (e.g., scribbles, bounding boxes, or masks) [29], [30]. In rotoscoping, the user refines and verifies the object mask at every frame. There is a vast volume of previous works on this task [29], [30], [31], [32], [33], [34], [35], [36]. In [33], [36], local classifiers, that integrate multiple local features such as color and edge, are proposed for propagating masks to the next frame. In [31], [32], [35], a spatio-temporal graph of a video is established and solved by minimizing energy functions. Some methods solve the segmentation task by tracking [30], [34].

2.2.2 Round-based Methods

Recently, Caelles et al. [1] introduced new workflow for the video object cutout that focuses on minimizing the user's effort. In this scenario, an algorithm handles an entire video in a batch process after user annotations on a few selected frames are given. Additional user inputs on a frame affect the segmentation results for all the video frames. Caelles et al. [1] proposed modifying their previous one-shot video object segmentation model [4] to make it applicable for this interactive scenario. Benard and Gygli [14] combined an interactive image segmentation method [37] with one-shot video object segmentation [4]. First, an object is selected by the given initial strokes or clicks, then a one-shot object detector is learned to segment the object in other frames. Oh et al. [15] proposed to jointly learn two deep networks where one operates interactively with the user and the other propagates the object mask. This approach outperforms earlier methods based on one-shot learning by a large margin in terms of speed and accuracy. However, the accuracy rapidly gets saturated with a small number of interactions as the results are continually overridden by propagated information from new interactions that might not be as good as the current state.

Our interactive system uses round-based interaction. In contrast to the previous works, our system memorizes all the user-annotated frames and uses these memories to detect the target object in a feed-forward fashion. This new mechanism ensures that our system is robust in handling the challenges that arise during propagation and also brings a substantial increase in speed when compared to one-shot learning approaches.

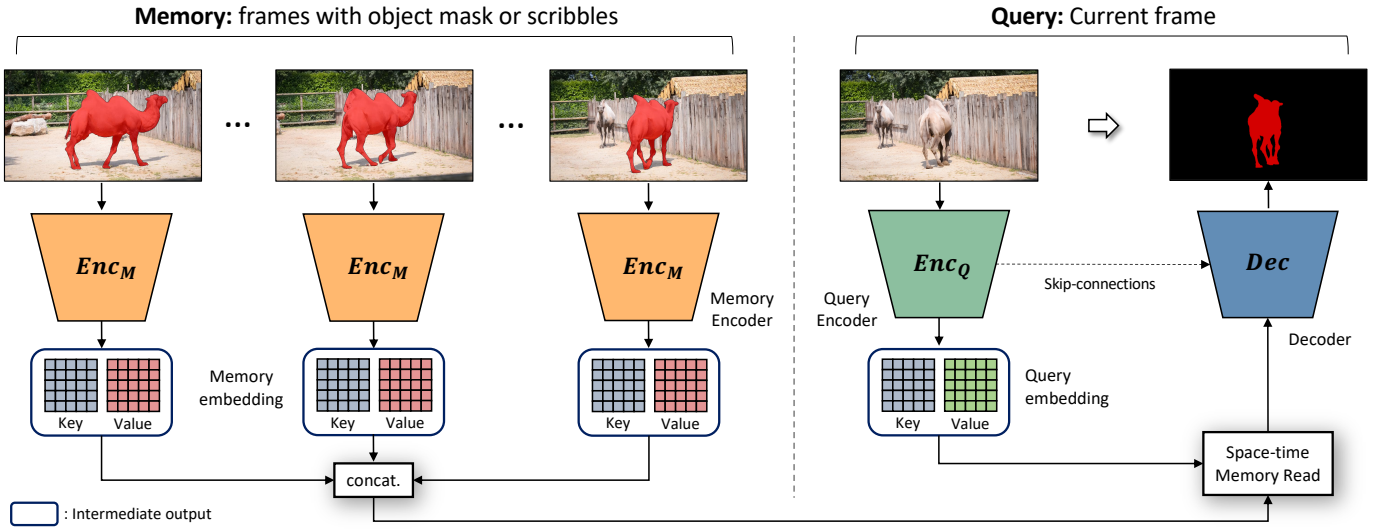


Fig. 3: Overview of our framework. Our network consists of two encoders each for the memory and the query frame, a space-time memory read block, and a decoder. The memory encoder (Enc_M) takes an RGB frame and the object mask. The object mask is represented as a probability map (the softmax output is used for the estimated object masks). The query encoder (Enc_Q) takes the query image as input.

2.3 Memory Networks

Memory networks are the neural networks that have external memory where information can be written and read according to their purpose. Memory networks that can be trained end-to-end were first proposed in the NLP research for the purpose of the document Q&A [11], [12], [13]. Commonly in those approaches, memorable information is separately embedded into key (input) and value (output) feature vectors. Keys are used to address relevant memories whose corresponding values are returned. Recently, the memory networks have been applied to some vision problems such as personalized image captioning [38], visual tracking [39], movie understanding [40], summarization [41], and video inpainting [42].

Inspired by the concept of the memory networks, we extend the idea to make it suitable for our task, user-guided video object segmentation. Video frames with user guidance (e.g., masks and scribbles) form memories, and a frame that is to be segmented acts as the query. The memory is dynamically updated with newly predicted masks and additional user inputs and it greatly helps us to address challenges such as appearance changes, occlusions, and error accumulations without the online learning.

Our goal is to have pixel-wise predictions based on a set of annotated frames as memory. Thus each pixel in the query frame needs to access information in the memory frames at different space-time locations. To this end, we design our memory as 4D tensors in order to contain pixel-level information and propose the space-time memory read operation to localize and read relevant information from the 4D memory[2]. Conceptually, our memory reading can be considered to be a spatio-temporal attention algorithm because we are computing *when-and-where* to attend in the memory to determine whether a query pixel belongs to a foreground object or not.

2. 4D memory tensors' shape is (time×height×width×channels).

3 Space-Time Memory Networks (STM)

In our framework, we consider videos frames with annotations as the *memory* frames and a video frame for which we want to get an object mask as the *query* frame. Various types of annotation can be used, and we test our system with binary masks, probability maps (network outputs), hand-drawn scribbles, and combinations of them. The overview of our framework is shown in Fig. 3.

Both the memory and the query frames are first encoded into pairs of key and value maps through the dedicated deep encoders. Note that the query encoder takes only an image as the input, while the memory encoder takes both an image and the corresponding annotation(s). Each encoder outputs **key** and **value** maps. The **key** is used for addressing memory locations. Specifically, similarities between **key** features of the query and the memory frames are computed to determine when-and-where to retrieve relevant memory **values** from. Therefore, the **key** is learned to encode visual semantics for matching object parts while being robust to appearance variations. On the other hand, the **value** stores detailed information for producing the mask estimation (e.g., the target object and object boundaries). The **values** from the query and the memory contain information for somewhat different purposes. Specifically, the ***value*** *for the query frame* is learned in order to store detailed appearance information for us to decode accurate object masks. The ***value*** *for the memory frames* learns to encode both the visual semantics and the mask information about whether each feature pixel belongs to the foreground or the background.

Subsequently, the keys and values go through our space-time memory read block. Every pixel on the key feature maps of the query and the memory frames is densely

matched over the spatio-temporal space of the video. Relative matching scores are then used to address the value feature map of the memory frame, and the corresponding values are combined to return outputs. Finally, the decoder takes the output of the read block and reconstructs the mask for the query frame.

3.1 Key and Value Embedding

3.1.1 Query Encoder

The query encoder takes the query frame as the input. The encoder outputs two feature maps – key and value – through two parallel convolutional layers attached to the backbone network. These convolutional layers serve as bottleneck layers that reduce the feature channel size of the backbone network output (to 1/8 for the key and 1/2 for the value). No non-linearity is applied. The output of the query embedding is a pair of 3D key and value feature maps ($\mathbf{k}^Q \in \mathbb{R}^{H\times W\times C/8}, \mathbf{v}^Q \in \mathbb{R}^{H\times W\times C/2}$), where H is the height, W is the width, and C is the feature dimension of the backbone network output feature map.

3.1.2 Memory Encoder

The memory encoder has the same structure except for the inputs. The input to the memory encoder consists of an RGB frame and its annotations. The type of annotation depends on the application[3]. Refer to Section 3.6 and 3.7 for the representation of annotations for two different applications. All the inputs are concatenated along the channel dimension before being fed into the memory encoder.

If there is more than one memory frame, each of them is independently embedded into key and value map. Then, the key and value maps from different memory frames are stacked along the temporal dimension. The output memory consists of a pair of 4D key and value feature tensors ($\mathbf{k}^M \in \mathbb{R}^{T\times H\times W\times C/8}, \mathbf{v}^M \in \mathbb{R}^{T\times H\times W\times C/2}$), where T is the number of the memory frames.

We take ResNet50 [43] as the backbone network for both the memory and the query encoder. We use the stage-4 (`res4`) feature map of the ResNet50 as the base feature map for computing the key and value feature maps. For the memory encoder, the first convolution layer is modified to be able to take a tensor with more than three channels by implanting additional filter channels. The network weights are initialized from the ImageNet pre-trained model, except for the newly added filters which are initialized randomly.

3.2 Space-time Memory Read

In the memory read operation, soft weights are first computed by measuring the similarities between all the pixels of the query key map and the memory key map. The similarity matching is performed in a non-local manner by comparing every space-time location in the memory key map with every spatial location in the query key map. Then, the value of the memory is retrieved by a weighted summation with the soft weights and it is concatenated with the query value.

3. For the semi-supervised setting, a mask annotation is used. For the interactive mode, both a mask and user scribbles are given.

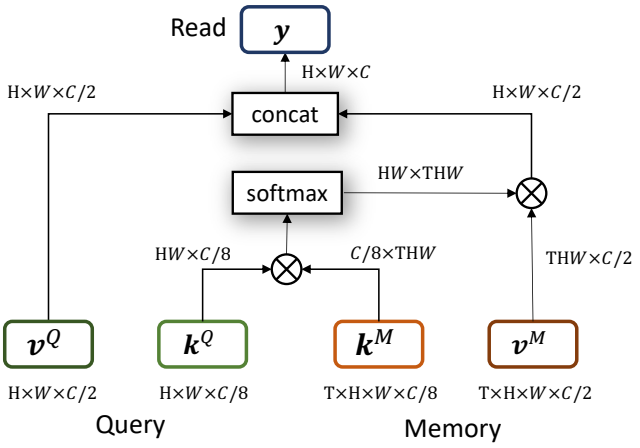


Fig. 4: Detailed implementation of the space-time memory read operation using basic tensor operations as described in Sec. 3.2. ⊗ denotes matrix inner-product.

This memory read operates for every location on the query feature map and can be summarized as follows:

$$\mathbf{y}_i = [\mathbf{v}_i^Q, \frac{1}{Z}\sum_{\forall j} f(\mathbf{k}_i^Q, \mathbf{k}_j^M)\mathbf{v}_j^M], \tag{1}$$

where i and j are the index of the query and the memory location, $Z = \sum_{\forall j} f(\mathbf{k}_i^Q, \mathbf{k}_j^M)$ is the normalizing factor and $[\cdot,\cdot]$ denotes the concatenation. The similarity function f is as follows:

$$f(\mathbf{k}_i^Q, \mathbf{k}_j^M) = \exp(\mathbf{k}_i^Q \circ \mathbf{k}_j^M), \tag{2}$$

where $\circ$ denotes the dot-product.

Our formulation can be seen as an extension of the early formulation of the differential memory networks [11], [12], [13] to 3D spatio-temporal space for video pixel matching. Accordingly, the proposed read operation localizes the space-time location of the memory for retrieval. It is also related to non-local self-attention mechanisms [44], [45] in that it performs non-local matching. However, our formulation is motivated by a different purpose as it is designed to attend to others (memory frames) for the information retrieval, not to itself for the self-attention. As depicted in Fig. 4, our memory read operation can be easily implemented by a combination of basic tensor operations in modern deep learning platforms.

3.3 Decoder

The decoder takes the output of the read operation and reconstructs the query frame's object mask. We employ the refinement module used in [7] as the building block of our decoder. The read output is first compressed to have 256 channels with a convolutional layer and a residual block [46], then a number of refinement modules gradually upscale the compressed feature map by a factor of two at a time. The refinement module at every stage takes both the output of the previous stage and a feature map from the query encoder at the corresponding scale through skip-connections. The output of the last refinement block is used to reconstruct the object mask through the final convolutional layer followed by a softmax operation. Every

convolutional layer in the decoder uses 3×3 filters, producing 256-channel output except for the last one that produces 2-channel output. The decoder estimates the mask in a 1/4 scale of the input image.

3.4 Multi-object Segmentation

The description of our framework is based on having one target object in the video. However, recent benchmarks require a method that can deal with multiple objects [18], [25]. To meet this requirement, we extend our framework with a mask merging operation. We run our model for each object independently and compute mask probability maps for all objects. Then, similar to the soft aggregation operation in [7]. we merge the predicted maps as follow:

$$p_{i,m} = \sigma\big(l(\hat{p}_{i,m})\big) = \frac{\hat{p}_{i,m}/(1-\hat{p}_{i,m})}{\sum_{j=0}^{M} \hat{p}_{i,j}/(1-\hat{p}_{i,j})}, \quad \text{s.t.} \quad \hat{p}_{i,0} = \Pi_{j=1}^{M}(1-\hat{p}_{i,j}), \quad (3)$$

where σ and l represent the softmax and the logit function respectively, $\hat{p}_{i,m}$ is the network output probability of the object m at the pixel location i, m=0 indicates the background, and M is the total number of objects.

In [7], the mask merging is performed only during the testing as a post-processing step. On the other hand, we construct the operation as a differential network layer and apply it during both the training and the testing. Furthermore, if multiple objects are present, we provide additional information to the memory encoder about other objects. Specifically, probability masks for all other objects, which are computed as $o_{i,m} = \sum_{j \neq m}^{M} p_{i,j}$ are additionally given to the network.

3.5 Training Data

To overcome the scarcity of training video data, we adopt two-stage training with different data sources. Our network is first pre-trained on a simulation dataset generated from static image data. Then, it is further fine-tuned for real-world videos through the main training.

3.5.1 Pre-training on Images

One advantage of our framework is that it does not require long training videos. This is because the method learns the semantic spatio-temporal matching between distant pixels without any assumption on temporal smoothness. This means that we can train our network with only a few frames[4]. This enables us to simulate training videos using image datasets. Some previous works [3], [7] trained their networks using static images and we take a similar strategy. A synthetic video clip that consists of 3 frames is generated by applying random affine transforms[5] to a static image with different parameters. We leverage the image datasets annotated with object masks (salient object detection – [47], [48], semantic instance segmentation – [49], [50], [51]) to pre-train our network. By doing so, we can expect our model to be robust against a wide variety of object appearances and categories.

4. Minimum 2; one as the memory frame and the other as the query.
5. We use rotation, sheering, zooming, translation, and cropping.

3.5.2 Main Training on Videos

After the pre-training, we use real video data for the main training. In this step, training videos from YouTube-VOS [25] and DAVIS-2017 [18] are used, depending on the target evaluation benchmark. To make a training sample, we sample 3 temporally ordered frames from a training video. To learn the appearance change over a long time, we randomly skip frames during the sampling. The maximum number of frames to be skipped is gradually increased from 0 to 25 during the training as in curriculum learning [52].

While the same data is used for training, the training strategy differs depending on the application as the testing scheme is different.

3.6 Semi-supervised Video Object Segmentation

In the semi-supervised setting, the ground truth mask of the target object is given in the first frame and the goal is to estimate the object masks in all the other frames. In our system, video frames are sequentially processed starting from the second frame using the ground truth annotation given in the first frame. During the video processing, we consider the previous frames with object masks (either given at the first frame or estimated at other frames) as the memory frames and the current frame without the object mask as the query frame. Newly obtained masks for the previous frames are saved as memory, and the accumulation of memories greatly helps us to address challenges such as appearance changes and occlusions with no cost.

3.6.1 Training with Growing Memory

Similar to the testing scenario, the size of the memory is dynamically growing with the network's previous outputs during the training. Combining memories is implemented as a tensor concatenation, thus the whole model is end-to-end trainable. As the system moves forward frame-by-frame, the computed segmentation output at the previous step is added to the memory for the next frame. In the example of the 3-frame training clip, the second frame's mask is estimated by using the memory of the first frame and its ground truth mask. Then, for the third frame, the memory of both the first and second frame are used. Losses computed from all the predictions are back-propagated for training. The object mask is represented as a single channel probability map between 0 and 1 (the softmax output is used for estimated masks). The raw network output without thresholding, which is a probability map of being a foreground object, is directly used for the memory embedding to model the uncertainty of the estimation.

3.6.2 Memory Management

Writing all the previous frames on to the memory may raise practical issues such as GPU memory overflow and slow running speed. Instead, we select frames to be put onto the memory by a simple rule. The first and the previous frame with object masks are the most important pieces of reference information [3], [7], [8]. The first frame always provides reliable information as it comes with the ground truth mask. The previous frame is similar in appearance to the current frame and thus we can expect accurate pixel matching and

mask propagation. Therefore, we put these two frames into the memory by default.

For the intermediate frames, we simply save a new memory frame every N frame. N is a hyperparameter that controls the trade-off between speed and accuracy, and we use $N = 5$ unless indicated otherwise. It is noteworthy that our framework achieves the effect of online learning and online adaptation without additional training. The advantages of online model updating are easily accomplished by putting the previous frames into the memory without updating the model parameters. Thus, our method runs considerably faster than most of the previous methods while achieving state-of-the-art accuracy.

3.7 Interactive Video Object Segmentation

For the interactive video object segmentation, we follow the workflow proposed by Caelles et al. [1]. In this workflow, the user initially selects a frame and draws scribbles to guide segmentation. Then, an algorithm computes the segmentation maps for all the video frames in a batch process. We call the process of user annotation and segmentation a *round*, and the user may go through multiple rounds to enhance the quality.

We consider video frames given with user annotations as memory frames. There are two types of scribbles: initial and feedback scribbles. For the first round, initial scribbles that indicate the target object are drawn on the object region. In the following rounds, feedback scribbles are drawn on false pixels to guide corrections for refinement. In these cases, all the available information, current segmentation and scribbles, is used to get a memory embedding[6]. In every round, the selected frame and the corresponding user guidance are written onto the memory. Then, all the video frames as queries are segmented using the updated memory in parallel.

3.7.1 Synthetic User Inputs

We train our system with simulated user inputs as in [1], [15]. Similar to [1], we use a set of morphological operations to synthesize scribbles. First, regions to draw scribbles are determined. Then, the skeletonization (or thinning) algorithm [53] is applied to generate scribbles. For initial scribbles, positive scribbles are drawn on the object region and there are no negative scribbles. For feedback scribbles. both positive and negative scribbles are generated from false negative and positive regions, respectively. Random affine transforms are applied to the scribbles to simulate randomness of human activities. We empirically validated that our model trained with simulated user inputs, despite its simplicity, works well with real users as shown in our demo video.

3.7.2 Multi-round Training

Inspired by [15], we employ a multi-round training scheme. In other words, we make our training loop consist of multiple rounds of user interactions. This training scheme is effective in improving the performance of interactive models by reducing the gap between training and a real testing scenario. To be specific, our training loop runs over two rounds with a 3-frame training video clip. In the first round, initial positive scribbles are drawn on the target object region, then all the video frames (including the first frame) are segmented as queries using the memory formed by the first frame. In the next round, feedback scribbles are drawn on the false regions of the third frame. Then, the entire video is segmented again using the updated memory assembled by both the first and third frame. All the outputs are evaluated to compute losses.

6. If there is no current segmentation (the first interaction), a zero-valued mask is inputted.

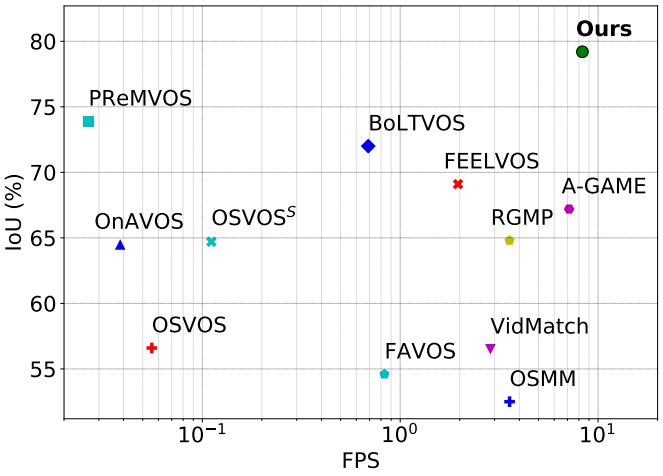


Fig. 5: Accuracy versus speed comparison for semi-supervised video object segmentation on the DAVIS-2017 validation. The FPS axis is in the log scale.

3.8 Implementation Details

For all the experiments, we use randomly cropped 384×384 patches for training. We set the mini-batch size to 4 and disabled all the batch normalization layers. We minimize the cross-entropy loss using the Adam optimizer [54]. Pre-training takes about four days and main training takes about three days using four NVIDIA GeForce 1080 Ti GPUs. More training details can be found in the supplementary material.

4 Evaluation

4.1 Semi-supervised Video Object Segmentation

We evaluate our model on YouTube-VOS [16] and DAVIS [17], [18] benchmarks. We prepared two models trained on each benchmark's training set. For the evaluation on YouTube-VOS, we used 3471 training videos following the official split [16]. For DAVIS, we used 60 videos from the DAVIS-2017 train set. Both DAVIS 2016 and 2017 are evaluated using a single model trained on DAVIS-2017 for a fair comparison with the previous works [7], [8]. In addition, we provide the results for the DAVIS with our model trained with additional training data from YouTube-VOS. Note that we use the network output directly without post-processing to evaluate our method. We measure the region similarity $\mathcal{J}$ and the contour accuracy $\mathcal{F}$ for evaluation. For YouTube-VOS, we uploaded our results to the online evaluation server [16]. For DAVIS, we used the official benchmark code [17]. Our code and model will be available online.

TABLE 1: The quantitative evaluation of semi-supervised video object segmentation on YouTube-VOS [25] validation set. Results for other methods are directly copied from [16], [27], [28], [55].

	Overall	Seen		Unseen	
		$\mathcal{J}$	$\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$
OSMN [8]	51.2	60.0	60.1	40.6	44.0
MSK [3]	53.1	59.9	59.5	45.0	47.9
RGMP [7]	53.8	59.5	-	45.2	-
OnAVOS [26]	55.2	60.1	62.7	46.6	51.4
RVOS [28]	56.8	63.6	67.2	45.5	51.0
OSVOS [4]	58.8	59.8	60.5	54.2	60.7
S2S [25]	64.4	71.0	70.0	55.5	61.2
A-GAME [27]	66.1	67.8	-	60.8	-
PReMVOS [56]	66.9	71.4	75.9	56.5	63.7
BoLTVOS [55]	71.1	71.6	-	64.3	-
Ours	**79.4**	**79.7**	**84.2**	**72.8**	**80.9**

TABLE 2: The quantitative evaluation of semi-supervised video object segmentation on the DAVIS-2016 validation set. OL indicates online learning. **(+YV)** indicates the use of YouTube-VOS for training.

	OL	$\mathcal{J}$ Mean	$\mathcal{F}$ Mean	Time(s)
BVS [57]		60.0	58.8	0.37
OFL [58]		68.0	63.4	120
PLM [5]	✓	70.0	62.0	0.3
VPN [59]		70.2	65.5	0.63
OSMN [8]		74.0	72.9	0.14
SFL [60]	✓	74.8	74.5	7.9
PML [24]		75.5	79.3	0.27
S2S **(+YV)** [25]	✓	79.1	-	9
MSK [3]	✓	79.7	75.4	12
OSVOS [4]	✓	79.8	80.6	9
MaskRNN [2]	✓	80.7	80.9	-
VideoMatch [6]		81.0	-	0.32
FEELVOS **(+YV)** [9]		81.1	82.2	0.45
RGMP [7]		81.5	82.0	0.13
A-GAME **(+YV)** [27]		82.0	82.2	**0.07**
FAVOS [61]		82.4	79.5	1.8
LSE [62]	✓	82.9	80.3	-
CINN [23]	✓	83.4	85.0	>30
PReMVOS [56]	✓	84.9	88.6	>30
OSVOS^S [22]	✓	85.6	86.4	4.5
OnAVOS [26]	✓	86.1	84.9	13
DyeNet [21]	✓	86.2	-	2.32
Ours		84.8	88.1	0.10
Ours **(+YV)**		**88.7**	**89.9**	0.10

4.1.1 YouTube-VOS

YouTube-VOS [25] is the latest large-scale dataset for video object segmentation and it consists of 4453 videos annotated with multiple objects. The dataset is about 30 times larger than the popular DAVIS benchmark that consists of 120 videos. It also has validation data for the unseen object categories. Thus, it is suitable for measuring the generalization performance of different algorithms. The validation set consists of 474 videos including 91 object categories. It has separate measures for 65 of seen and 26 of unseen object categories. We compare our method to existing methods that are trained on YouTube-VOS training set by [16], [27], [55]. As shown in Table 1, our method significantly outperforms all other methods in every evaluation metric.

TABLE 3: The quantitative evaluation of semi-supervised video object segmentation on the DAVIS-2017 validation set. OL indicates online learning. **(+YV)** indicates the use of YouTube-VOS for training. *: average of $\mathcal{J}$ Mean and $\mathcal{F}$ Mean. † indicates timing approximated from DAVIS-2016 assuming linear scaling in the number of objects.

	OL	$\mathcal{J}$ Mean	$\mathcal{F}$ Mean	Time(s)
OSMN [8]		52.5	57.1	0.28†
FAVOS [61]		54.6	61.8	1.2†
VidMatch [6]		56.5	68.2	0.35
OSVOS [4]	✓	56.6	63.9	18†
MaskRNN [2]	✓	60.5	-	-
OnAVOS [26]	✓	64.5	71.2	26
OSVOS^S [4]	✓	64.7	71.3	9†
RGMP [7]		64.8	68.6	0.26†
CINN [23]	✓	67.2	74.2	>120
A-GAME **(+YV)** [27]		67.2	72.7	0.14†
FEELVOS **(+YV)** [9]		69.1	74.0	0.51
DyeNet [21]	✓	*74.1		9.32†
BoLTVOS [55]	✓	72.0	80.6	1.45
PReMVOS [56]	✓	73.9	81.7	37.6
Ours		69.2	74.0	**0.13**
Ours **(+YV)**		**79.2**	**84.3**	**0.13**

4.1.2 DAVIS-2016

DAVIS-2016 [17] is one of the most popular benchmark datasets for video object segmentation tasks. We use the validation set that contains 20 videos annotated with high-quality masks, each with a single target object. We compare our method with the state-of-the-art methods in Table 2. In the table, we indicate the use of online learning and provide the approximate runtimes of each method. Most of the previous top-performing methods relied on online learning that severely harms the running speed. Our method achieves the best accuracy out of all the competing methods without online learning, and produces competitive results with the top-performing online learning based methods while running in a fraction of the time. Our method trained with additional data from YouTube-VOS outperforms all the other methods by a large margin.

4.1.3 DAVIS-2017

DAVIS-2017 [18] is a multi-object extension of DAVIS-2016. The validation set consists of 59 objects in 30 videos. In Table Table 3, we report the results of multi-object video segmentation on the validation set. Again, our method shows the best performance among fast methods without online learning. With additional training data from YouTube-VOS, our method largely outperformed all the previous state-of-the-art methods. Our results on the test-dev set are included in the supplementary materials.

The large performance leap achieved by using additional training data indicates that DAVIS is too small to train a generalizable deep network due to over-fitting. It also explains why top performing online learning methods on the DAVIS benchmark do not show good performance on the large-scale YouTube-VOS benchmark. Online learning methods are hardly aided at all by a large amount of training data. Those methods usually require an extensive parameter search (e.g., data synthesis methods, optimization iterations,

Fig. 6: The qualitative results of semi-supervised video object segmentation on YouTube-VOS and DAVIS. Frames are sampled at important moments (e.g.before and after occlusions).

learning rate, and post-processing), which is not easy to do for a large-scale benchmark.

4.1.4 Qualitative Results

Fig. 6 shows qualitative examples of our results. We have chosen challenging videos from YouTube-VOS and DAVIS validation sets and sample important frames (e.g.before and after occlusions). As can be seen in the figure, our method is robust to occlusions and complex motions. All of our pre-computed results will be available online.

4.2 Interactive Video Object Segmentation

In the interactive video object segmentation, the users are deeply involved in the process of segmentation. The segmentation results are made based on the user inputs, and the user draws scribbles to give a correction based on the segmentation results. Thus, it is difficult to quantify the performance of the interactive model as it is affected by the user's skill. To resolve this issue, Caelles et al. [1] introduced a framework for evaluating interactive video object segmentation[7]. First, initial scribbles are collected by hiring human annotators. Three different initial scribbles for each video sequence are provided and each one serves as a starting point for interactive segmentation. For the feedback scribbles, the framework involves the robot agent acting like a human who draws scribbles according to the intermediate results of an algorithm. Through this framework, interactive algorithms produce results that work with the same user, and thus a fair comparison can be made.

The performance is measured using two metrics: area under the curve (AUC) and the accuracy at 60 seconds ($\mathcal{J}\&\mathcal{F}@60s$). Both metrics are computed on a curve showing the mean of the region similarity ($\mathcal{J}$) and the contour accuracy ($\mathcal{F}$) as a function of time. AUC measures the overall accuracy of the segmentation process that may go through multiple rounds. $\mathcal{J}\&\mathcal{F}@60s$ measures the accuracy with a 60 second time budget. To be specific, this metric is computed by performing an interpolation of the curve at 60 seconds.

7. the DAVIS evaluation framework is available at https://github.com/albertomontesg/davis-interactive.

4.2.1 State-of-the-art Comparison

We compare our methods against both the baseline and the state-of-the art methods: Scribble-OSVOS [1] and IPNet [15]. Scribble-OSVOS [1] is a baseline method that adapts OSVOS [4] to train only on scribbles instead of the full mask. IPNet [15] is one of the best performing methods and it won the DAVIS interactive challenge 2018 [1]. IPNet incorporates two networks each for the interactive image segmentation and the mask propagation. During the evaluation, we use the validation set of the DAVIS-2017 dataset and we let the methods interact with the robot agent up to 8 times. The $\mathcal{J}\&\mathcal{F}$ scores with the growing number of interactions are shown in Fig. 7. The baseline method is not on the same level as the other methods. In the first round, IPNet shows the best performance, but from the second round, our method takes a big lead with additional interactions creating a huge performance gap.

4.2.2 DAVIS Interactive Challenge 2019

To compare our method with other cutting-edge techniques, we submitted our results to the DAVIS interactive challenge 2019 [63]. For the challenge, the test-dev set of DAVIS-2017, where the ground truth segmentation is not publicly available, is used. The challenge results are summarized in Table 4. Our method won first place outperforming all the challenge participants by a large margin [19].

4.2.3 Qualitative Results

Fig. 8 shows qualitative examples of the interactive segmentation process with our model. We use the DAVIS evaluation framework for simulating the user inputs. As shown in Fig. 8, our method is capable of segmenting target objects in challenging test videos that include similar-looking objects and complex object motions with the aid of

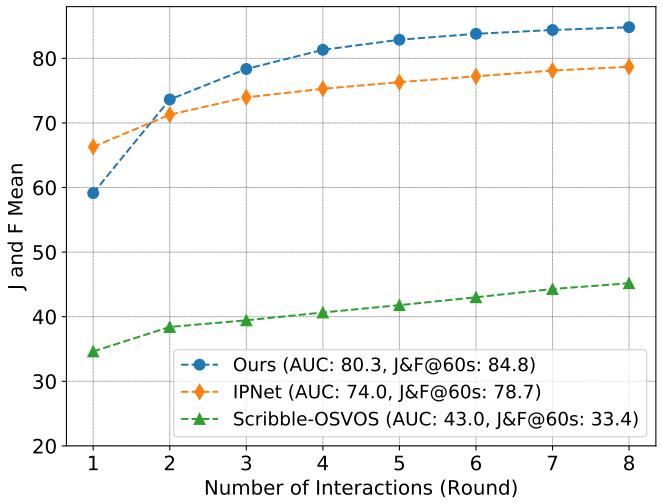


Fig. 7: State-of-the-art comparison of interactive video object segmentation on the DAVIS-2017 validation set. The $\mathcal{J}\&\mathcal{F}$ Mean with the growing number of interactions is shown. We compare our model against IPNet [15] and Scribble-OSVOS [1]. The AUC and $\mathcal{J}\&\mathcal{F}@60s$ are shown in the legend.

TABLE 4: Leaderboard for The DAVIS interactive challenge 2019.

Participant	AUC	$\mathcal{J}\&\mathcal{F}@60s$
Ours	**78.3**	**79.1**
Yuk Heo (Korea University) [64]	64.7	60.9
Zihang Lin (Sun Yat-sen University) [65]	62.1	60.1
YK_CL (Youku Company) [66]	58.9	49.1
Yang Yu	52.4	53.2

user scribbles. For the real user example, we provide a demo video that records real-time demos with real users using our GUI application. Our demo software will be made available online.

5 Analysis

5.1 Training Data

We trained our model through two training stages: the pre-training on static images [47], [48], [49], [50], [51] and the main training on video datasets [18], [25]. In this section, we validate and analyze the effect of the pre-training and the main training through the ablation study on the semi-supervised video object segmentation. In Table 5, we compare the performance of our model trained with different training data. In addition, we provide a cross-dataset validation to measure the generalization performance.

TABLE 5: Training data analysis on YouTube-VOS and DAVIS-2017 validation sets. We compare models trained through different training stages (Sec. 3.5). In addition, we report the cross-validation results (i.e., evaluating DAVIS using the model trained on YouTube-VOS, and vice versa.).

Variants	YouTube-VOS	DAVIS-2017	
	Overall	$\mathcal{J}$	$\mathcal{F}$
Pre-training only	69.1	57.9	62.1
Main training only	68.2	38.1	47.9
Full training	**79.4**	69.2	74.0
Cross validation	56.3	**78.6**	**83.5**

TABLE 6: Memory management analysis on the validation sets of YouTube-VOS and DAVIS. We compare results obtained by different memory management rules. We report *Overall* and $\mathcal{J}$ Mean scores for YouTube-VOS and DAVIS, respectively. Time is measured on DAVIS-2017.

Memory frame(s)	YouTube -VOS	DAVIS		Time
		2016	2017	
First	68.9	81.4	67.0	**0.07**
Previous	69.7	83.2	69.6	0.10
First & Previous	78.4	87.8	77.7	0.11
+ Every 5 frames	**79.4**	**88.7**	**79.2**	0.13

5.1.1 Pre-training Only

It is interesting that our pre-training only model outperforms the main train only model as well as all other methods on YouTube-VOS, without using any real video. However, we get maximum performance by using both training strategies.

5.1.2 Main Training Only

Without the pre-training stage, the performance of our model drops by 11.2 in the *Overall* score on YouTube-VOS [25]. This indicates that the amount of training video data is still insufficient to bring out the potential of our network even though YouTube-VOS [25] provides more than 3000 training videos. In addition, very low performance on DAVIS implies a severe over-fitting issue as the training loss was similar to the complete model (we did not apply early stopping). We conjecture that diverse objects encountered during the pre-training helped our model's generalization and also to prevent over-fitting.

5.1.3 Cross Validation

We evaluate our model trained on DAVIS to YouTube-VOS, and vice versa. Our model trained on DAVIS shows limited performance on YouTube-VOS. This was an expected result because DAVIS is too small to learn a generalization ability to other datasets. On the other hand, our model trained on YouTube-VOS performs well on DAVIS and outperforms all other methods.

5.2 Effect of Memory

In this section, we analyze what are the positive effects of larger memory. In the semi-supervised scenario, we put the previous frames with estimated object masks into the memory. On the other hand, frames given user inputs are used to build up the memory in the interactive scenario.

5.2.1 Semi-supervised Scenario

For the minimal memory consumption and fastest runtime, we can save either the first and/or the previous frames in the memory. For the maximum accuracy, our final model

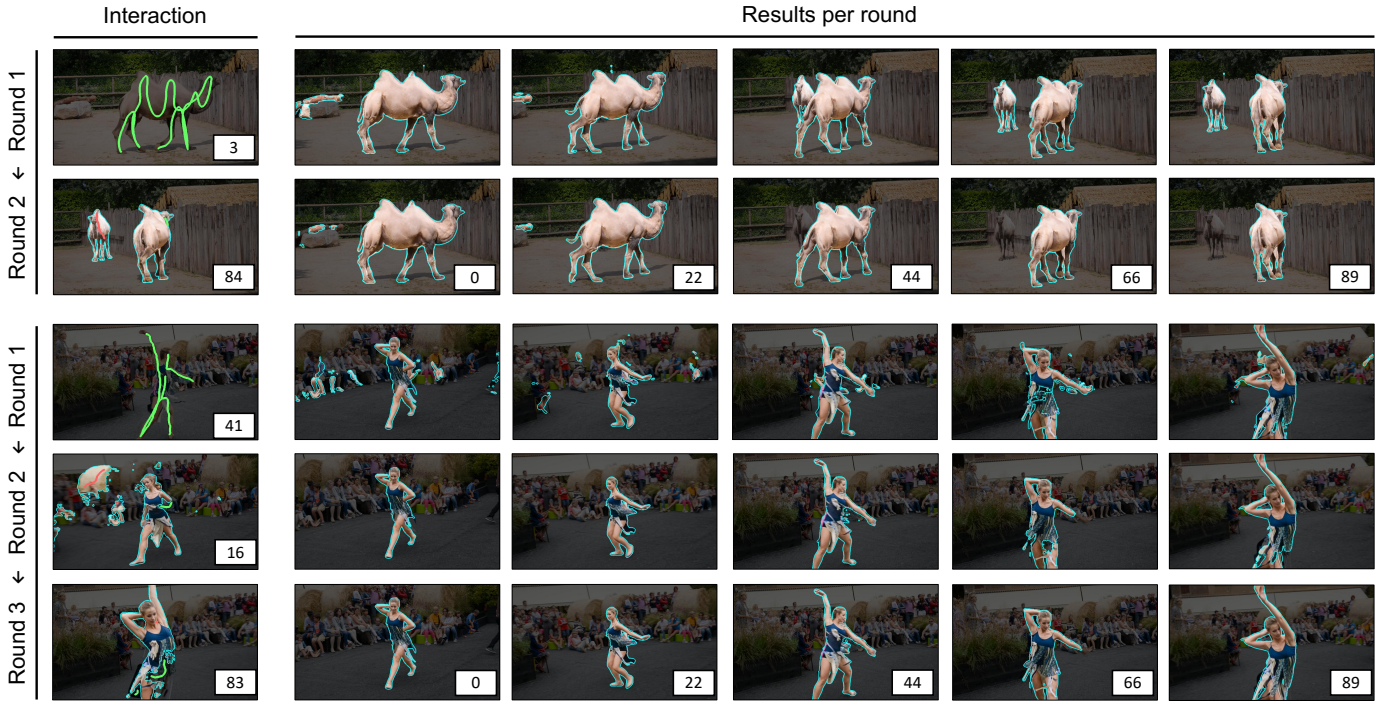


Fig. 8: The qualitative results for interactive video object segmentation on the DAVIS-2017 validation set. For every round, additional memory frames (initial or feedback) with user interaction and the corresponding results are shown. Scribbles are automatically generated by the robot agent provided by [1]. Feedback interactions are given for the worst frame of the previous round (chosen by the robot agent [1]) and results are shown for uniformly sampled frames.

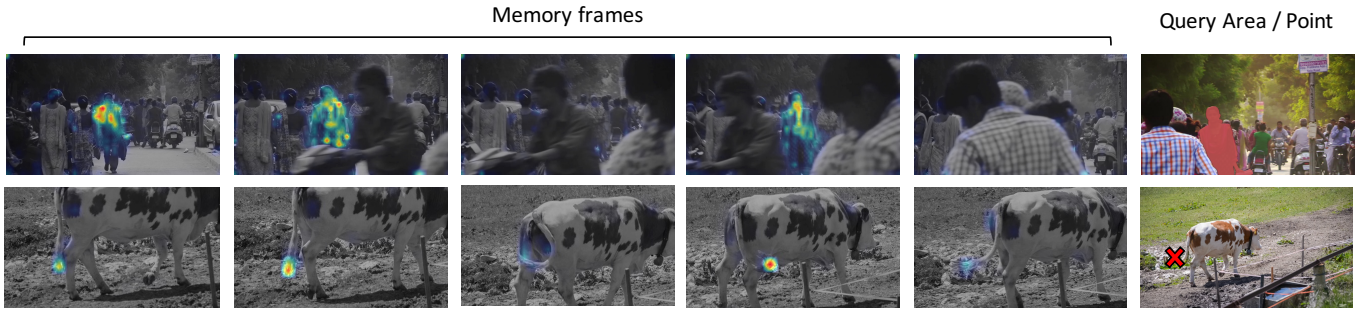


Fig. 9: Visualization of our space-time read operation. We first compute the similarity scores in Eq. (2) for the pixel(s) in the query image (marked in red), then visualize the normalized soft weights to the memory frames. (top) We visualize the averaged weights for the pixels inside the object area. (bottom) We visualize the retrieved weights for the selected pixel. We enlarged some memory frames because the area of interest is too small for visualization.

Fig. 10: Visual comparisons of the results both with and without the use of the intermediate frame memories.

saves a new intermediate memory frame at every fifth frame in addition to the first and the previous frames as explained in Sec. 3.6.2.

We compare different memory management rules in Table 6. Saving both the first and the previous frame in the memory is most important, and our model achieves state-of-the-art accuracy with the two memory frames. This is because our model is strong enough to handle large appearance changes while being resistant to drifting and error accumulation by effectively exploiting the memory. On top of that, having the intermediate frame memories further boosts performance by tackling extremely challenging cases as shown in Fig. 10.

For a deeper analysis, we show the frame-level accuracy distribution in Fig. 11. We sort Jaccard scores of all the objects in all the video frames and plot the scores in order to analyze the performance on challenging scenes. We compare our final model (*Every 5 frames*) with *First and Previous* to check the effect of using additional memory frames.

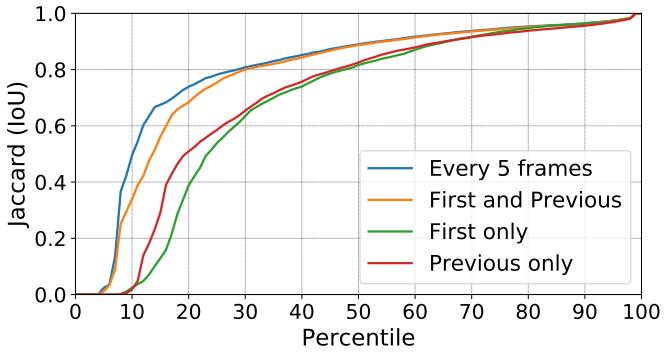


Fig. 11: Jaccard score distribution on DAVIS-2017.

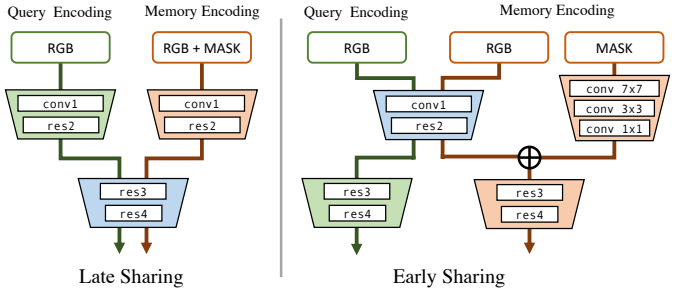


Fig. 12: Our weight sharing models. The examples assume that the late sharing model shares `res3` and `res4`, and the early sharing model shares `conv1` and `res2`. $\bigoplus$ indicates element-wise addition.

While both settings perform equally well on the successful range (over 30th percentile), the effect of additional memory frames becomes clearly visible for difficult cases (under 30th percentile). The huge accuracy gap between 10th and 30th percentile indicates that our network handles challenging cases better with additional memory frames. Comparing *First only* and *Previous only*, the previous frame looks more useful for handling failure cases.

5.2.2 Interactive Scenario

As can be seen in Fig. 7, we can easily observe the positive effect of larger memory in the interactive scenario. For our method on the plot, the number of interactions (rounds) also indicates the number of frames in the memory. The more user inputs, the better the segmentation quality.

The memory in the interactive scenario has different characteristics than that in the semi-supervised scenario. While scribbles is a sparse and incomplete form of annotation, it is reliable as it is provided by the user. In contrast, most of the mask annotations (except for the first frame) in the semi-supervised scenario are from the network thus it often contains errors. Accumulated scribbles eventually provide more information than the complete mask annotation. For example, our interactive model outperforms our semi-supervised model in more than 5 user interactions.

5.3 Weight Sharing for Encoders

The query and memory encoders are different depending on whether or not the object location information (i.e., segmentation masks) needs to be encoded together. In designing a shared encoder, the type and amount of resources that can be saved depends on when location information is injected. Late sharing is a scheme in which the networks share the weights *after* the location information is injected. By this way, the network parameters in the deeper layers (e.g. Res3 and Res4) – that account for most of the parameters – can be shared. However, there is no advantage in terms of the computational complexity (FLOPs) as the intermediate feature representation cannot be shared. Early sharing is a scheme in which the networks shares the weights *before* the location information is fused. Specifically, the forepart of the backbone network is shared but it starts to have separated weights once the mask information is injected for memory encoding. With this sharing scheme, we can spare not only the number of parameters but also the computational overhead by reusing pre-computed features. Examples of late and early sharing models are illustrated in Fig. 12.

TABLE 7: Weight sharing configurations and results. Our weight sharing models are evaluated on DAVIS-2017 validation set in the semi-supervised setting. GFLOPs are computed assuming the input resolution as 384×384.

Sharing Type	Shared Blocks			DAVIS-2017		Two Encoders	
	R2	R3	R4	$\mathcal{J}$	$\mathcal{F}$	#Params	GFLOPs
Late	✓	✓	✓	77.4	81.8	8.56M	19.67
		✓	✓	77.0	82.2	8.77M	19.67
Early	✓			78.3	83.4	16.90M	17.53
	✓	✓		**78.5**	83.2	15.80M	14.66
	✓	✓	✓	75.2	80.6	10.25M	11.11
No				78.2	**84.3**	17.09M	19.67

The results from our weight sharing models are shown in Table 7. Late sharing significantly reduces the number of parameters (almost by a half) while not sacrificing the performance a lot. Early sharing is advantageous not only to the model size but also to computational complexity. Surprisingly, early sharing to a certain extent does not degrade the performance.

5.4 Memory Visualization

In Fig. 9, we visualize our memory read operation to validate the learned space-time matching. As can be observed in the visualization, our read operation accurately matches corresponding pixels between the query and the memory frames.

6 Conclusion

We have presented a novel space-time memory network for the user-guided video object segmentation. Two types of user guidance are considered: **masks** in the semi-supervised scenario and **scribbles** in the interactive scenario. In both scenarios, our method performs the best among the existing methods in terms of both the accuracy and the speed. We believe the proposed space-time memory network has great potential to become a breakthrough method in addressing many other pixel-level estimation problems. Regarding future work, we are looking for other applications that

are suitable for our framework including object tracking, interactive image/video segmentation, and inpainting.

REFERENCES

[1] S. Caelles, A. Montes, K.-K. Maninis, Y. Chen, L. Van Gool, F. Perazzi, and J. Pont-Tuset, "The 2018 davis challenge on video object segmentation," *arXiv preprint arXiv:1803.00557*, 2018.

[2] Y.-T. Hu, J.-B. Huang, and A. Schwing, "Maskrnn: Instance level video object segmentation," in *Advances in Neural Information Processing Systems*, 2017.

[3] F. Perazzi, A. Khoreva, R. Benenson, B. Schiele, and A. Sorkine-Hornung, "Learning video object segmentation from static images," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[4] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool, "One-shot video object segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017.

[5] J. S. Yoon, F. Rameau, J. Kim, S. Lee, S. Shin, and I. S. Kweon, "Pixel-level matching for video object segmentation using convolutional neural networks," in *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[6] Y.-T. Hu, J.-B. Huang, and A. Schwing, "Videomatch: Matching based video object segmentation," in *European Conference on Computer Vision (ECCV)*, 2018.

[7] S. W. Oh, J.-Y. Lee, K. Sunkavalli, and S. J. Kim, "Fast video object segmentation by reference-guided mask propagation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[8] L. Yang, Y. Wang, X. Xiong, J. Yang, and A. K. Katsaggelos, "Efficient video object segmentation via network modulation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[9] P. Voigtlaender, Y. Chai, F. Schroff, H. Adam, B. Leibe, and L.-C. Chen, "Feelvos: Fast end-to-end embedding learning for video object segmentation," in *CVPR*, 2019.

[10] S. W. Oh, J.-Y. Lee, N. Xu, and S. J. Kim, "Video object segmentation using space-time memory networks," in *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[11] S. Sukhbaatar, J. Weston, R. Fergus *et al.*, "End-to-end memory networks," in *Advances in neural information processing systems*, 2015, pp. 2440–2448.

[12] A. Miller, A. Fisch, J. Dodge, A.-H. Karimi, A. Bordes, and J. Weston, "Key-value memory networks for directly reading documents," *arXiv preprint arXiv:1606.03126*, 2016.

[13] A. Kumar, O. Irsoy, P. Ondruska, M. Iyyer, J. Bradbury, I. Gulrajani, V. Zhong, R. Paulus, and R. Socher, "Ask me anything: Dynamic memory networks for natural language processing," in *International Conference on Machine Learning*, 2016, pp. 1378–1387.

[14] A. Benard and M. Gygli, "Interactive video object segmentation in the wild," *arXiv preprint arXiv:1801.00269*, 2017.

[15] S. W. Oh, J.-Y. Lee, N. Xu, and S. J. Kim, "Fast user-guided video object segmentation by interaction-and-propagation networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[16] N. Xu, L. Yang, D. Yue, J. Yang, Y. Fan, Y. Liang, and T. Huang, "Youtube-vos: A large-scale video object segmentation benchmark," in *arXiv preprint arXiv:1809.03327*, 2018.

[17] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung, "A benchmark dataset and evaluation methodology for video object segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[18] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool, "The 2017 davis challenge on video object segmentation," *arXiv:1704.00675*, 2017.

[19] S. W. Oh, J.-Y. Lee, N. Xu, and S. J. Kim, "A unified model for semi-supervised and interactive video object segmentation using space-time memory networks," *The 2019 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2019.

[20] A. Khoreva, R. Benenson, E. Ilg, T. Brox, and B. Schiele, "Lucid data dreaming for object tracking," *arXiv preprint arXiv:1703.09554*, 2017.

[21] X. Li and C. C. Loy, "Video object segmentation with joint re-identification and attention-aware mask propagation," in *European Conference on Computer Vision (ECCV)*, 2018.

[22] K.-K. Maninis, S. Caelles, Y. Chen, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool, "Video object segmentation without temporal information," *arXiv preprint arXiv:1709.06031*, 2017.

[23] L. Bao, B. Wu, and W. Liu, "Cnn in mrf: Video object segmentation via inference in a cnn-based higher-order spatio-temporal mrf," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[24] Y. Chen, J. Pont-Tuset, A. Montes, and L. Van Gool, "Blazingly fast video object segmentation with pixel-wise metric learning," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[25] N. Xu, L. Yang, D. Yue, J. Yang, B. Price, J. Yang, S. Cohen, Y. Fan, Y. Liang, and T. Huang, "Youtube-vos: Sequence-to-sequence video object segmentation," in *European Conference on Computer Vision (ECCV)*, 2018.

[26] P. Voigtlaender and B. Leibe, "Online adaptation of convolutional neural networks for video object segmentation," in *British Machine Vision Conference*, 2017.

[27] J. Johnander, M. Danelljan, E. Brissman, F. S. Khan, and M. Felsberg, "A generative appearance model for end-to-end video object segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[28] C. Ventura, M. Bellver, A. Girbau, A. Salvador, F. Marques, and X. Giro-i Nieto, "Rvos: End-to-end recurrent network for video object segmentation," in *CVPR*, 2019.

[29] B. Bratt, *Rotoscoping: Techniques and tools for the Aspiring Artist*. Focal Press, 2012.

[30] W. Li, F. Viola, J. Starck, G. J. Brostow, and N. D. Campbell, "Roto++: Accelerating professional rotoscoping using shape manifolds," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, p. 62, 2016.

[31] J. Wang, P. Bhat, R. A. Colburn, M. Agrawala, and M. F. Cohen, "Interactive video cutout," in *ACM Transactions on Graphics (ToG)*, vol. 24. ACM, 2005, pp. 585–594.

[32] B. L. Price, B. S. Morse, and S. Cohen, "Livecut: Learning-based interactive video segmentation by evaluation of multiple propagated cues," in *IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2009, pp. 779–786.

[33] X. Bai, J. Wang, D. Simons, and G. Sapiro, "Video snapcut: robust video object cutout using localized classifiers," in *ACM Transactions on Graphics (ToG)*, vol. 28. ACM, 2009, p. 70.

[34] A. Agarwala, A. Hertzmann, D. H. Salesin, and S. M. Seitz, "Keyframe-based tracking for rotoscoping and animation," in *ACM Transactions on Graphics (ToG)*, vol. 23, no. 3. ACM, 2004, pp. 584–591.

[35] N. Shankar Nagaraja, F. R. Schmidt, and T. Brox, "Video segmentation with just a few strokes," in *IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 3235–3243.

[36] F. Zhong, X. Qin, Q. Peng, and X. Meng, "Discontinuity-aware video object cutout," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 6, p. 175, 2012.

[37] N. Xu, B. Price, S. Cohen, J. Yang, and T. S. Huang, "Deep interactive object selection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 373–381.

[38] C. C. Park, B. Kim, and G. Kim, "Attend to you: Personalized image captioning with context sequence memory networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 6432–6440.

[39] T. Yang and A. B. Chan, "Learning dynamic memory networks for object tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 152–167.

[40] S. Na, S. Lee, J. Kim, and G. Kim, "A read-write memory network for movie story understanding," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 677–685.

[41] S. Lee, J. Sung, Y. Yu, and G. Kim, "A memory network approach for story-based temporal summarization of 360 videos," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 1410–1419.

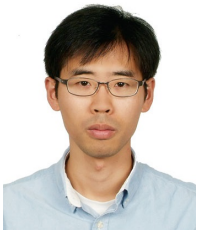
[42] S. W. Oh, S. Lee, J.-Y. Lee, and S. J. Kim, "Onion-peel networks for deep video completion," in *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

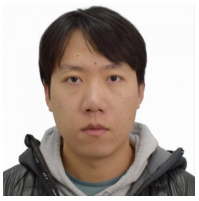
[44] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in

Advances in Neural Information Processing Systems, 2017, pp. 5998–6008.
[45] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
[46] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *European Conference on Computer Vision (ECCV)*. Springer, 2016, pp. 630–645.
[47] J. Shi, Q. Yan, L. Xu, and J. Jia, "Hierarchical image saliency detection on extended cssd," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 4, pp. 717–729, 2016.
[48] M.-M. Cheng, N. J. Mitra, X. Huang, P. H. Torr, and S.-M. Hu, "Global contrast based salient region detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 569–582, 2015.
[49] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.
[50] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik, "Semantic contours from inverse detectors," in *IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2011, pp. 991–998.
[51] 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 (ECCV)*. Springer, 2014, pp. 740–755.
[52] J. Yang, S. E. Reed, M.-H. Yang, and H. Lee, "Weakly-supervised disentangling with recurrent transformations for 3d view synthesis," in *Advances in Neural Information Processing Systems*, 2015, pp. 1099–1107.
[53] Z. Guo and R. W. Hall, "Parallel thinning with two-subiteration algorithms," *Commun. ACM*, vol. 32, no. 3, pp. 359–373, Mar. 1989. [Online]. Available: http://doi.acm.org/10.1145/62065.62074
[54] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.
[55] P. Voigtlaender, J. Luiten, and B. Leibe, "Boltvos: Box-level tracking for video object segmentation," *arXiv preprint arXiv:1904.04552*, 2019.
[56] J. Luiten, P. Voigtlaender, and B. Leibe, "Premvos: Proposal-generation, refinement and merging for video object segmentation," in *Asian Conference on Computer Vision*, 2018.
[57] N. Märki, F. Perazzi, O. Wang, and A. Sorkine-Hornung, "Bilateral space video segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 743–751.
[58] Y.-H. Tsai, M.-H. Yang, and M. J. Black, "Video segmentation via object flow," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 3899–3908.
[59] V. Jampani, R. Gadde, and P. V. Gehler, "Video propagation networks," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
[60] J. Cheng, Y.-H. Tsai, S. Wang, and M.-H. Yang, "Segflow: Joint learning for video object segmentation and optical flow," *IEEE International Conference on Computer Vision (ICCV)*, 2017.
[61] J. Cheng, Y.-H. Tsai, W.-C. Hung, S. Wang, and M.-H. Yang, "Fast and accurate online video object segmentation via tracking parts," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
[62] H. Ci, C. Wang, and Y. Wang, "Video object segmentation by learning location-sensitive embeddings," in *European Conference on Computer Vision (ECCV)*, 2018.
[63] S. Caelles, J. Pont-Tuset, F. Perazzi, A. Montes, K.-K. Maninis, and L. Van Gool, "The 2019 davis challenge on vos: Unsupervised multi-object segmentation," *arXiv preprint arXiv:1905.00737*, 2019.
[64] Y. Heo, Y. J. Koh, and C. Kim, "Interactive video object segmentation using sparse-to-dense networks," *The 2019 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2019.
[65] Z. Lin, J. Xie, C. Zhou, J. Hu, and W. Zheng, "Interactive video object segmentation via spatio-temporal context aggregation and online learning," *The 2019 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2019.
[66] H. Ren, Y. Yang, and X. Liu, "Robust multiple object mask propagation with efficient object tracking," *The 2019 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2019.

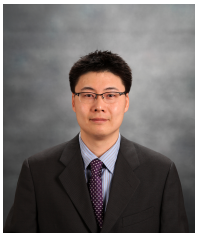
Seoung Wug Oh is a Ph.D. Student in Computer Science at Yonsei University. He is a member of the Computational Intelligent and Photography Lab with Prof. Seon Joo Kim as his advisor. He got his B.S. degree in Computer Science from Yonsei University, in 2014. His research focuses on computer vision, deep learning, and computational photography. In particular, He is interested in deep learning for image/video enhancement and editing.



Joon-Young Lee is a Senior Research Scientist at Adobe Research. He received his Ph.D. and M.S. degrees in Electrical Engineering from KAIST, Korea in 2015 and 2009, respectively. He received the B.S. degree in Electrical and Electronic Engineering from Yonsei University, Korea in 2008. His research interests include deep learning and computer vision. He has served as an Area Chair of ICCV 2019, CVPR 2020, and ECCV 2020.



Ning Xu is a Research Scientist at Adobe Research since 2018. His research interests focus on computer vision and deep learning. He graduated from the ECE Department of University of Illinois at Urbana-Champaign in 2017, advised by Prof. Thomas Huang. Before joining UIUC, he obtained the B.S. degree in Electrical Engineering from Shanghai Jiao Tong University in 2011.



Seon Joo Kim is an Associate Professor in Computer Science Department at Yonsei University, Seoul, Korea. His research interests focus on computer vision, computational photography, and deep learning. He graduated from the CS Department of University of North Carolina at Chapel Hill in 2008. He has served as an Area Chair of ICCV 2017, CVPR 2018/2020, and as a local chair for ICCV 2019. He also serves as an area editor for Computer Vision and Image Understanding.