Hierarchical Spatiotemporal Transformers for Video Object Segmentation

Jun-Sang Yoo
Korea University
junsang7777@korea.ac.kr

Hongjae Lee
Korea University
jimmy9704@korea.ac.kr

Seung-Won Jung*
Korea University
swjung83@korea.ac.kr

Abstract

This paper presents a novel framework called HST for semi-supervised video object segmentation (VOS). HST extracts image and video features using the latest Swin Transformer and Video Swin Transformer to inherit their inductive bias for the spatiotemporal locality, which is essential for temporally coherent VOS. To take full advantage of the image and video features, HST casts image and video features as a query and memory, respectively. By applying efficient memory read operations at multiple scales, HST produces hierarchical features for the precise reconstruction of object masks. HST shows effectiveness and robustness in handling challenging scenarios with occluded and fast-moving objects under cluttered backgrounds. In particular, HST-B outperforms the state-of-the-art competitors on multiple popular benchmarks, i.e., YouTube-VOS (85.0%*), DAVIS 2017 (*85.9%*), and DAVIS 2016 (*94.0%*).*

1. Introduction

Semi-supervised video object segmentation (VOS) is the task of extracting a target object from a video sequence given an object mask of the first frame. It is a very challenging task because the appearance of the target object can change drastically over time. In addition, occlusion, cluttered backgrounds, and other objects similar to the target object make the task further challenging. As this task continues to advance, it has opened up a realm of possibilities for diverse high-level vision applications [xxx]. Consequently, extensive research has been conducted on semi-supervised VOS over the last decade. The interested reader can refer to [14, 35, 60] for a systematic literature review.

Recently, memory-based VOS methods [5, 15–17, 19, 22, 23, 25, 28, 32, 33, 37, 46, 49, 50, 53] have achieved remarkable performance. The key idea is to build a memory containing the information from the past frames with given or predicted masks and use the current frame as a query for matching. As shown in Figure 1(a), these methods typically apply a convolutional neural network (CNN)-based encoder to each frame and perform dense matching between the features extracted from the query and memory. Due to the non-local nature of this matching, they show robustness in handling moving objects and cameras. In particular, the space-time memory network (STM) [32] introduces a space-time memory read operation that performs dense matching between the query and the memory in the feature space to cover all space-time pixel locations. However, the global-to-global matching in STM requires high computational complexity and suffers from false matching to objects similar to the target object. Therefore, many follow-up studies attempted to enforce local constraints using kernelized memory [37] and optical flow [53].

Meanwhile, the success of the Vision Transformer (ViT) [11] has brought significant attention to a Transformer-based solution for VOS. Several recent Transformer-based methods [12, 30, 48, 58] have shown state-of-the-art performance on several VOS benchmarks. However, these methods apply an image Transformer to each frame, as shown in Figure 1(b), and are thus still challenging to enforce Transformers to handle the temporal coherence of the segmentation. In this paper, we introduce a new approach that fully exploits spatiotemporal features for semi-supervised VOS. Inspired by the Swin Transformer [26] and its extension to video frames, called the Video Swin Transformer [27], we propose a novel integration of them for VOS, called HST. HST first extracts multi-scale features from the image and video using their respective Transformers, as shown in Figure 1(c). Then, the image features from the current frame are used as a query, and the video features from the past frames and their object masks are used as memory. Although HST performs dense matching between the query and memory, it does not suffer from false matching to objects similar to the target object due to the locality inductive bias of the Swin Transformers. We also apply an efficient hierarchical memory read operation to reduce computational complexity. HST shows robustness in segmenting small, fast-moving, and occluded objects under cluttered backgrounds. Our baseline model, HST-B, yields competitive performance in several VOS

*Corresponding author.

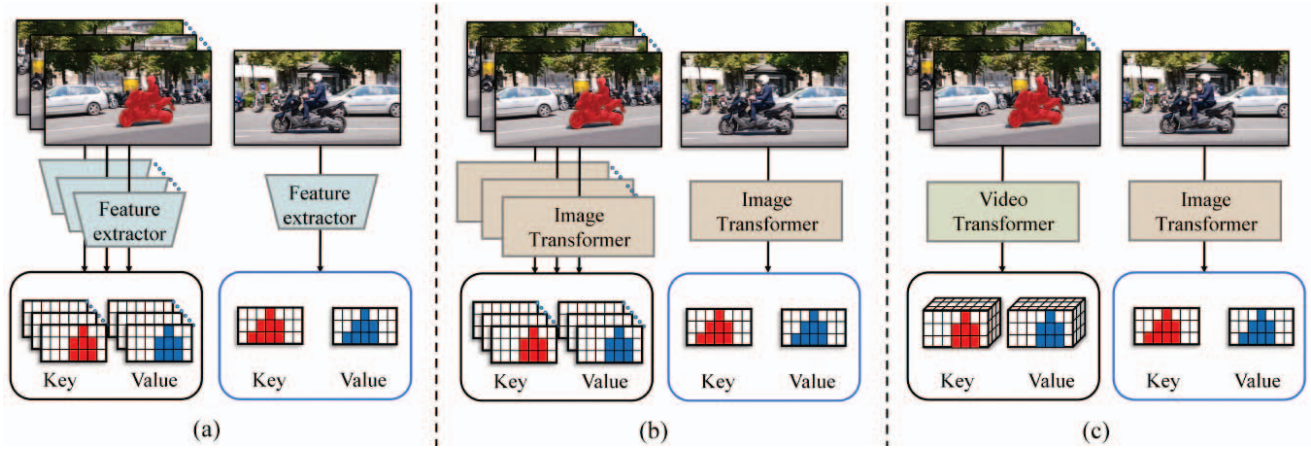

Figure 1: Comparison of the methods for extracting key and value maps from current and past frames: (a) Feature extractor is applied to each frame, (b) image Transformer is applied to each frame, and (c) image and video Transformers are applied to current and past frames, respectively.

benchmarks, including the YouTube-VOS 2018 and 2019 validation datasets (85.0% & 84.9%) and the DAVIS 2016 validation (94.0%) and 2017 validation and test datasets (85.9% & 79.9%).

The main contributions are summarized as follows:

- We propose a Swin Transformer-inspired VOS framework called HST that uses image and video Swin Transformers to extract spatial and spatiotemporal features. To the best of our knowledge, HST is the first to integrate image and video Swin Transformers for VOS.
- We apply a dedicated memory read operation for HST that efficiently measures the similarities between multi-scale spatial and spatiotemporal features.
- Experimental results on the DAVIS and YouTube-VOS datasets demonstrate the state-of-the-art performance of HST.

2. Related Work

2.1. Semi-supervised Video Object Segmentation

Semi-supervised VOS methods have been developed to propagate the manual annotation from the first frame to the entire video sequence. Early semi-supervised VOS methods, such as OSVOS [2] and MoNet [51], fine-tune pre-trained networks at test time using the annotation from the first frame as the ground-truth. OnAVOS [45] applies an online adaptation mechanism to use pixels with very confident predictions from the following frames as additional training examples. MaskTrack [34] and PReMVOS [29] further estimate optical flow to facilitate the propagation of the segmentation mask.

Although promising results have been shown, online learning-based methods inevitably have high computational complexity, restricting their practical use. Recent efforts thus have been devoted to offline learning-based methods such that the trained networks can robustly handle any input videos without additional training. To this end, OSMN [56] uses spatial and visual modulators to adapt the segmentation model to the appearance of a specific object. VideoMatch [17] applies a soft matching layer to compute the similarity of the foreground and background between the first frame and every input frame. FEELVOS [44] and CFBI [57, 59] perform pixel-level matching not only between the first and current frames but also between the previous and current frames. STM [32] embeds the past frames and their prediction masks in memory and uses the current frame as the query for global matching. KMN [37], RMNet [53], and HMMN [38] further use local constraints such as optical flow and kernel to overcome the drawback of global matching. STCN [6] extracts key features for each image independently for effective feature reuse and replaces dot product by L2 similarity for better memory coverage.

2.2. Vision Transformers

The Transformer [43] has been introduced as a network architecture solely based on attention mechanisms. Compared to recurrent neural networks (RNNs) that require extensive sequential operations, Transformer networks are more parallelizable and require less training time, making them attractive to several natural language processing tasks [1, 8, 10, 41]. Recently, Transformer networks have been successfully applied to many computer vision tasks and have shown significant performance improvement over CNN-based networks. The representative work called ViT [11] divides an input image into non-overlapping

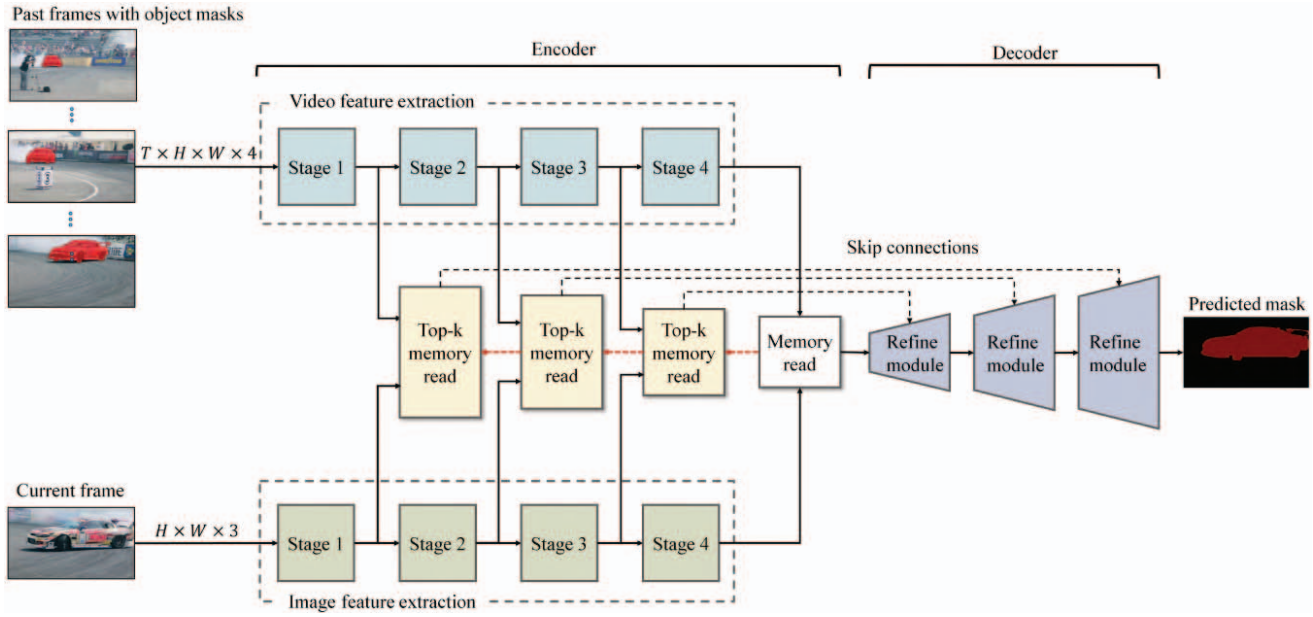

Figure 2: An overview of HST. The encoder consists of image and video Transformers to extract multi-scale spatial and spatiotemporal features from current and past frames. The memory read block performs dense matching on the coarsest scale, and the top-k memory read blocks operate on the finer scales. The decoder receives multi-scale features from the encoder and produces a final mask prediction.

patches and performs linear embedding to construct input for the Transformer encoder. DeiT [42] integrates a teacher-student strategy to the Transformer such that the student model can be efficiently trained on a small dataset. A notable extension of ViT called Swin Transformer [26] builds multi-resolution feature maps on Transformers and restricts self-attention within local windows, leading to linear computational complexity with respect to image size. Video Swin Transformer [27] expands the scope of local attention from the spatial domain to the spatiotemporal domain, achieving state-of-the-art accuracy on several video recognition benchmarks. Furthermore, the powerful feature representation ability of the Swin Transformer has reached outstanding performance on a variety of tasks, including video categorization and image inpainting [3, 9, 18].

2.3. Transformer-based Segmentation

Several recent endeavors have been made to apply vision Transformers to dense prediction tasks. DETR [4] integrates a CNN backbone with a Transformer encoder and decoder to build a fully end-to-end object detector and shows that dense prediction tasks such as panoptic segmentation can be handled by adding a mask head on top of the decoder outputs. SegFormer [52] obtains a segmentation mask using a hierarchical pyramid ViT architecture as an encoder and a simple MLP-based structure with upsampling operations as a decoder. Segmenter [40] exploits a mask Transformer decoder to predict a better segmentation mask. Towards a more VOS-dedicated Transformer design, VIS [48] applies an instance sequence matching and segmentation strategy. TransVOS [30] extracts features from the current frame and reference sets and feeds them to the Transformer encoder to model the temporal and spatial relationships. SST [12] uses a sparse attention-based Transformer block to extract pixel-level embedding and spatial-temporal features. AOT [58] associates multiple target objects into the same embedding space to perform multi-object segmentation as efficiently as single object segmentation. AOT also shows that the performance can be further improved by changing a ResNet encoder to a Swin Transformer encoder.

However, many Transformer-based VOS methods still use a CNN-based encoder for feature extraction [12, 30, 58], limiting the modeling capacity of Transformers. Fully Transformer-based feature extraction methods have been attempted, but they apply the standard Transformer or Swin Transformer [58] to each frame separately, leading to sub-optimal extraction of spatiotemporal information in a video sequence. HST integrates image and video Transformers towards a complete spatiotemporal feature extraction for VOS.

3. Approach

We explain our method for segmenting one target object in a video, but multi-object segmentation can be readily conducted by following the common standard of indepen-

dent segmentation and merging [6, 32, 37, 38]. We use the features extracted from the past frames (with given or estimated object masks) and the current frame as memory and query, respectively. The query should contain spatial information, such as the position, shape, and texture of the target object, and the memory should contain spatiotemporal information, such as the trajectory and deformation of the target object and changes in the background to support temporally coherent target object segmentation. To this end, we present HST that can fully exploit spatial and spatiotemporal information from the current and past frames. Moreover, since dense matching between the query and memory is needed to take full advantage of the information in video frames, we design a hierarchical memory read operation that efficiently matches multi-scale spatial and spatiotemporal features.

Figure 2 illustrates the overall flow of HST. We adopt Swin Transformer [26] and Video Swin Transformer [27] to design a query encoder (image as input) and a memory encoder (images and masks as input). For brevity, we call these two Swin Transformers image Transformer and video Transformer, respectively. Each Transformer extracts multi-scale features, resulting in the key and value maps for matching with each other. The decoder takes all the computed features and outputs a mask prediction. The following subsections detail each component of HST.

3.1. Image Feature Extraction

Our image feature extractor is based on Swin Transformer [26] that incorporates inductive bias for the spatial locality, which is preferable for dense prediction tasks such as VOS. Image Transformer first splits a current frame of size $H \times W \times 3$ into non-overlapping patches of size $P_x \times P_y \times 3$ and applies a linear embedding layer, resulting in a C-dimensional embedding for each patch or "token". Unlike the standard Transformer that computes self-attention across all tokens [43], Swin Transformer computes self-attention only within each window. The query encoder of HST consists of four stacks of Swin Transformer blocks with patch merging blocks for generating multi-scale features [26]. To introduce cross-window connections, a Swin Transformer block is embodied with consecutive multi-head self-attention units with and without a shifted window.

For each window consisting of $M \times M$ patches, the *query*, *key* and *value* matrices Q, K and V are computed as

$$Q = XP_Q, \quad K = XP_K, \quad V = XP_V, \tag{1}$$

where P_Q, P_K and P_V are projection matrices that are shared across different windows. Generally, we have $Q, K, V \in \mathbb{R}^{M^2 \times d}$. The attention matrix is thus computed by the self-attention mechanism in a local window as

$$\text{Attention}(Q, K, V) = \text{SoftMax}(QK^T/\sqrt{d} + B)V, \tag{2}$$

where B is the learnable relative positional encoding. In practice, following [43], we perform the attention function for h times in parallel and concatenate the results for multi-head self-attention (MSA).

Next, a multi-layer perceptron (MLP) that has two fully-connected layers with GELU non-linearity between them is used for further feature transformations. The LayerNorm (LN) layer is added before both MSA and MLP, and the residual connection is employed for both modules. The whole process is formulated as

$$\begin{aligned} Z &= \text{MSA}(\text{LN}(X)) + X, \\ X &= \text{MLP}(\text{LN}(Z)) + Z. \end{aligned} \tag{3}$$

However, when the partition is fixed for different layers, there is no connection across local windows. Therefore, regular and shifted window partitioning are used alternately to enable cross-window connections [26], where shifted window partitioning means shifting the feature by $(\lfloor \frac{M}{2} \rfloor, \lfloor \frac{M}{2} \rfloor)$ pixels before partitioning.

3.2. Video Feature Extraction

Our video feature extractor is based on Video Swin Transformer [27] that extends the scope of local attention from the spatial domain to the spatiotemporal domain. Specifically, the past T frames and their corresponding object masks with size $T \times H \times W \times 4$ (RGB + mask) are divided into non-overlapping patches of size $P_t \times P_x \times P_y \times 4$, followed by a linear embedding layer to obtain a C-dimensional embedding for each token. To incorporate inductive bias for the spatiotemporal locality, video Transformer computes self-attention only within each 3D window. The memory encoder of HST consists of four stacks of Video Swin Transformer blocks with patch merging blocks for generating multi-scale features [27], where each Video Swin Transformer block is embodied with consecutive multi-head self-attention units with and without a 3D shifted window.

consisting of T frames which each contatin $H \times W \times 3$ pixels. We obtain $\frac{T}{P} \times \frac{H}{M} \times \frac{W}{M} \times C$ tokens using a 3D patch partitioning layer. Then, it computes the 3D self-attention separately for each window. We follow equation (1), (2) where $Q, K, V \in \mathbb{R}^{PM^2 \times d}$ by introducing 3D relative position bias $B \in \mathbb{R}^{P^2 \times M^2 \times M^2}$.

The model is built by replacing the MSA module in the standard Transformer layer with the 3D shifted window based MSA module and keeping the other components unchanged. Therefore, the whole process is formulated as

$$\begin{aligned} Z &= \text{3D-MSA}(\text{LN}(X)) + X, \\ X &= \text{3D-MLP}(\text{LN}(Z)) + Z. \end{aligned} \tag{4}$$

Similar to image feature extraction, this 3D shifted window design introduces connections between neighboring non-overlapping 3D windows in the previous layer, where shifted window partitioning means shifting the feature by $(\lfloor \frac{P}{2} \rfloor, \lfloor \frac{M}{2} \rfloor, \lfloor \frac{M}{2} \rfloor)$ pixels before partitioning.

3.3. Memory Read and Decoding

3.3.1 Memory Read

We now have image and video features ready to use for object segmentation. Let $F^i_{image} \in \mathbb{R}^{H_i \times W_i \times C_i}$ and $F^i_{video} \in \mathbb{R}^{T_i \times H_i \times W_i \times C_i}$ denote image and video features obtained after the i-th stage of the query encoder and memory encoder, respectively. The feature dimensions are given as $H_i = H \times \left(\frac{1}{2}\right)^{i+1}$, $W_i = W \times \left(\frac{1}{2}\right)^{i+1}$, and $C_i = C \times 2^{i-1}$ [26, 27]. We fix T_i to T to maintain the temporal resolution. Considering F^i_{image} as a query and F^i_{video} as memory, we extract key and value maps from them [32]. The key and value maps of the query are denoted as $k^Q_i \in \mathbb{R}^{\frac{C_i}{8} \times H_i W_i}$ and $v^Q_i \in \mathbb{R}^{\frac{C_i}{2} \times H_i W_i}$, respectively, and those of the memory are denoted as $k^M_i \in \mathbb{R}^{\frac{C_i}{8} \times T_i H_i W_i}$ and $v^M_i \in \mathbb{R}^{\frac{C_i}{2} \times T_i H_i W_i}$, respectively.

Due to extremely high dimensionality of the key and value maps, we apply dense matching between the query and memory only at the last stage as follows:

$$s_4(\mathbf{q}, \mathbf{p}) = \left(k^Q_4(\mathbf{q})\right)^{\mathrm{T}} k^M_4(\mathbf{p}), \tag{5}$$

$$W_4(\mathbf{q}, \mathbf{p}) = \mathrm{SoftMax}_{\mathbf{p}}(s_4(\mathbf{q}, \mathbf{p})), \tag{6}$$

where $\mathbf{p} = (p_t, p_x, p_y)$ and $\mathbf{q} = (q_x, q_y)$ denote the grid cell locations in the memory and query, respectively, and thus $\left(k^Q_4(\mathbf{q})\right)^{\mathrm{T}} k^M_4(\mathbf{p})$ performs the dot product between two $\frac{C_4}{8}$-dimensional vectors at the locations $\mathbf{p}$ and $\mathbf{q}$ in the memory and query, and T indicates the transpose operator. $s_4 \in \mathbb{R}^{H_4 W_4 \times T_4 H_4 W_4}$ thus contains similarity values in every space-time locations, and $\mathrm{SoftMax}_{\mathbf{p}}$ performs the SoftMax operation along the memory axis. v^M_4 is multiplied with W_4 and then concatenated with v^Q_4 as follows:

$$y_4 = \left[v^Q_4, v^M_4 W^{\mathrm{T}}_4\right], \tag{7}$$

where [,] represents the concatenation along the feature dimension. $y_4 \in \mathbb{R}^{C_4 \times H_4 W_4}$ represents the output of the memory read operation at the fourth stage.

Since the computational complexity required for (Eq. 5)-(Eq. 7) grows quadratically with respect to the size of the feature map, we apply an efficient read operation called top-k read [5, 38] at the earlier stages. Specifically, the affinity maps for the earlier stages s_i $(i = 1, 2, 3)$ are obtained using only the top-k indices as follows:

$$s_i(\mathbf{q}, :) = \left(k^Q_i(\mathbf{q})\right)^{\mathrm{T}} k^M_i(\mathbf{p}), \mathbf{p} \in \Omega^i_{\mathbf{q}}, \tag{8}$$

where $\Omega^i_{\mathbf{q}}$ denotes the set of the top-k indices for the query pixel $\mathbf{q}$ found from s_4 that are mapped to the i-th stage [38]. $\Omega^3_{\mathbf{q}}$, $\Omega^2_{\mathbf{q}}$, and $\Omega^1_{\mathbf{q}}$ contain $4k$ positions in k^M_3, $16k$ positions in k^M_2, and $64k$ positions in k^M_1, respectively, such that more pixels can be matched at the higher scales. $s_i(\mathbf{q}, :)$ thus collects the similarity values in these top $4^{4-i}k$ locations. $W_i \in \mathbb{R}^{H_i W_i \times 4^{4-i} k}$ is obtained by applying the SoftMax operation to s_i. Finally, only a sparse matching to the selected locations from the memory is performed as follows:

$$y_i = \left[v^Q_i, \tilde{v}^M_i W^{\mathrm{T}}_i\right], i = \{1, 2, 3\}, \tag{9}$$

where $\tilde{v}^M_i \in \mathbb{R}^{\frac{C_i}{2} \times 4^{4-i} k}$ is constructed by sampling $4^{4-i}k$ samples for each query pixel from v^M_i. The output of the memory read $y_i \in \mathbb{R}^{C_i \times H_i W_i}$ is passed to the decoder to extract a mask prediction.

3.3.2 Decoder

We use the refinement module in [31] as the building block of our decoder. The output of the last stage memory read, i.e., y_4, is gradually upsampled with convolutional layers. The refinement module at each stage also takes the output of the top-k memory read at the corresponding scale through skip connections. The refinement module produces an object mask with the size $H_1 \times W_1$ $\left(= \frac{H}{4} \times \frac{W}{4}\right)$, which is bilinearly upsampled to the original resolution. The soft aggregation of the output masks [32] is applied when handling multiple objects.

3.4. Architecture Variants

We introduce four architecture variants of HST, i.e., HST-T, HST-S, HST-B, and HST-L, by using the following hyper-parameter settings.

- HST-T: $C = 96$, $L = \{2, 2, 6, 2\}$, $M = 7$
- HST-S: $C = 96$, $L = \{2, 2, 18, 2\}$, $M = 7$
- HST-B: $C = 128$, $L = \{2, 2, 18, 2\}$, $M = 12$
- HST-L: $C = 192$, $L = \{2, 2, 18, 2\}$, $M = 12$

where L is the number of block layer, M is window size. The image and video Transformers of the base model (HST-B) require 193.6 M parameters, and those for the rest three variants require approximately 0.25× (HST-T), 0.5× (HST-S), and 2× (HST-L) of the parameters, respectively.

4. Experiments

4.1. Implementation Details

Training. We followed the same training strategy as STM [32], HMMN [38], PCVOS [33]. We initialized the image Transformer blocks with ImageNet pre-trained

Table 1: Comparison on the DAVIS 2016 validation set. (+Y) indicates YouTube-VOS is additionally used for training, and OL denotes the use of online-learning strategies during test time. * denotes time measurements from the corresponding papers. † denotes the results obtained using the first and previous frames as input of the video Transformer.

Method	OL	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$	Time (s)
OSVOS [2]	✓	80.2	79.8	80.6	9*
MaskRNN [16]	✓	80.8	80.7	80.9	-
PReMVOS [29]	✓	86.8	84.9	88.6	30*
STM [32] (+Y)		89.3	88.7	89.9	0.10
KMN [37] (+Y)		90.5	89.5	91.5	-
HMMN [38] (+Y)		90.4	89.6	92.0	0.07
AOT [58] (+Y)		91.1	90.1	92.1	0.06
STCN [6] (+Y)		91.6	90.8	92.5	0.05
AOCVOS [55] (+Y)		91.6	88.5	94.7	0.32
PCVOS [33] (+Y)		91.9	90.8	93.0	0.11
QDMN [25] (+Y)		92.0	90.7	93.2	0.13
HST-T† (+Y)		92.1	91.0	93.1	0.11
HST-T (+Y)		92.9	92.6	93.2	0.21
HST-S† (+Y)		92.2	91.2	93.1	0.15
HST-S (+Y)		93.0	92.2	93.8	0.28
HST-B† (+Y)		93.1	91.9	94.3	0.24
HST-B (+Y)		94.0	93.2	94.8	0.36
HST-L† (+Y)		93.7	92.8	94.5	0.29
HST-L (+Y)		**94.2**	**93.4**	**95.0**	0.51

Table 2: Comparison on the DAVIS 2017 validation and test-dev set. (+Y) indicates YouTube-VOS is additionally used for training.

Methods	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$
Validation 2017 Split			
STM [32] (+Y)	81.8	79.2	84.3
SST [12] (+Y)	82.5	79.9	85.1
KMN [37] (+Y)	82.8	80.0	85.6
CFBI+ [59] (+Y)	82.9	80.1	85.7
AOCVOS [55] (+Y)	83.8	81.7	85.9
HMMN [38] (+Y)	84.7	81.9	87.5
AOT [58]	79.3	76.5	82.2
AOT [58] (+Y)	84.9	82.3	87.5
STCN [6] (+Y)	85.4	82.6	88.6
QDMN [25] (+Y)	85.6	82.5	88.6
PCVOS [33] (+Y)	**86.1**	**83.0**	89.2
HST-T (+Y)	83.6	80.9	86.2
HST-S (+Y)	84.0	80.7	87.3
HST-B	79.9	76.9	82.9
HST-B (+Y)	85.9	82.5	**89.2**
HST-L (+Y)	85.6	82.2	89.0
Testing 2017 Split			
STM [32] (+Y)	72.2	69.3	75.2
CFBI [57] (+Y)	74.8	71.1	78.5
KMN [37] (+Y)	77.2	74.1	80.3
CFBI+ [59]] (+Y)	78.0	74.4	81.6
HMMN [38] (+Y)	78.6	74.7	82.5
STCN [6] (+Y)	77.8	74.3	81.3
AOCVOS [55] (+Y)	79.3	74.7	83.9
AOT [58] (+Y)	79.6	75.9	83.3
HST-T (+Y)	78.9	75.7	82.2
HST-S (+Y)	79.2	75.8	82.6
HST-B (+Y)	79.9	76.5	83.4
HST-L (+Y)	**80.2**	**76.8**	**83.6**

weights and randomly initialized the other layers. Because the Video Transformer blocks take additional masks as input, they cannot be simply pre-trained using video datasets. Therefore, we initialized the Video Transformer blocks by replicating the image Transformer block's ImageNet pre-trained weights along the temporal dimension. Then, we pre-trained HST on the image datasets, including MSRA10K, ECSSD, PASCAL-S, PASCAL VOC2012, and COCO datasets [7, 13, 20, 24, 39]. For these image datasets, we synthesized three consecutive frames by augmenting each image via random affine transformations, including rotation, shearing, zooming, translation, and cropping.

After the pre-training on the synthesized image dataset, the main training was conducted using either DAVIS 2017 or YouTube-VOS 2019 training set, depending on the target benchmark. During the main training, three frames were randomly sampled from a video with a gradually increasing maximum interval (from 0 to 25). During both the pre-training and main training, we minimized the pixel-wise cross-entropy loss with Adam optimizer [21], and the learning rate was set to 1e-5. We used an input size of 384×384 and set P = 1 (temporal) and M = 4 (spatial). Following STM, we employed the soft aggregation when multiple target objects exist in a video [32].

Inference. We used the first, previous, and intermediate frames sampled at every eight frames as input for the video Transformer. We used the same number of $k = 128$ for top-k guided memory matching during the training and inference. We measured the run-time of our and compared methods using two NVIDIA RTX 3090 GPUs.

4.2. Comparisons

We compared our HST with state-of-the-art methods on the DAVIS [35, 36] and YouTube-VOS [54] benchmarks. For the DAVIS benchmark, 60 videos from the DAVIS 2017 training set were used for the main training, following the standard protocol. In addition, we report our results on the DAVIS benchmark using additional training videos from Youtube-VOS for a fair comparison with several recent methods. For the Youtube-VOS benchmark, 3471 videos in the training set were used for training.

DAVIS is a densely annotated VOS dataset and the most

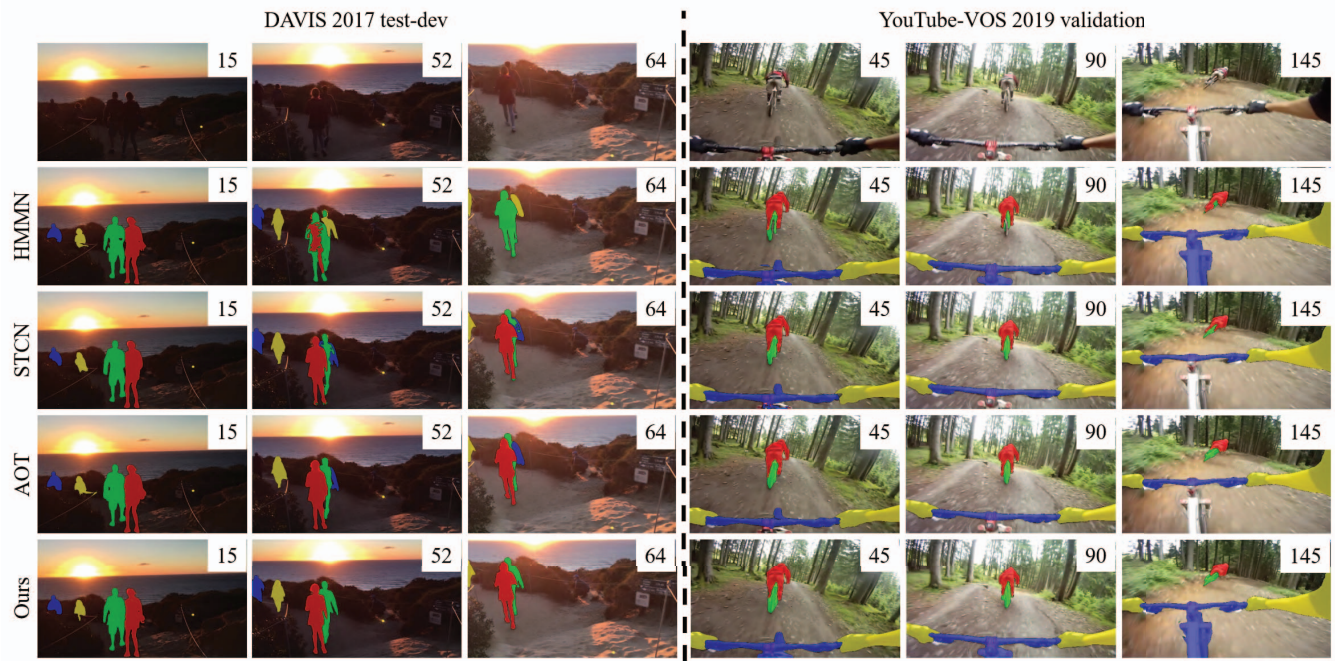

Figure 3: Qualitative performance comparison of HST with HMMN [38], STCN [6], and AOT [58].

widely-used benchmark to evaluate VOS techniques. The DAVIS dataset consists of two sets: (1) DAVIS 2016, which is an object-level annotated dataset (single object); and (2) DAVIS 2017, which is an instance-level annotated dataset (multiple objects). The official metrics, i.e., region similarity $\mathcal{J}$ and contour accuracy $\mathcal{F}$, were measured for comparison. To evaluate HST, we used an input size of 480p resolution. As shown in Table 1, HST-B outperforms the second-best method by 2.0% $\mathcal{J}\&\mathcal{F}$ on the DAVIS 2016 validation set. Furthermore, additional experiments were conducted using the first and previous frames as input of the video Transformer to test the trade-off between the processing time and segmentation accuracy. We also conducted comparisons on the DAVIS 2017 validation and test-dev sets, and the results are given in Table 2. Our HST-B showed the competitive performance to PCVOS [33] on the DAVIS 2017 validation set and achieved state-of-the-art performance on the DAVIS 2017 testing set. Our HST-B trained without using the YouTube-VOS training dataset still showed improved performance over the other models trained without using the YouTube-VOS training dataset.

YouTube-VOS is a large-scale benchmark for VOS. To evaluate our HST on the YouTube-VOS benchmark, we used an input size of 480p resolution. We measured the region similarity ($\mathcal{J}_{\mathcal{S}}$, $\mathcal{J}_{\mathcal{U}}$) and contour accuracy ($\mathcal{F}_{\mathcal{U}}$, $\mathcal{F}_{\mathcal{U}}$) for 65 seen and 26 unseen object categories separately. Table 3 shows the performance comparison of HST with state-of-the-art methods on the YouTube-VOS 2018 and 2019 validation sets, demonstrating that HST-B surpasses the state-of-the-art methods in both seen and unseen object categories. Figure 3 shows qualitative performance comparison with HMMN [38], STCN [6], and AOT [58]. HMMN [38] failed in separating multiple occluded objects. STCN [6] and AOT [58] produced incorrect results for incoming or outgoing objects in the scene. On the other hand, HST predicted target objects accurately in these challenging scenarios. More results are provided in the supplementary material.

4.3. Ablation Experiments

We conducted ablation studies using HST-B on the DAVIS 2017 dataset. More details about the models used for the ablation studies are provided in the supplementary material.

Pre-training. As shown in Table 4.1, the model pre-trained on the image datasets performed favorably with 74.2 % $\mathcal{J}\&\mathcal{F}$. Due to the effectiveness of the pre-training, the fully trained model exhibited 3.1 % higher $\mathcal{J}\&\mathcal{F}$ than the model trained on the DAVIS 2017 training dataset only. Furthermore, it shows competitive performance without using synthesized static datasets.

Memory management. As a default setting, HST uses the first, previous, and intermediate frames sampled at every eight frames as input for the video Transformer. Table 4.2 shows that HST performed reasonably well with 84.9 % $\mathcal{J}\&\mathcal{F}$ when only the first and previous frames were used as input. In environments where memory is scarce, it is advised to use only these two frames as input.

Table 3: Quantitative evaluation on the YouTube-VOS validation set

Methods	$\mathcal{J}\&\mathcal{F}$	Seen $\mathcal{J}_S$	Seen $\mathcal{F}_S$	Unseen $\mathcal{J}_U$	Unseen $\mathcal{F}_U$
Validation 2018 Split					
STM [32]	79.4	79.7	84.2	72.8	80.9
KMN [37]	81.4	81.4	85.6	75.3	83.3
SST [12]	81.7	81.2	-	76.0	-
MaskVOS [47]	81.9	81.4	86.6	75.9	83.9
CFBI+ [59]	82.8	81.8	86.6	77.1	85.6
HMMN [38]	82.6	82.1	87.0	76.8	84.6
STCN [6]	83.0	81.9	86.5	77.9	85.7
QDMN [25]	83.8	82.7	87.5	78.4	86.4
AOCVOS [55]	84.0	83.2	87.8	79.3	87.3
AOT [58]	84.1	83.7	88.5	78.1	86.1
PCVOS [33]	84.6	83.0	88.0	79.6	87.9
HST-T	83.2	82.7	86.8	78.2	85.1
HST-S	83.9	83.4	87.0	78.4	86.8
HST-B	85.0	84.3	**89.2**	79.0	87.6
HST-L	**85.1**	**84.4**	89.1	**79.2**	**87.8**
Validation 2019 Split					
SST [12]	81.8	80.9	-	76.6	-
CFBI+ [59]	82.6	81.7	86.2	77.1	85.2
HMMN [38]	82.6	82.1	87.0	77.3	85.0
STCN [6]	82.7	81.1	85.4	78.2	85.9
AOT [58]	84.1	83.5	88.1	78.4	86.3
AOCVOS [55]	84.1	82.7	87.1	80.0	87.8
PCVOS [33]	84.6	82.6	87.3	80.0	88.3
HST-T	83.5	82.9	87.4	78.2	85.5
HST-S	84.1	83.3	88.3	78.0	86.7
HST-B	84.9	83.6	**88.5**	79.5	88.1
HST-L	**85.0**	**83.7**	88.3	**79.7**	**88.3**

Table 4: Ablation studies for HST-B on the DAVIS 2017 validation set. DM: Dense matching

Ablation	Method	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$
1. Effect of pre-training				
Training	Pre.	74.2	70.4	76.3
	Main	82.8	80.0	85.6
	Full	85.9	82.5	89.2
2. Comparison of memory management strategies				
Memory frames	First & prev.	84.9	81.6	88.2
	+ Every 8 frames	85.9	82.5	89.2
3. Effect on hierarchical memory read				
Memory read	Last stage only	83.5	80.3	86.7
	All stages w/ top-k	85.9	82.5	89.2
	All stages w/ DM	86.4	83.6	89.7
4. Effect on utilization of other object masks				
Mask	w/o other object mask	84.1	81.2	86.9
	w/ other object mask	85.9	82.5	89.2
5. Effect on spatiotemporal feature				
Feature	Image feature only	83.0	79.9	86.1
	Image and video features	85.9	82.5	89.2

Hierarchical memory read. To show the effectiveness of using multi-scale features for the memory read, we obtained the result using the output of the memory read at the last stage only, i.e., y_4, as input for the decoder. As shown in Table 4.3, the performance decreased significantly, demonstrating the necessity of multi-scale features for precise mask decoding. In addition, when our hierarchical top-k read was replaced by naive dense matching, we obtained a slightly better performance of 86.4 % $\mathcal{J}\&\mathcal{F}$. However, the dense matching at all stages required an average processing time of 2.78 s per frame, where the top-k matching consumed 0.42 s per frame.

Mask utilization. Our video Transformer takes given or predicted masks as input in addition to video frames. To better handle multiple object segmentation, we used the common strategy [6, 32, 37, 38] of including a binary object mask of other objects as additional input. Table 4.4 shows that the information on the other objects contributed to 1.8 % $\mathcal{J}\&\mathcal{F}$ improvement.

Video Transformer. To demonstrate the effectiveness of using both image and video Transformers for spatiotemporal feature extraction, we built a model by applying only image Transformer to the current and past frames for feature extraction. Table 4.5 shows that both image and video Transformers are essential for extracting spatiotemporal features, leading to 2.9 % $\mathcal{J}\&\mathcal{F}$ improvement.

5. Conclusions

In this paper, we presented a novel VOS framework called HST that exploits image and video Transformers as a means of spatiotemporal feature extraction from a video. To take full advantage of image and video Transformers, we used image and video features as a query and memory, respectively, and matched them at multiple scales with efficient hierarchical memory read operations. HST showed state-of-the-art performance in several benchmarks, including the DAVIS 2016 and 2017 validation sets and YouTube-VOS 2018 and 2019 validation datasets. Considering the conciseness and technical advantages of HST, we hope our work can motivate future VOS studies.

Acknowlegement

This work is supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2022R1A2C2002810).

References

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. 2014.

[2] 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/CVF Conference on Computer Vision and Pattern Recognition*, pages 221–230, 2017.

[3] Xiaochen Cai, Hengxing Cai, Boqing Zhu, Kele Xu, Weiwei Tu, and Dawei Feng. Multiple temporal fusion based weakly-supervised pre-training techniques for video categorization. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 7089–7093, 2022.

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proceedings of the European Conference on Computer Vision*, pages 213–229, 2020.

[5] Ho Kei Cheng, Yu-Wing Tai, and Chi-Keung Tang. Modular interactive video object segmentation: Interaction-to-mask, propagation and difference-aware fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5559–5568, 2021.

[6] Ho Kei Cheng, Yu-Wing Tai, and Chi-Keung Tang. Rethinking space-time networks with improved memory coverage for efficient video object segmentation. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 11781–11794, 2021.

[7] Ming-Ming Cheng, Niloy J Mitra, Xiaolei Huang, Philip HS Torr, and Shi-Min Hu. Global contrast based salient region detection. volume 37, pages 569–582. IEEE, 2014.

[8] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. 2014.

[9] Ye Deng, Siqi Hui, Sanping Zhou, Deyu Meng, and Jinjun Wang. Learning contextual transformer network for image inpainting. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 2529–2538, 2021.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. 2018.

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. 2020.

[12] Brendan Duke, Abdalla Ahmed, Christian Wolf, Parham Aarabi, and Graham W. Taylor. SSTVOS: Sparse spatiotemporal transformers for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5912–5921, 2021.

[13] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. volume 88, pages 303–338. Springer, 2010.

[14] Mingqi Gao, Feng Zheng, James JQ Yu, Caifeng Shan, Guiguang Ding, and Jungong Han. Deep learning for video object segmentation: a review. pages 1–75, 2022.

[15] Li Hu, Peng Zhang, Bang Zhang, Pan Pan, Yinghui Xu, and Rong Jin. Learning position and target consistency for memory-based video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4144–4154, 2021.

[16] Yuan-Ting Hu, Jia-Bin Huang, and Alexander Schwing. MaskRNN: Instance level video object segmentation. In *Proceedings of the Advances in Neural Information Processing Systems*, 2017.

[17] Yuan-Ting Hu, Jia-Bin Huang, and Alexander G. Schwing. VideoMatch: Matching based video object segmentation. In *Proceedings of the European Conference on Computer Vision*, pages 54–70, 2018.

[18] Muqi Huang and Lefei Zhang. Atrous pyramid transformer with spectral convolution for image inpainting. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 4674–4683, 2022.

[19] Xuhua Huang, Jiarui Xu, Yu-Wing Tai, and Chi-Keung Tang. Fast video object segmentation with temporal aggregation network and dynamic template matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8879–8889, 2020.

[20] Huaizu Jiang, Jingdong Wang, Zejian Yuan, Yang Wu, Nanning Zheng, and Shipeng Li. Salient object detection: A discriminative regional feature integration approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2083–2090, 2013.

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2014.

[22] Yu Li, Zhuoran Shen, and Ying Shan. Fast video object segmentation using the global context module. In *Proceedings of the European Conference on Computer Vision*, pages 735–750, 2020.

[23] Yongqing Liang, Xin Li, Navid Jafari, and Jim Chen. Video object segmentation with adaptive feature bank and uncertain-region refinement. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 3430–3441, 2020.

[24] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Proceedings of the European Conference on Computer Vision*, pages 740–755, 2014.

[25] Yong Liu, Ran Yu, Fei Yin, Xinyuan Zhao, Wei Zhao, Weihao Xia, and Yujiu Yang. Learning quality-aware dynamic memory for video object segmentation. In *European Conference on Computer Vision*, pages 468–486. Springer, 2022.

[26] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

[27] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *Pro-*

ceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 3202–3211, 2022.

[28] Xiankai Lu, Wenguan Wang, Martin Danelljan, Tianfei Zhou, Jianbing Shen, and Luc Van Gool. Video object segmentation with episodic graph memory networks. In *Proceedings of the European Conference on Computer Vision*, pages 661–679, 2020.

[29] Jonathon Luiten, Paul Voigtlaender, and Bastian Leibe. PReMVOS: Proposal-generation, refinement and merging for video object segmentation. In *Proceedings of the Asian Conference on Computer Vision*, pages 565–580, 2018.

[30] Jianbiao Mei, Mengmeng Wang, Yeneng Lin, Yi Yuan, and Yong Liu. TransVOS: Video object segmentation with transformers. 2021.

[31] Seoung Wug Oh, Joon-Young Lee, Kalyan Sunkavalli, and Seon Joo Kim. Fast video object segmentation by reference-guided mask propagation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7376–7385, 2018.

[32] 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/CVF International Conference on Computer Vision*, pages 9226–9235, 2019.

[33] Kwanyong Park, Sanghyun Woo, Seoung Wug Oh, In So Kweon, and Joon-Young Lee. Per-clip video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1352–1361, 2022.

[34] Federico Perazzi, Anna Khoreva, Rodrigo Benenson, Bernt Schiele, and Alexander Sorkine-Hornung. Learning video object segmentation from static images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2663–2672, 2017.

[35] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 724–732, 2016.

[36] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 DAVIS challenge on video object segmentation. 2017.

[37] Hongje Seong, Junhyuk Hyun, and Euntai Kim. Kernelized memory network for video object segmentation. In *Proceedings of the European Conference on Computer Vision*, pages 629–645, 2020.

[38] Hongje Seong, Seoung Wug Oh, Joon-Young Lee, Seongwon Lee, Suhyeon Lee, and Euntai Kim. Hierarchical memory matching network for video object segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12889–12898, 2021.

[39] Jianping Shi, Qiong Yan, Li Xu, and Jiaya Jia. Hierarchical image saliency detection on extended CSSD. volume 38, pages 717–729. IEEE, 2015.

[40] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7262–7272, 2021.

[41] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Proceedings of the Advances in Neural Information Processing Systems*, 2014.

[42] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *Proceedings of the International Conference on Machine Learning*, pages 10347–10357, 2021.

[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 *Proceedings of the Advances in Neural Information Processing Systems*, 2017.

[44] Paul Voigtlaender, Yuning Chai, Florian Schroff, Hartwig Adam, Bastian Leibe, and Liang-Chieh Chen. FEELVOS: Fast end-to-end embedding learning for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9481–9490, 2019.

[45] Paul Voigtlaender and Bastian Leibe. Online adaptation of convolutional neural networks for video object segmentation. 2017.

[46] Haochen Wang, Xiaolong Jiang, Haibing Ren, Yao Hu, and Song Bai. SwiftNet: Real-time video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1296–1305, 2021.

[47] Mengmeng Wang, Jianbiao Mei, Lina Liu, Guanzhong Tian, Yong Liu, and Zaisheng Pan. Delving deeper into mask utilization in video object segmentation. volume 31, pages 6255–6266. IEEE, 2022.

[48] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8741–8750, 2021.

[49] Ziqin Wang, Jun Xu, Li Liu, Fan Zhu, and Ling Shao. RANet: Ranking attention network for fast video object segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3978–3987, 2019.

[50] Peisong Wen, Ruolin Yang, Qianqian Xu, Chen Qian, Qingming Huang, Runmin Cong, and Jianlou Si. Dmvos: Discriminative matching for real-time video object segmentation. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2048–2056, 2020.

[51] Huaxin Xiao, Jiashi Feng, Guosheng Lin, Yu Liu, and Maojun Zhang. MoNet: Deep motion exploitation for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1140–1148, 2018.

[52] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. SegFormer: Simple and efficient design for semantic segmentation with transformers. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 12077–12090, 2021.

[53] Haozhe Xie, Hongxun Yao, Shangchen Zhou, Shengping Zhang, and Wenxiu Sun. Efficient regional memory network for video object segmentation. In *Proceedings of*

the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 1286–1295, 2021.

[54] 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*, pages 585–601, 2018.

[55] Xiaohao Xu, Jinglu Wang, Xiang Ming, and Yan Lu. Towards robust video object segmentation with adaptive object calibration. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 2709–2718, 2022.

[56] Linjie Yang, Yanran Wang, Xuehan Xiong, Jianchao Yang, and Aggelos K. Katsaggelos. Efficient video object segmentation via network modulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6499–6507, 2018.

[57] Zongxin Yang, Yunchao Wei, and Yi Yang. Collaborative video object segmentation by foreground-background integration. In *European Conference on Computer Vision*, pages 332–348, 2020.

[58] Zongxin Yang, Yunchao Wei, and Yi Yang. Associating objects with transformers for video object segmentation. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 2491–2502, 2021.

[59] Zongxin Yang, Yunchao Wei, and Yi Yang. Collaborative video object segmentation by multi-scale foreground-background integration. IEEE, 2021.

[60] Rui Yao, Guosheng Lin, Shixiong Xia, Jiaqi Zhao, and Yong Zhou. Video object segmentation and tracking: A survey. volume 11, pages 1–47. ACM New York, NY, USA, 2020.