Applied Intelligence (2024) 54:6138–6153
https://doi.org/10.1007/s10489-024-05486-y

Learning spatiotemporal relationships with a unified framework for video object segmentation

Jianbiao Mei[1] · Mengmeng Wang[1] · Yu Yang[1] · Zizhang Li[1] · Yong Liu[1]

Accepted: 24 April 2024 / Published online: 7 May 2024
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2024

Abstract

Video object segmentation (VOS) has made significant progress with matching-based methods, but most approaches still show two problems. Firstly, they apply a complicated and redundant two-extractor pipeline to use more reference frames for cues, increasing the models' parameters and complexity. Secondly, most of these methods neglect the spatial relationships (inside each frame) and do not fully model the temporal relationships (among different frames), i.e., they need adequate modeling of spatial-temporal relationships. In this paper, to address the two problems, we propose a unified transformer-based framework for VOS, a compact and unified single-extractor pipeline with strong spatial and temporal interaction ability. Specifically, to slim the common-used two-extractor pipeline while keeping the model's effectiveness and flexibility, we design a single dynamic feature extractor with an ingenious dynamic input adapter to encode two significant inputs, i.e., reference sets (historical frames with predicted masks) and query frame (current frame), respectively. Moreover, the relationships among different frames and inside every frame are crucial for this task. We introduce a vision transformer to exploit and model both the temporal and spatial relationships simultaneously. By the cascaded design of the proposed dynamic feature extractor, transformer-based relationship module, and target-enhanced segmentation, our model implements a unified and compact pipeline for VOS. Extensive experiments demonstrate the superiority of our model over state-of-the-art methods on both DAVIS and YouTube-VOS datasets. We also explore potential solutions, such as sequence organizers, to improve the model's efficiency. On DAVIS17 validation, we achieve ∼50% faster inference speed with only a slight 0.2% ($J\&F$) drop in segmentation quality. Codes are available at https://github.com/sallymmx/TransVOS.git.

Keywords Video object segmentation · Transformer · Spatial-temporal · Dynamic feature extractor · Sequence organizers

1 Introduction

Video Object Segmentation (VOS), as a fundamental task in the computer vision community, has attracted more and more attention in recent years due to its potential application in autonomous driving, object tracking [3, 12, 13], activity recognition [55], and video editing, etc. In this paper, we focus on semi-supervised VOS, which provides the target objects' masks in the first frame, and the algorithms should produce the segmentation masks for those objects in the subsequent frames. Under this setting, VOS remains challenging due to object occlusion, deformation, appearance variation, and similar object confusion in video sequences.

When processing videos in sequential order, the natural idea is to use more reference/historical frames that contain abundant temporal information. Recently, state-of-the-art VOS performance has been achieved by matching-based algorithms [5, 11, 22, 27, 32, 39, 42, 43, 45, 48], while most of which still show two problems like complicated and redundant pipeline and inadequately modeling of spatial-temporal relationships when referring to historical frames for segmenting.

✉ Yong Liu
yongliu@iipc.zju.edu.cn

Jianbiao Mei
jianbiaomei@zju.edu.cn

Mengmeng Wang
mengmengwang@zju.edu.cn

Yu Yang
yu.yang@zju.edu.cn

Zizhang Li
zzli@zju.edu.cn

[1] Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou 310027, People's Republic of China

First, many existing methods [5, 15, 19, 22, 27, 32, 42, 48] usually use a complicated and redundant two-extractor pipeline, in which the query extractor encodes features of the current frame and the memory/reference extractor encodes historical information from reference frames. This two-extractor pipeline is flexible for encoding reference sets of different sizes; however, containing many redundant parameters and increasing the model's complexity. Siamese architecture effectively reduces the number of parameters and simplifies the complicated pipeline, while existing ways [14, 16, 23, 31, 47] are of limited use and unable to keep the flexibility and effectiveness. For example, the segmentation masks' abundant edges and contour features are not fully leveraged by directly concatenating the predicted masks with high-level semantic features. In addition, concatenating the previous frame's mask with the query frame may bring significant displacement shifting errors. However, using optical flow to warp the mask is time-consuming.

Second, these methods mostly neglect the spatial relationships (inside each frame) and do not fully model the temporal relationships (among different frames). However, the spatial and temporal relationship is crucial for learning the robust target appearance across frames and handling practical scenarios such as object occlusion, deformation, and appearance variation. To better depict this point, we define two relationships in this paper, i.e., **temporal relationships** (Fig. 1 (b)) and **spatial relationships** (Fig. 1 (c)). The former is the relationships among pixels in different frames, representing the correspondence of target objects among past and current frames, which is vital for learning robust global target object features and helps handle appearance change across frames. The latter represents the relationships among pixels in a specific frame, including object appearance information for target localization and segmentation, which helps obtain accurate mask boundaries and is essential for learning local target object structure as explored in [25]. A group of matching-based methods [5, 11, 15, 19, 22, 27, 32, 37, 39, 42, 48, 53] provide partial solutions for capturing above correspondence and achieve competitive performance. Among them, the Space-Time Memory (STM) based approaches [5, 15, 19, 22, 27, 32, 42, 48] have achieved great success. The basic idea of these methods is to compute the similarities of target objects between the current and past frames by feature matching. However, as illustrated in Fig. 1(a), most of these methods only compute attentions among pixels in the query frame against pixels in each reference frame, ignoring the temporal dependency among historical frames and spatial correlations of pixels inside a specific frame. There are a few methods that pay attention to these issues. For instance, EGMN [27] proposes a fully-connected graph to capture cross-frame correlation, effectively exploiting the temporal relationships. However, EGMN still omits spatial relationships.

To address the above two problems, in this paper, we propose a new framework for VOS, which is a compact and unified single-extractor pipeline and has strong spatial and temporal interaction ability.

Specifically, to represent the reference sets and query frames in a unified way, we develop a plain yet effective feature extractor that has a dynamic input adapter and accepts the reference sets and the query frames in the meantime, significantly simplifying the existing VOS framework while keeping the effectiveness and flexibility. It is based on the assumption that the convolution network can be generic to different inputs of visual patterns. Therefore, we designed the dynamic input adapter to encode the reference sets and the query frames to different visual patterns. And then, a convolution network is used to map these visual patterns into feature embedding. The dynamic input adapter uses different layers to encode different inputs in practice. For reference sets, the RGB image, the mask's foreground, and the mask's background are encoded and fused to enhance the target appearance. At the same time, only the RGB image is encoded for the query frames. By this means, the dynamic feature extractor can encode the inputs in a unified way and keeps the flexibility and effective-

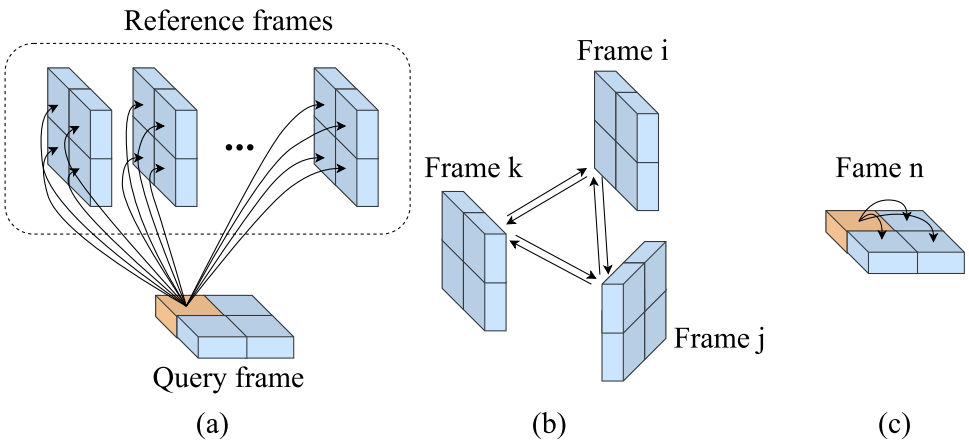

Fig. 1 (a) relationships between pixels in the query frame and pixels in reference frames. Temporal relationships (b) are relationships among pixels in different frames, representing the correspondence of target objects among reference and query frames. Spatial relationships (c) are relationships among pixels in a specific frame of reference and query frames, including object appearance information for target localization and segmentation

ness of the separate extractors but with a compact architecture.

Moreover, since the dependencies both among different frames and inside every frame are crucial for this task, we introduce the vision transformer to jointly capture the spatial and temporal relationships, generating discriminate spatial-temporal features for segmenting. Our model takes the features of the reference sets and the query frame as the input sequence and exploits the transformers to establish spatial-temporal dependency simultaneously. Also, we design a Target Attention Block (TAB) to extract the target's mask features from the query frame, helping obtain the target mask prediction from the outputs of the transformer. Above all, By the cascaded design of the proposed dynamic feature extractor, transformer-based relationship module, and target-enhanced segmentation, our model implements a unified and compact pipeline for VOS.

Finally, we explore potential solutions, such as sequence organizers, to improve the model's efficiency. Since the computational complexity of the self-attention mechanism is proportional to the square of the length of the input sequences. And not all pixels in the reference sets are important for the target segmentation of the query frame. Therefore, we design the sequence organizers to compress the redundant reference representation. By this means, compared to the vanilla model, we achieve ∼50% faster inference speed with only a slight 0.2% ($J\&F$) drop in segmentation quality on DAVIS17 validation, as shown in Section 4.5.

Our main contributions can be summarized as follows:

- We proposed a compact single-extractor framework to simplify the existing VOS pipeline. Specifically, we designed a dynamic feature extractor to present the two kinds of inputs, i.e., reference sets (history frames with predicted masks) and query frames (current frames), in a unified way, containing fewer parameters while keeping the effectiveness and flexibility of the two extractors.
- Considering that the dependencies among different frames and inside every frame are crucial for this task, we attach the vision transformer to the dynamic feature extractor, generating discriminate spatial-temporal features for segmenting. Our model is robust for appearance variation, occlusion, and confusion with sufficient spatial-temporal interaction.
- We comprehensively evaluate the proposed model on three benchmark datasets, including DAVIS 2016/2017 [34, 35] and YouTube-VOS [49]. The results demonstrate the effectiveness and efficiency of our method in comparison with the previous approaches. Also, we do extended experiments to explore how to improve our model's efficiency.

2 Related works

2.1 Tracking-based methods

These methods [3, 13, 41, 46] integrate object tracking techniques to indicate target location and spatial area for segmentation. SiamMask [46] adds a mask branch on SiamRPN [17] to narrow the gap between tracking and segmentation. FTAN-DTM [13] takes object segmentation as a sub-task of tracking, introducing the "tracking-by-detection" model into VOS. SAT [3] fuses object tracking and segmentation into a truly unified pipeline. It combines SiamFC++ [50] and proposes an estimation-feedback mechanism to switch between the mask box and tracking box, making segmentation and tracking tasks enhance each other. The integration of the tracker helps improve the inference speed, while the accuracy of tracking tasks often limits these methods' performance.

2.2 Matching-based methods

Recently, state-of-the-art performance has been achieved by matching-based methods [2, 11, 22, 27, 30, 32, 37, 39, 42, 43, 45, 48, 56], which perform feature matching to learn target object appearances offline. VideoMatch [11] measures similarity by soft matching with foreground and background features. FEELVOS [39] and CFBI [53] perform the nearest neighbor matching between the current frame and the first and previous frames in the feature space. STM [32] introduces an external memory to store past frames' features and uses the attention-based matching method to retrieve information from memory. KMN [37] applies Query-to-Memory matching with a kernelized memory read to reduce the non-locality of the STM. RMNet [48] proposes to replace STM's global-to-global matching with local-to-local matching to alleviate the ambiguity of similar objects. EGMN [27] organizes the memory network as a fully connected graph to store frames as nodes and capture cross-frame relationships by edges. SwiftNet [42] designed Pixel-Adaptive Memory to compress spatiotemporal redundancy. However, these methods do not fully utilize the spatial-temporal relationships among reference sets and query frames. In this paper, we introduce a vision transformer to model spatial-temporal dependency, which can help handle large object appearance changes.

2.3 Transformer-based methods

Recently, transformers have achieved great success in vision tasks like image classification [7], object detection [2], semantic segmentation [44], object tracking [51, 52], etc.

Due to the importance of spatial and temporal relationships for segmenting, we also employ the vision transformer in the VOS task, which is inspired by DETR [2]. Different from DETR and MaskFormer [4], which only model spatial relationships in a specific frame with the transformer, we fully exert the long-range dependencies modeling power of the transformer to simultaneously exploit spatial and temporal relationships among pixels of past frames and the current frame, which is vital and benefit to the VOS task. In addition, the proposed dynamic feature extractor and transformer complement each other and form a unified architecture. The former adaptively encodes two types of inputs, i.e., query frames and reference sets, while the latter effectively models two types of relationships among the input sequences.

There are also some transformer-based methods, SST [8], JOINT [29] and AOT [54]. SST uses the transformer's encoder with sparse attention to capture the spatial-temporal information among the current and preceding frames. However, mask representations are not explored in SST. JOINT combines inductive and transductive learning and extends the transduction branch to a transformer architecture. Nevertheless, its network structure is complicated. AOT proposed Identification Embedding that encodes all masks simultaneously and a Long Short-Term Transformer that captures spatial-temporal dependencies. AOT achieves good performance with fast inference speed, while short-term attention implies the temporal smoothness assumption, which may not be robust to fast-moving and small objects. Besides, SST, JOINT, and AOT did not employ the transformer's decoder and could not enjoy its substantial power.

2.4 Feature extractors

Reference sets (history frames with predicted masks) and query frames (current frames) are essential inputs for semi-supervised VOS. The former implies historical information, while the latter contains the appearance of the current target. Only one extractor was used to encode the inputs in the early years. MaskTrack [33] concatenated the query frame with the previous mask prediction as the input of a single ConvNet. After that, siamese architecture became popular for using two reference frames for cues. RGMP [31] and AGSS-VOS [23] concatenate the current frame with the previous frame's mask or warped mask to form a 4-channel input so as the reference sets. Then a shared extractor with a 4-channel input layer is used to extract features. For more temporal information, STM-based methods [5, 22, 27, 32, 42, 48] used two extractors, .i.e, a 4-channel memory/reference extractor and a query extractor to extract features from reference sets and the query frame, respectively. The two-extractor pipeline is flexible for encoding reference sets of different sizes while swollen and containing many redundant parameters. We argue that a more compact and flexible pipeline can be implemented with the proposed dynamic feature extractor.

3 Methods

The overview of our framework is illustrated in Fig. 3. It mainly consists of a dynamic feature extractor, a vision transformer, a target attention block, and a segmentation head. When segmenting a specific frame, we firstly use the dynamic feature extractor to extract the features of the current frame and reference sets. The outputs of the extractor are fed into a bottleneck layer to reduce the channel number. Then features are flattened before feeding into a vision transformer, which simultaneously models the temporal and spatial relationships. Moreover, the target attention block takes both the transformer's encoder and decoder's outputs as input and then outputs the feature maps, representing the target mask features. Finally, a segmentation head is attached after the target attention block to obtain the predicted object mask.

3.1 Dynamic feature extractor

As discussed in Section 1, we need a unified feature extractor that can effectively extract the features of the reference sets and the query frame and map them into an embedding space to be ready for feeding into the following vision transformer. In order to utilize more temporal cues, the existing pipelines mostly use two separate extractors to operate these two inputs, respectively, which are flexible and effective for encoding different sizes of reference sets but contain many redundant parameters and increase the model's complexity. Some methods [14, 31] apply siamese architecture to encode the two types of inputs, which is more lightweight and compact while showing obvious problems like insufficient use of mask features and significant shifted error caused by concatenating the previous mask with the current frame. To overcome the above issues, we combine both advantages, i.e., maintaining the effectiveness and flexibility of two-extractor pipelines while being more lightweight and compact like Siamese architecture, and design a dynamic feature extractor to represent the two types of inputs in a unified way. As demonstrated by the experiments in Table 4, compared to the two-extractor pipeline, our model equipped with the proposed dynamic feature extractor has much fewer parameters (about 20% reduction, "*Dynamic*" vs. "*Independent*") while maintaining effectiveness and flexibility.

Dynamic input adapter Specifically, we design a dynamic

input adapter to adaptively encode two types of inputs, i.e., query frames (RGB frames) and reference sets (the pairs of RGB frames with corresponding object masks). As shown in Fig. 2, when taking the RGB frames as input, it will go through the first path, which has one regular convolution operation. The second path will be used for reference sets, containing three convolutions to encode the RGB frame, the object's foreground mask, and the background mask. The output features of the three convolutions are added together to represent the reference sets. Our method can use arbitrary convolution networks as the feature extractor by replacing the first layer with the dynamic input adapter. Here we employ the first four stages of ResNet [10] as the feature extractor. After going through the dynamic input adapter, the features from the query frame and reference sets are first concatenated along the temporal dimension and then fed into the convolution network (CNN). Finally, the reference sets and current frame are mapped to feature maps $\mathbf{f} \in \mathbb{R}^{(T+1)\times C\times H\times W}$, where H, W, C are the height, width, and channel. T is the number of reference pairs.

Before feeding into the vision transformer, we use a 1x1 convolution layer to reduce the spatial channel of the feature maps from C to d ($d < C$), resulting in new feature maps $\mathbf{f}^{'} \in \mathbb{R}^{(T+1)\times d\times H\times W}$. Then, the spatial and temporal dimensions of $\mathbf{f}^{'}$ are flattened into one dimension, producing feature vectors $\mathbf{X} \in \mathbb{R}^{(T+1)HW\times d}$, which serves as the input of the transformer encoder.

3.2 Relationship modeling

We introduce the vision transformer to model the relationships of the two types of inputs, i.e., reference sets (history frames with predicted masks) and query frames (current frames), making the whole pipeline simple and modularized. Transformers have strong capabilities for modeling spatial-temporal relationships. First, the positional encoding explicitly introduces space-time position information, which could help the encoder model spatial-temporal relationships among pixels in the input frames. Second, the encoder could learn the target object's correspondence among the input frames and model the target object's structure in a specific frame. Third, the decoder could predict the spatial positions of the target objects in the query frame and focus on the most relevant object, which learns robust target representations for the target object and empowers our network to handle similar object confusion better.

Positional encoding The transformer's core component self-attention module is permutation invariant. However, both spatial and temporal positional information is vital for establishing spatial and temporal relationships and accurate object segmentation. Equipping with the space-time location information in feature maps, the encoder could better capture the spatial and temporal dependency among all elements in the input sequences, helping our network handle challenging situations like object occlusion and deformation. Therefore,

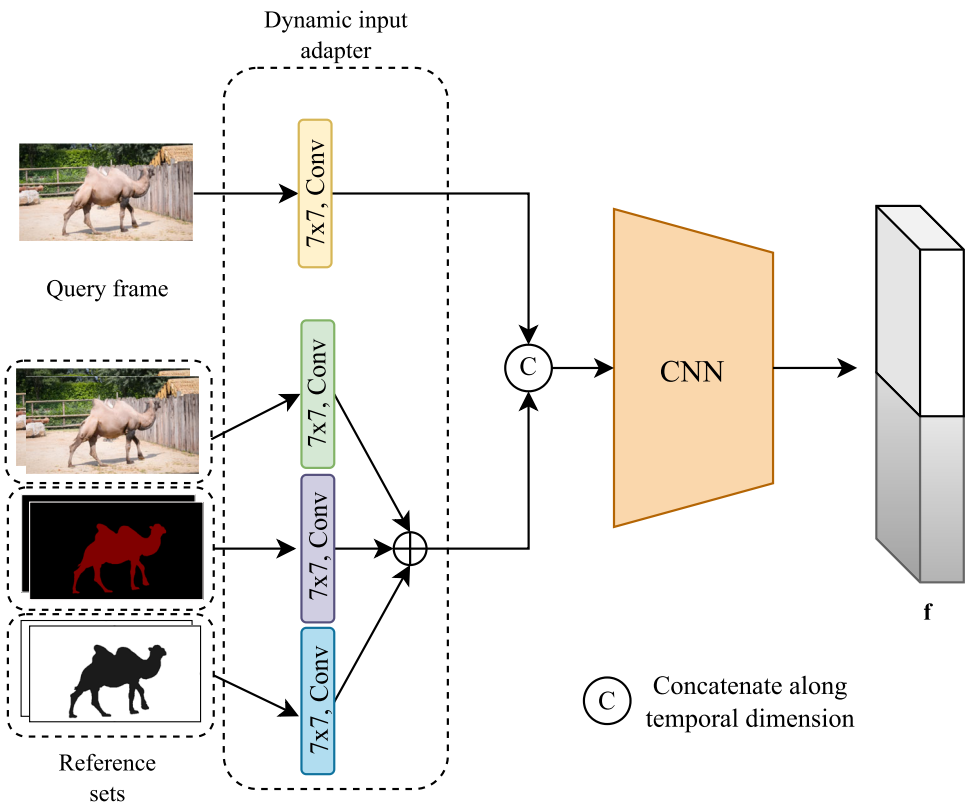

Fig. 2 The overall architecture of the proposed dynamic feature extractor. The feature extractor is used to extract the features of the current frame and reference sets in a unified way. "+", indicates adding operation. For a better view, we only illustrate two reference frames

explicitly embedding space-time position information into the transformer model is essential. We add sinusoidal positional encoding PE [38] to the embedded features $\mathbf{X}$ to form the inputs $\mathbf{Z}$ of the transformer. Mathematically,

$$\mathbf{Z} = \mathbf{X} + PE \tag{1}$$

$$PE(pos, 2i) = sin(pos/10000^{2i/d}) \tag{2}$$

$$PE(pos, 2i+1) = cos(pos/10000^{2i/d}) \tag{3}$$

where pos and i are the spatial-temporal position and the dimension of the features $\mathbf{X}$, respectively.

Transformer encoder The transformer encoder is used to model the spatial-temporal relationships among elements of the input sequences. It takes features $\mathbf{Z}$ as input and outputs encoded features $\mathbf{E}$. The encoder consists of L encoder layers, each with a standard architecture, including a multi-head self-attention module and a fully connected feed-forward network. The multi-head self-attention module captures spatial-temporal relationships from different representation sub-spaces.

At each encoder layer l, let $\mathbf{z}_{p,t}^{l-1} \in \mathbb{R}^d$ represents an element of representation $\mathbf{Z}^{l-1}(\mathbf{Z}^0 = \mathbf{Z}, \mathbf{Z}^L = \mathbf{E})$ encoded by the preceding encoder layer, where p and t denote the spatial and temporal position, respectively. For m-th attention head, the query/key/value vector $(\mathbf{q}_{p,t}^{l,m}/\mathbf{k}_{p,t}^{l,m}/\mathbf{v}_{p,t}^{l,m})$ from the element $\mathbf{z}_{p,t}^{l-1}$ is computed by:

$$\begin{aligned}\mathbf{q}_{p,t}^{l,m} &= \mathbf{W}_q^{l,m}\mathbf{z}_{p,t}^{l-1}\\ \mathbf{k}_{p,t}^{l,m} &= \mathbf{W}_k^{l,m}\mathbf{z}_{p,t}^{l-1}\\ \mathbf{v}_{p,t}^{l,m} &= \mathbf{W}_v^{l,m}\mathbf{z}_{p,t}^{l-1}\end{aligned} \tag{4}$$

The self-attention weights are computed by:

$$\boldsymbol{\alpha}_{p,t}^{l,m} = \sigma\left(\frac{\mathbf{q}_{p,t}^{l,m}}{\sqrt{d_m}} \cdot \left[\{\mathbf{k}_{p',t'}^{l,m}\}_{\substack{p'=1,\cdots,HW\\ t'=1,\cdots,T+1}}\right]\right) \tag{5}$$

Then, the multi-head self-attention feature is calculated by

$$\mathbf{s}_{p,t}^{l} = \sum_{m=1}^{M} \mathbf{W}_o^{l,m}\left[\sum_{t'=1}^{T+1}\sum_{p'=1}^{HW}(\boldsymbol{\alpha}_{p,t}^{l,m})_{p',t'} \cdot \mathbf{v}_{p',t'}^{l,m}\right] \tag{6}$$

where T represents the number of the reference frames, $\mathbf{W}_q^{l,m}, \mathbf{W}_k^{l,m}, \mathbf{W}_v^{l,m} \in \mathbb{R}^{d_m \times d}$ and $\mathbf{W}_o^{l,m} \in \mathbb{R}^{d \times d_m}$ are learnable weights ($d_m = d/M$ by default), σ indicates the $softmax$ function. Note that we compute attention along the spatial-temporal dimension. Thus we can model the spatial relationships and temporal relationships in the meanwhile.

After the multi-head self-attention module, residual connections and the layer normalization (LN) are used. Furthermore, features are further passed through a FFN attached with residual connections and layer normalization to acquire the output features $\mathbf{Z}^l$ of encoder layer l.

Transformer decoder The transformer decoder aims to focus on the most relevant object in the query frame and help predict the spatial positions of the target. It takes encoded features $\mathbf{E}$ and target query $\mathbf{x}_q$ as input and output decoded features $\mathbf{x}_o$. We only utilize one target query in the decoder to query the features of the specific target object. The decoder also consists of L decoder layers, each including a multi-head self-attention module, a multi-head cross-attention module, and a fully connected feed-forward network. The multi-head self-attention module in our model integrates the target information from different representation sub-spaces. Moreover, the multi-head cross-attention module is mainly leveraged to retrieve target object features from the encoder.

At each decoder layer l', let $\mathbf{x}^{l'-1}(\mathbf{x}^0 = \mathbf{x}_q, \mathbf{x}^L = \mathbf{x}_o) \in \mathbb{R}^d$ represents the representation extracted by the preceding decoder layer. The multi-head self-attention module is similar to that in the transformer encoder layer. Since there is only one target query, it calculates the attention weights against itself. Therefore, the computation of self-attention feature $\mathbf{x}_s^{l'}$ can be simplified as:

$$\mathbf{x}_s^{l'} = \sum_{m'=1}^{M} \mathbf{W}_{so}^{m'}(\mathbf{W}_{sv}^{m'}\mathbf{x}^{l'-1}) \tag{7}$$

where m' indexes the attention head in multi-head self-attention module, $\mathbf{W}_{sv}^{m'} \in \mathbb{R}^{d_{m'} \times d}$ and $\mathbf{W}_{so}^{m'} \in \mathbb{R}^{d \times d_{m'}}$ are learnable weights ($d_{m'} = d/M$ by default).

Then features $\hat{\mathbf{x}}_s^{l'}$ are passed through a multi-head cross-attention module after the residual connections and the layer normalization (LN):

$$\hat{\mathbf{x}}_s^{l'} = \mathrm{LN}(\mathbf{x}_s^{l'} + \mathbf{x}^{l'-1}) \tag{8}$$

Let $\mathbf{e}_{p,t} \in \mathbb{R}^d$ represents an element of $\mathbf{E}$, p and t denote the spatial and temporal position, respectively. For m'-th ($m' \leq M$) attention head in multi-head cross-attention module, the key and value vectors $\mathbf{k}_{p,t}^{l',m'}$, $\mathbf{v}_{p,t}^{l',m'}$ are computed as:

$$\begin{aligned}\mathbf{k}_{p,t}^{l',m'} &= \mathbf{W}_k^{l',m'}\mathbf{e}_{p,t}\\ \mathbf{v}_{p,t}^{l',m'} &= \mathbf{W}_v^{l',m'}\mathbf{e}_{p,t}\end{aligned} \tag{9}$$

The cross-attention weights are computed by:

$$\boldsymbol{\alpha}_s^{l',m'} = \sigma\left(\frac{\mathbf{W}_q^{l',m'}\hat{\mathbf{x}}_s^{l'}}{\sqrt{d_{m'}}} \cdot \left[\{\mathbf{k}_{p',t'}^{l',m'}\}_{\substack{p'=1,\cdots,HW\\ t'=1,\cdots,T+1}}\right]\right) \tag{10}$$

Then the cross-attention feature $\mathbf{x}_c^{l'}$ is calculated by:

$$\mathbf{x}_c^{l'} = \sum_{m'=1}^{M} \mathbf{W}_o^{l',m'} [\sum_{t=1}^{T+1} \sum_{p=1}^{HW} (\boldsymbol{\alpha}_s^{l',m'})_{p,t} \cdot \mathbf{v}_{p,t}^{l',m'}] \tag{11}$$

where T denotes the size of the reference set, $\mathbf{W}_q^{l',m'}$, $\mathbf{W}_k^{l',m'}$, $\mathbf{W}_v^{l',m'} \in \mathbb{R}^{d_{m'} \times d}$ and $\mathbf{W}_o^{l',m'} \in \mathbb{R}^{d \times d_{m'}}$ are learnable weights($d_{m'} = d/M$ by default). σ indicates the $softmax$ function.

Similar to the encoder layer, residual connections and the layer normalization (LN) are used after the multi-head cross-attention module. Moreover, features are further passed through a FFN attached with residual connections and layer normalization to acquire the output features $\mathbf{x}^{l'}$ of decoder layer l'.

3.3 Segmentation

Target attention block. To obtain the target mask prediction from the outputs of the transformer, the model needs to extract the target's mask features from the query frame. We design a Target Attention Block (TAB) to achieve this goal. TAB computes the attentions between the query frame's features $\mathbf{E}_Q$ in features $\mathbf{E}$, and the output features $\mathbf{x}_o$ from the decoder. $\mathbf{x}_o$ and $\mathbf{E}_Q$ are fed into a multi-head self-attention module (with M head) to obtain the attention maps, which boost the features of foreground and suppress the disturbance of the background. We concatenate the attention maps with $\mathbf{E}_Q$ as the input $\mathbf{S}$ of the following segmentation head to enhance the target features. The above procedure can be formulated as follows:

$$Attn_i(\mathbf{x}_o, \mathbf{E}_Q) = \sigma(\frac{(\mathbf{W}_q^i \mathbf{x}_o)^T (\mathbf{W}_k^i \mathbf{E}_Q)}{\sqrt{d_i}}) \tag{12}$$

$$\mathbf{S} = [\mathbf{E}_Q, Attn_1(\mathbf{x}_o, \mathbf{E}_Q), \cdots, Attn_M(\mathbf{x}_o, \mathbf{E}_Q)] \tag{13}$$

where i indexes the attention head, $\mathbf{W}_q^i, \mathbf{W}_k^i \in \mathbb{R}^{d_i \times d}$, are learnable weights ($d_i = d/M$ by default).

Segmentation head The features $\mathbf{S}$ are fed into a segmentation head which outputs the final mask prediction. Here, we use the refine module used in [31, 32] as the building block to construct our segmentation head. It consists of two blocks, each taking the previous stage's output and the current frame's feature maps from the feature extractor at the corresponding scale through skip connections. The refine module upscales the compressed feature maps by a factor of two at a time. Then a 2-channel convolution and a $softmax$ operation are attached behind blocks to attain the predicted mask in 1/4 scale of the input image. Finally, we use bi-linear interpolation to upscale the predicted mask to the original scale.

Multi-object segmentation Our framework can be extended to multi-object segmentation easily. Specifically, the network first predicts the mask for every target object. Then, a soft aggregation operation is used to merge all the predicted maps. We apply this way during both the training and inference to keep both stages consistent. For each location l in predicted mask $\mathbf{M}_i$ of object $i (i < N)$, the probability $p_{l,i}$ after soft aggression operation can be expressed as:

$$p_{l,i} = \sigma(\text{logit}(\hat{p}_{l,i})) = \frac{\hat{p}_{l,i}/(1 - \hat{p}_{l,i})}{\sum_{j=0}^{N-1} \hat{p}_{l,j}/(1 - \hat{p}_{l,j})} \tag{14}$$

where N is the number of objects. σ and logit represent the $softmax$ and $logit$ functions, respectively. The probability of the background is obtained by subtracting from 1 after computing the output of the merged foreground.

3.4 Training and inference

Training Our proposed model only requires short training video clips since it has no temporal smoothness assumptions. Nonetheless, our model can still learn long-term dependency. Just like most STM-based methods [19, 22, 27, 32, 37], we synthesize video clips by applying data augmentations (random affine, color, flip, resize, and crop) on a static image of datasets [6, 9, 18, 24]. Then we use the synthetic videos to pretrain our model. This pre-training procedure helps our model be robust against various object appearances and categories. After that, we train our model on real videos. We randomly select T frames from a video sequence of DAVIS [34, 35] or YouTube-VOS [49] and apply data augmentation on those frames to form a training video clip. By doing so, we can expect our model to learn long-range spatial-temporal information. We add cross-entropy loss $\mathcal{L}_{cls}$ and mask IoU loss $\mathcal{L}_{IoU}$ as the multi-object training loss $\mathcal{L}$, which can be expressed as:

$$\mathcal{L} = \frac{1}{N} \sum_{i=0}^{N-1} [\mathcal{L}_{cls}(\mathbf{M}_i, \mathbf{Y}_i) + \mathcal{L}_{IoU}(\mathbf{M}_i, \mathbf{Y}_i)] \tag{15}$$

$$\mathcal{L}_{cls}(\mathbf{M}_i, \mathbf{Y}_i) = -\frac{1}{|\Omega|} \sum_{p \in \Omega} [\mathbf{Y}_i \log(\frac{\exp(\mathbf{M}_i)}{\sum_{j=0}^{N-1} (\exp(\mathbf{M}_j))})]_p \tag{16}$$

$$\mathcal{L}_{IoU}(\mathbf{M}_i, \mathbf{Y}_i) = 1 - \frac{\sum_{p \in \Omega} \min(\mathbf{Y}_i^p, \mathbf{M}_i^p)}{\sum_{p \in \Omega} \max(\mathbf{Y}_i^p, \mathbf{M}_i^p)} \tag{17}$$

where Ω denotes the set of all pixels in the object mask, $\mathbf{M}_i, \mathbf{Y}_i$ represent the predicted mask and ground truth of object i, N is the number of objects. Note that N is set to 1 when segmenting a single object.

Inference Our model uses past frames with corresponding predicted masks to segment the current frame during the online reference phase. To balance the inference speed and accuracy, we do not use external memory to store every past frame's features but only use the first frame with ground truth and the previous frame with its predicted masks as the reference sets. Because the former always provides the most reliable information, and the latter is the most similar to the current frame. Note that our model is flexible and can use more reference frames to obtain more historical information for segmenting the current frame.

4 Experiments

In this section, we first introduce the implementation details of our approach and the datasets, and the evaluation metrics. Then we perform extensive experiments to demonstrate that our model consistently outperforms or obtains a comparable performance with the state-of-the-art methods on DAVIS [34, 35] and YouTube-VOS [49] benchmarks. We also give some qualitative results to show the effectiveness of our model. Next, we conduct comprehensive ablation studies to analyze the effect of the individual components of our method and some configurations. Finally, we explore how to improve efficiency and strike a balance between segmentation quality and inference speed.

4.1 Implementation details

We use the first four stages of ResNet50 [10] pretrained on ImageNet [24] and replace its input layer with the proposed dynamic input adapter to form our feature extractor. The number of transformer encoder layers and decoder layers is set to $L = 6$. The multi-head attention layers have $M = 8$ heads, width $d = 256$, while the feed-forward networks have hidden units of 2048. Dropout ratio of 0.1 is used. The proposed model is trained with the input resolution of 480p, and the length T of the training video clip is set to 2. Similar to [32], the maximum temporal interval of sampling increases by 5 every 20 training epochs. We freeze all batch normalization layers and minimize our loss using AdamW optimizer ($\beta = (0.9, 0.999)$, $eps = 10^{-8}$, and the weight decay is 10^{-4}) with the initial learning rate $lr = 10^{-4}$. During training, we adopt a bootstrapping strategy for the cross-entropy loss, where only the top 40% pixels with maximum training loss are considered. The model is trained with batchsize 4 for 160 epochs on 4 TITAN RTX GPUs, taking about 1.5 days. Note that our model is flexible for different sizes of the reference sets. In the inference stage, to balance the accuracy and efficiency, our model with input resolution 480p only refers to the first and previous frames to segment the current frame. We conduct all inference experiments on a single TITAN RTX GPU.

4.2 Datasets and evaluation metrics

We evaluate our approach on DAVIS [34, 35] and YouTube-VOS [49] benchmarks. Both DAVIS2016 and DAVIS2017 have experimented. DAVIS2016 is an annotated single-object dataset containing 30 training and 20 validation video sequences. DAVIS2017 is a multi-objects dataset expanded from DAVIS2016, including 60 training video sequences, 30 validation video sequences, and 30 test video sequences. The youTube-VOS dataset is a large-scale VOS dataset with 3471 training videos and 474 validation videos. And each video contains a maximum of 12 objects. The validation set includes seen objects from 65 training categories and unseen objects from 26 categories, appropriate for evaluating algorithms' generalization performance. We use the evaluation metrics provided by the DAVIS benchmark to evaluate our model. $J\&F$ evaluates the general quality of the segmentation results, J evaluates the mask IoU, and F estimates the quality of contours.

4.3 Comparison with the state-of-the-art

DAVIS We compare the proposed model with the state-of-the-art methods on DAVIS benchmark [34, 35]. We also present the results trained with additional data from YouTube-VOS [49]. The evaluation results on DAVIS16-val and DAVIS17-val are reported in Table 1. When adding YouTube-VOS for training, our method achieves state-of-the-art performance on DAVIS17-val (83.9% in $J\&F$), outperforming the online-learning methods with a large margin and having higher performance than the matching-based methods such as STM, RMNet, and CFBI. Specifically, our model outperforms transformer-based SST with 1.4% in $J\&F$, surpasses JOINT with a gap of 0.4% in $J\&F$, and exceeds AOT-B with 1.8% in $J\&F$. When only using DAVIS for training, our model achieves better quantitative results than those with the same configuration and even better than several methods like FEELVOS and AGAME, which apply YouTube-VOS for training. On DAVIS16-val, our model has comparable performance with state-of-the-art methods. Compared to KMN, our model has the same $J\&F$ score with a higher J score while a slightly lower F score. Since DAVIS 2016 is a single object dataset, segmentation details such as boundaries play an important role in performance evaluation. We believe that the Hide-and-Seek training strategy, which provides more precise boundaries, helps KMN a lot. We also report the results on the DAVIS17 test-dev in Table 2. Our

Table 1 Comparison with the state-of-the-art on the DAVIS16 and DAVIS17-val. 'OL' indicates the use of online-learning strategy. '+YT' means the use of YouTube-VOS for training. Runtime of other methods was obtained from the corresponding papers

Methods	OL	DAVIS16 val				DAVIS17 val		
		FPS	$\mathcal{J}\&\mathcal{F}(\%)$	$\mathcal{J}(\%)$	$\mathcal{F}(\%)$	$\mathcal{J}\&\mathcal{F}(\%)$	$\mathcal{J}(\%)$	$\mathcal{F}(\%)$
OSVOS [1]	✓	0.22	80.2	79.8	80.6	60.3	56.7	63.9
OnAVOS [40]	✓	0.08	85.5	86.1	84.9	67.9	64.5	71.2
PReMVOS [28]	✓	0.03	86.8	84.9	88.6	77.8	73.9	81.7
STM-cycle(+YT) [20]	✓	-	-	-	-	72.3	69.3	75.3
FRTM(+YT) [36]	✓	21.9	83.5	-	-	76.7	-	-
RGMP [31]		7.7	81.8	81.5	82.0	66.7	64.8	68.6
RaNet [47]		30	85.5	85.5	85.4	65.7	63.2	68.2
AGSS [23]		-	-	-	-	66.6	63.4	69.8
GC [19]		25	86.6	87.6	85.7	71.4	69.3	73.5
AFB-URR [22]		-	-	-	-	74.6	73.0	76.1
AGAME(+YT) [14]		14.3	82.1	82.0	82.2	70.0	67.2	72.7
FEELVOS(+YT) [39]		2.2	81.7	81.1	82.2	71.5	69.1	74.0
STM(+YT) [32]		6.3	89.3	88.7	89.9	81.8	79.2	84.3
KMN(+YT) [37]		8.3	**90.5**	89.5	91.50	82.8	80.0	85.6
EGMN(+YT) [27]		-	-	-	-	82.8	80.2	85.2
AOT-B(+YT) [54]		22.7	89.9	88.8	90.9	82.1	79.4	84.8
CFBI(+YT) [53]		6	89.4	88.3	90.5	81.9	79.1	84.6
JOINT(+YT) [29]		-	-	-	-	83.5	80.8	86.2
SST(+YT) [8]		-	-	-	-	82.5	79.9	85.1
RMNet(+YT) [48]		12	88.8	88.9	88.7	83.5	81.0	86.0
SwiftNet(+YT) [42]		25	90.4	90.5	90.3	81.1	78.3	83.9
SITVOS(+YT) [16]		11.8	**90.5**	89.5	91.4	83.5	80.4	86.5
Ours		6.6	85.8	85.4	86.2	78.1	75.7	80.5
Ours(+YT)		6.6	**90.5**	89.8	91.2	**83.9**	**81.4**	86.4

Table 2 Compare to the state of the art on the DAVIS17 test-dev set and YouTube-VOS 2018 validation set. 'OL' indicates the use of online-learning strategy. The subscripts of $\mathcal{J}$ and $\mathcal{F}$ on YouTube-VOS denote seen objects (s) and unseen objects (u). The metric overall means the average of $\mathcal{J}_s, \mathcal{J}_u, \mathcal{F}_s, \mathcal{F}_u$

Methods	OL	DAVIS17 test-dev			YouTube-VOS 2018 val				
		$\mathcal{J}\&\mathcal{F}(\%)$	$\mathcal{J}(\%)$	$\mathcal{F}(\%)$	Overall	$\mathcal{J}_s(\%)$	$\mathcal{J}_u(\%)$	$\mathcal{F}_s(\%)$	$\mathcal{F}_u(\%)$
OSVOS [1]	✓	50.9	47.0	54.8	58.8	59.8	54.2	60.5	60.7
OnAVOS [40]	✓	52.8	49.9	55.7	55.2	60.1	46.6	62.7	51.4
PReMVOS [28]	✓	71.6	67.5	75.7	-	-	-	-	-
STM-cycle [20]	✓	58.6	55.3	62.0	70.8	72.2	62.8	76.3	71.9
FRTM [36]	✓	-	-	-	72.1	72.3	65.9	76.2	74.1
RGMP [31]		52.9	51.3	54.4	53.8	59.5	45.2	-	-
AGSS [23]		-	-	-	71.3	71.3	65.5	75.2	73.1
AGAME [14]		-	-	-	66.1	67.8	60.8	-	-
FEELVOS [39]		57.8	55.2	60.5	-	-	-	-	-
RaNet [47]		55.3	53.4	-	-	-	-	-	-
STM [32]		72.2	69.3	75.2	79.4	79.7	72.8	84.2	80.9
GC [19]		-	-	-	73.2	72.6	68.9	75.6	75.7
CFBI [53]		74.8	71.1	78.5	81.4	81.1	75.3	85.8	**83.4**
AFB-URR [22]		-	-	-	79.6	78.8	74.1	83.1	82.6
KMN [37]		77.2	74.1	80.3	81.4	81.4	75.3	85.6	83.3
EGMN [27]		-	-	-	80.2	80.7	74.0	85.1	80.9
SST [8]		-	-	-	81.7	81.2	76.0	-	-
RMNet [48]		75.0	71.9	78.1	81.5	82.1	75.7	85.7	82.4
SwiftNet [42]		-	-	-	77.8	77.8	72.3	81.8	79.5
Ours		76.9	73.0	**80.9**	**81.8**	82.0	75.0	**86.7**	**83.4**

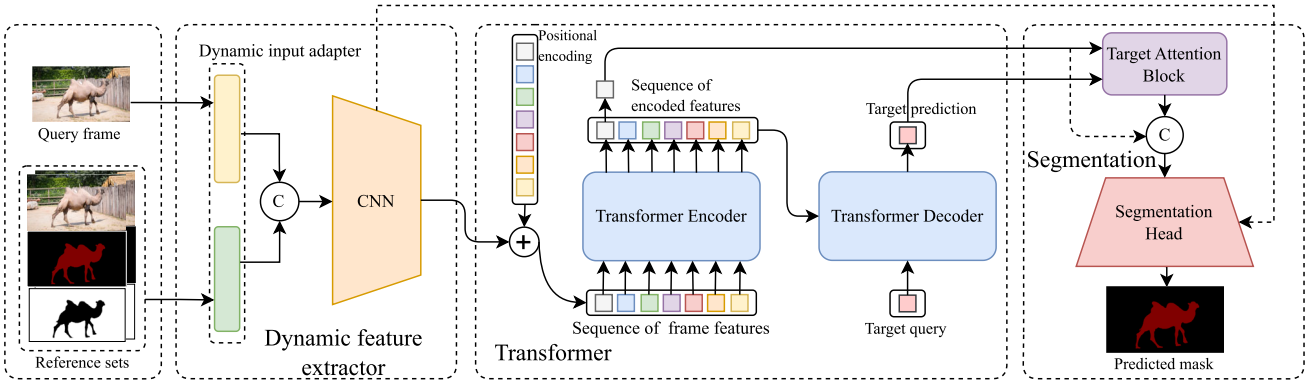

Fig. 3 Overview of our model. The feature extractor is used to extract the features of the current frame and reference sets. The vision transformer is exploited to model the temporal and spatial relationships. The target attention block (TAB) is used to extract the target mask features. The segmentation head is designed to obtain the predicted object mask. "+", "C" indicate adding and concatenating operation, respectively. For a better view, we only illustrate two reference frames

model outperforms all the online-learning methods. Except for slightly lower than KMN of 0.3% in $J\&F$, our model surpasses all the methods in the second part. Note that we only use simple data augmentation when pretraining on static image datasets, while KMN applies more complicated pretraining strategies, which helps improve the performance.

YouTube-VOS Table 2 compares with state-of-the-art methods on YouTube-VOS 2018 validation [49]. On this benchmark, our method obtains an overall score of 81.8% and outperforms all the methods in the first and second parts, demonstrating that the proposed method is robust and effective. Specifically, our model surpasses STM by 2.4% in the overall score. Note that we only refer to the first and previous frames to segment the current frame, while STM contains a large memory bank that saves a new memory frame every five. Also, our model outperforms KMN and CFBI with gaps of both 0.4% in the overall score. Besides, it surpasses the most related transformer-based SST. Our model achieves the best F scores while not the best for J scores. We explain that the model may pay more attention to spatial relationships to obtain more accurate target boundaries and acquire higher overall scores on YouTube-VOS 2018.

Qualitative results The visualization (Fig. 3) results on DAVIS17-val are shown in Fig. 4, and the qualitative results on DAVIS17 test-dev and YouTube-VOS 2018 validation are shown in Fig. 5. The proposed method handles object

Fig. 4 Qualitative results on the DAVIS2017-val. The groundtruth is visualized in the first row, and the next three rows show comparisons of our method with STM [32] and CFBI [53]. Our model handles object occlusion better due to the strong ability of spatial-temporal modeling

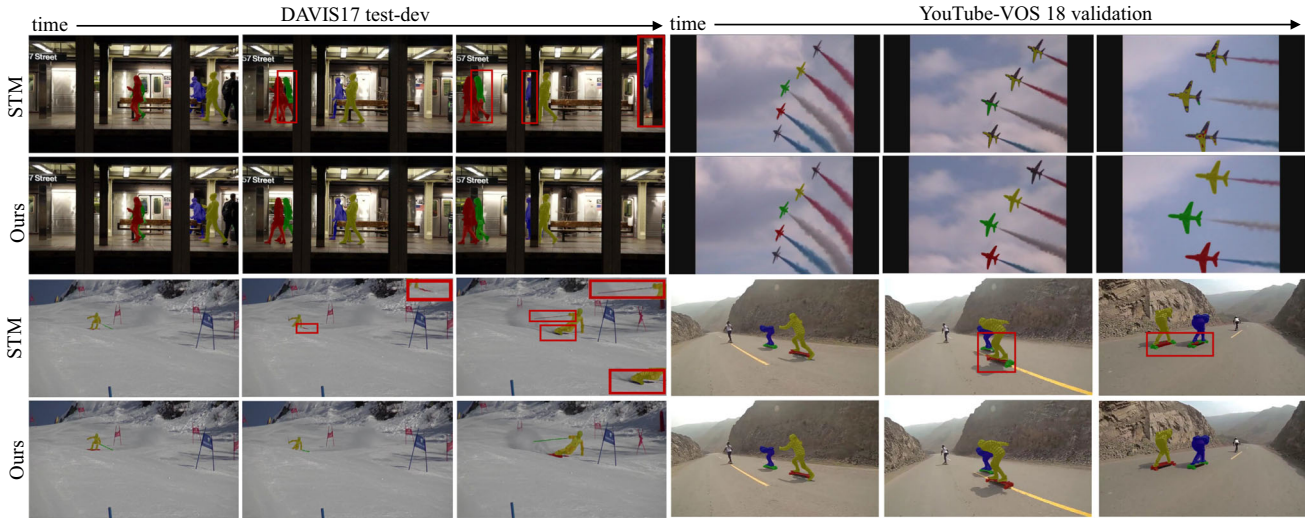

Fig. 5 Qualitative results o DAVIS2017 test-dev and YouTube-VOS 2018 validation. Compared to STM, our model performs better when segmenting highly similar objects and fast-moving objects

occlusion better due to the strong ability of spatial-temporal modeling. Also, our model performs better when segmenting highly similar objects and fast-moving objects.

4.4 Ablation study

We conduct all the ablation experiments on DAVIS17 validation [35]. The model used in this section does not do pre-training on synthetic videos, and the input resolution is 240p unless specified. Moreover, we test the model with only the first and previous frames referred by default. Here we list the ablation studies about dynamic feature extractor, mask utilization, reference sets, transformer structure, backbone, training strategy, and input resolution.

Dynamic feature extractor In Table 4, we compare our model equipped with the proposed dynamic feature extractor with two variants: i) the existing approach of using two independent extractors (as in STM [32]; denoted *'Independent'*), and ii) using a siamese architecture and concatenating the object mask to the reference frame features (as in AGAME [14]; denoted as *'Siamese'*). Results showed that our model employs fewer parameters (about 20% reduction) than i) but obtains higher performance (+7.8% in $J\&F$ score) than ii).

Mask utilization To demonstrate the effectiveness of our dynamic feature extractor, we implement three typical ways to utilize the predicted masks of past frames. (1) the predicted masks are multiplied with the encoded features of the RGB frame, denoted as '*multiply*'; (2) the encoded features of the RGB frame and the predicted mask are multiplied first and then added to the former, denoted as '*residual*'; (3) the predicted masks and the RGB frame are fed into a dynamic input adapter, denoted as '*adapter*'. As shown in Table 3, compared to directly multiplying the predicted mask with encoded features (line 1) and fusing the mask with residual structure (line 2), our dynamic input adapter gains 15.1%($J\&F$) and 8.0%($J\&F$) improvement.

Reference sets We test how reference sets affect the performance of our proposed model. We experiment with four types of reference set configurations: (1) Only the first frame with the ground-truth masks; (2) Only the previous frame with its predicted mask; (3) Both the first and previous frame with their masks; (4) The reference set is dynamically updated by appending new frames with the predicted masks every five

Table 3 Ablation studies of mask utilization and reference sets with input resolution 240p on DAVIS 2017 validation set

Mask utilization	Reference sets	$\mathcal{J}_M(\%)$	$\mathcal{J}_R(\%)$	$\mathcal{J}_D(\%)$	$\mathcal{F}_M(\%)$	$\mathcal{F}_R(\%)$	$\mathcal{F}_D(\%)$	$\mathcal{J}\&\mathcal{F}(\%)$	FPS
multiply	1st frame	51.4	59.9	13.4	58.3	63.6	14.4	54.9	-
residual	1st frame	58.5	67.1	17.6	65.5	75.0	18.6	62.0	-
adapter	1st frame	66.5	78.2	13.3	73.6	83.5	15.9	70.0	23.0
adapter	previous frame	64.3	74.8	11.7	70.5	81.3	14	67.4	17.6
adapter	1st & previous frames	73.1	86.6	1.8	79.7	91.5	5.3	76.4	17.1
adapter	Every 5 frames	70.2	82.2	6.0	77.6	89.0	8.1	73.9	5.1

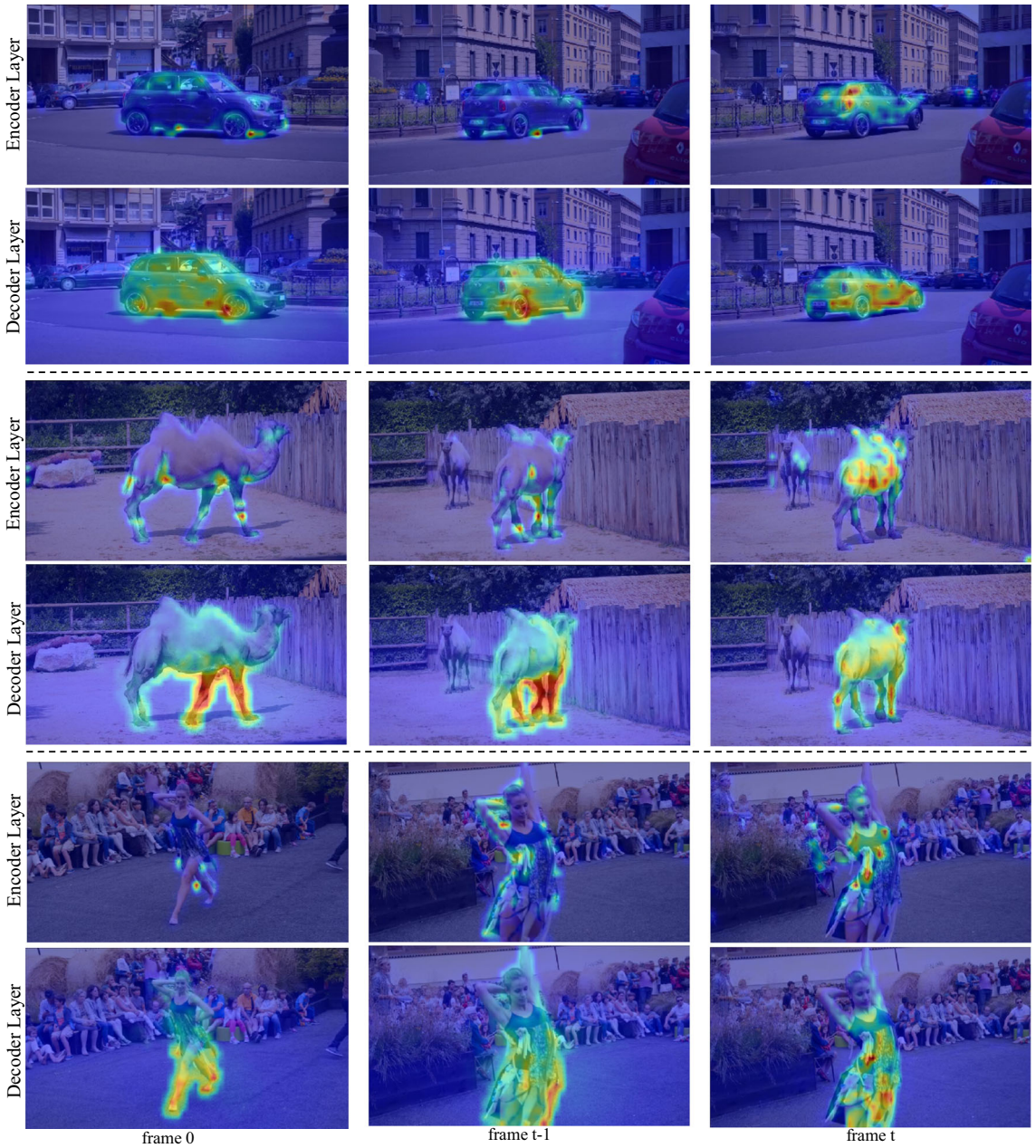

Fig. 6 Visualization of attention maps from the transformer encoder/decoder layers. The first row of each sample ("car", "camel", and "twirl girl") shows the attention maps from the fourth transformer encoder layer. We take the center 16x16 patch of the object of the current frame (t-th frame) as the query to get the attention weights. The second row is the visualization of the cross-attention weights from the fourth transformer decoder layers with only one target query. The decoder layers pay more attention to the target object and can reduce the interference of the background

Table 4 Impacts of different types of feature extractors. Models are tested with the input resolution of 240p on DAVIS17-val

Feature extractor	$\mathcal{J}$ (%)	$\mathcal{F}$ (%)	$\mathcal{J}\&\mathcal{F}$ (%)	Parameters (M)	FPS
Siamese	64.9	72.4	68.6	35.61	18.1
Independent	72.8	80.3	76.5	43.08	17.0
Dynamic	73.1	79.7	76.4	34.56	17.0

Table 5 Ablation studies of different components with input resolution 240p on DAVIS 2017 validation set. 'TD' denotes the transformer decoder

Components	$\mathcal{J}_M$ (%)	$\mathcal{J}_R$ (%)	$\mathcal{J}_D$ (%)	$\mathcal{F}_M$ (%)	$\mathcal{F}_R$ (%)	$\mathcal{F}_D$ (%)	$\mathcal{J}\&\mathcal{F}$ (%)
w/o TD	71.7	83.4	5.2	78.7	89.7	7.5	75.2
w/ TD	73.1	86.6	1.8	79.7	91.5	5.3	76.4

frames. As Table 3 shows, our model could achieve superior performance even with two frames referred. Interestingly, we find that updating the memory every five frames as STM [32] for VOS may not be beneficial in all methods because low-quality segmentation results of historical frames may mislead subsequent mask prediction.

Transformer structure We visualize attention maps from the transformer encoder/decoder layers in Fig. 6. The attention maps from the fourth encoder layer show that the encoder focuses on the target object but is still disturbed by the background. In contrast, the transformer decoder can eliminate background influence (Table 4) and pay more attention to the target object. We also do quantitative experiments to explore the effectiveness and necessity of the transformer decoder in Table 5. It can be seen that by equipping with the transformer decoder, our model obtains 1.2%($J\&F$) improvement over removing it. Therefore, it is essential to employ the transformer's decoder.

Backbone We experiment with different backbones, ResNet18, ResNet50 [10] and Swin Transformer [26]. As shown in Table 6, our model with a smaller backbone ResNet18 runs faster (7 fps improvement) than ResNet50 while the performance drops 4.1% ($J\&F$). The model with Swin-small as backbone gains 0.5% ($J\&F$) than ResNet50 but contains more parameters and runs slower (5 fps drop). Therefore, we take ResNet50 as the backbone to achieve a more balanced performance.

Training strategy We conduct experiments to explore the effectiveness of pretraining on synthetic videos. As Table 7 shows, our model only drops by 1.5% ($J\&F$) without pre-training, which means our proposed approach can learn general and robust target object appearance even trained with a small dataset.

Table 6 Ablation studies of different backbone with input resolution 240p on DAVIS 2017 validation set

Backbone	$\mathcal{J}_M$ (%)	$\mathcal{J}_R$ (%)	$\mathcal{J}_D$ (%)	$\mathcal{F}_M$ (%)	$\mathcal{F}_R$ (%)	$\mathcal{F}_D$ (%)	$\mathcal{J}\&\mathcal{F}$ (%)	FPS
ResNet18	68.8	80.6	7.7	75.9	86.3	9.1	72.3	24.0
ResNet50	73.1	86.6	1.8	79.7	91.5	5.3	76.4	17.0
Swin-small	73.9	87.0	5.2	80.0	93.4	8.5	76.9	12.0

Table 7 Training data analysis on DAVIS 2017 validation set. We do abaltion studies to explore how the pre-training affects our model's performance

Training strategy	$\mathcal{J}_M$ (%)	$\mathcal{J}_R$ (%)	$\mathcal{J}_D$ (%)	$\mathcal{F}_M$ (%)	$\mathcal{F}_R$ (%)	$\mathcal{F}_D$ (%)	$\mathcal{J}\&\mathcal{F}$ (%)
w/o pre-training	73.1	86.6	1.8	79.7	91.5	5.3	76.4
w/ pre-training	74.4	85.6	6.8	81.4	91.3	8.1	77.9

Input resolution We adjust the input resolution of the model as shown in Table 8, from which we can see that our method achieves better performance with a larger input size. Our proposed method with half input resolution runs faster (10.4 fps improvement) while the performance drops 4.0% ($J\&F$). Therefore, we compare our model with input resolution 480p to other state-of-the-art methods.

4.5 Exploration of efficiency improvement

This section aims to discuss how to strike a balance between segmentation accuracy and inference speed. As shown in Tables 6 and 8, our model can achieve a trade-off between latency and accuracy by replacing with a light backbone or reducing the input size to half of the input resolution. Besides, we observed that the transformer causes a large portion of the latency. Therefore, another way of improving the efficiency is to reduce the computation of the transformer. Firstly, we provide a slim version with only half the number of the original model's transformer encoder/decoder layers. The slim version achieves 11.1 fps with 82.8% ($J\&F$) on DAVIS17-val, as shown in Table 9.

Moreover, since the computational complexity of the self-attention mechanism is proportional to the square of the length of the input sequences, we further propose to compress the redundant reference representations for efficiency improvement. Inspired by [21, 42], we modify the PAM module in SwiftNet [42] as a sequence organizer to compress the redundant elements in reference sequences. Note that the PAM module is originally used to compress redundant pixels of the key and value embeddings in the memory bank. We briefly describe the compression process here based on SwiftNet. In the online inference phase, if frame $\mathbf{I}_t$ with the predicted masks $\mathbf{M}_t$ is chosen as a reference frame, for each

Table 8 Input resolution analysis. We compared models with different input resolution on DAVIS 2017 validation set

Input resolution	$\mathcal{J}_M$ (%)	$\mathcal{J}_R$ (%)	$\mathcal{J}_D$ (%)	$\mathcal{F}_M$ (%)	$\mathcal{F}_R$ (%)	$\mathcal{F}_D$ (%)	$\mathcal{J}\&\mathcal{F}$ (%)	FPS
240p	74.4	85.6	6.8	81.4	91.3	8.1	77.9	17.0
480p	81.4	90.6	7.0	86.4	93.7	8.8	83.9	6.6

Table 9 Exploration of efficiency improvement. Models are tested on DAVIS17-val

Variants	Parameters (M)	$\mathcal{J}(\%)$	$\mathcal{F}(\%)$	$\mathcal{J}\&\mathcal{F}(\%)$	FPS
Vanilla	34.56	81.4	86.4	83.9	6.6
Slim version	25.88	80.5	85.1	82.8	11.1
+ Sequence organizer	34.56	81.0	86.3	83.7	10.0

element in the representations $\mathbf{X}_t$ of $(\mathbf{I}_t, \mathbf{M}_t)$, the sequence organizer finds its most relevant element in representations $\mathbf{X}_{R,t-1}$ of the reference set at timestamp $t-1$ via dot-product and computes the cosine similarity as the feature score. Then elements in $\mathbf{X}_t$ are sorted by the feature scores. Finally, the top β (β is experimentally set to 10%) percents elements of $\mathbf{X}_t$ is selected and added to the reference set. From Table 9, we can see that the sequence organizer improves the inference speed by over 50% with a slight accuracy drop (0.2% in $J\&F$).

5 Conclusions

This paper proposes a new framework for video object segmentation (VOS), a compact and unified single-extractor pipeline with robust spatial and temporal interaction ability using a vision transformer. Specifically, we propose a dynamic feature extractor to encode the reference sets and query frames in a unified way, dramatically slimming the existing VOS framework while maintaining the performance and architecture's flexibility. Moreover, we attach the vision transformer to the dynamic feature extractor to model the spatial and temporal relationships among reference sets and query frames simultaneously, providing discriminating spatial-temporal features for segmenting. We implement an effective and modularized framework with the extractor, transformer, and segmentation head. Moreover, our model achieves top performance on several benchmarks, demonstrating its potential and effectiveness. Also, we explore how to strike a balance between the segmentation quality and inference speed. In the future, we will further improve our model's efficiency by designing a better sequence organizer and applying it after each transformer encoder layer.

Acknowledgements This work was supported by NSFC 62088101 Autonomous Intelligent Unmanned Systems.

References

1. Caelles S, Maninis KK, Pont-Tuset J, Leal-Taixé L, Cremers D, Van Gool L (2017) One-shot video object segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 221–230
2. Carion N, Massa F, Synnaeve G, Usunier N, Kirillov A, Zagoruyko S (2020) End-to-end object detection with transformers. In: European Conference on Computer Vision, Springer, pp 213–229
3. Chen X, Li Z, Yuan Y, Yu G, Shen J, Qi D (2020) State-aware tracker for real-time video object segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 9384–9393
4. Cheng B, Schwing A, Kirillov A (2021) Per-pixel classification is not all you need for semantic segmentation. Adv Neural Inf Process Syst 34
5. Cheng HK, Tai YW, Tang CK (2021) Rethinking space-time networks with improved memory coverage for efficient video object segmentation. Adv Neural Inf Process Syst 34:11781–11794
6. Cheng MM, Mitra NJ, Huang X, Torr PH, Hu SM (2014) Global contrast based salient region detection. IEEE Trans Pattern Anal Mach Intell 37(3):569–582
7. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S et al (2020) An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929
8. Duke B, Ahmed A, Wolf C, Aarabi P, Taylor GW (2021) Sstvos: Sparse spatiotemporal transformers for video object segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 5912–5921
9. Everingham M, Van Gool L, Williams CK, Winn J, Zisserman A (2010) The pascal visual object classes (voc) challenge. Int J Comput Vision 88(2):303–338
10. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
11. Hu YT, Huang JB, Schwing AG (2018) Videomatch: Matching based video object segmentation. In: Proceedings of the European conference on computer vision (ECCV), pp 54–70
12. Huang W, Gu J, Ma X, Li Y (2020) End-to-end multitask siamese network with residual hierarchical attention for real-time object tracking. Appl Intell 50(6):1908–1921
13. Huang X, Xu J, Tai YW, Tang CK (2020) 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, pp 8879–8889
14. Johnander J, Danelljan M, Brissman E, Khan FS, Felsberg M (2019) A generative appearance model for end-to-end video object segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 8953–8962
15. Lai Z, Lu E, Xie W (2020) Mast: A memory-augmented self-supervised tracker. In: Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition, pp 6479–6488
16. Lan M, Zhang J, He F, Zhang L (2022) Siamese network with interactive transformer for video object segmentation. In: Proceedings of the AAAI Conference on artificial intelligence, 36:1228–1236
17. Li B, Yan J, Wu W, Zhu Z, Hu X (2018) High performance visual tracking with siamese region proposal network. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 8971–8980
18. Li Y, Hou X, Koch C, Rehg JM, Yuille AL (2014) The secrets of salient object segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 280–287
19. Li Y, Shen Z, Shan Y (2020) Fast video object segmentation using the global context module. In: European Conference on Computer Vision, Springer, pp 735–750
20. Li Y, Xu N, Peng J, See J, Lin W (2020) Delving into the cyclic

mechanism in semi-supervised video object segmentation. Adv Neural Inf Process Syst 33:1218–1228
21. Liang Y, Ge C, Tong Z, Song Y, Wang J, Xie P (2022) Not all patches are what you need: Expediting vision transformers via token reorganizations. arXiv preprint arXiv:2202.07800
22. Liang Y, Li X, Jafari N, Chen J (2020) Video object segmentation with adaptive feature bank and uncertain-region refinement. Adv Neural Inf Process Syst 33:3430–3441
23. Lin H, Qi X, Jia J (2019) Agss-vos: Attention guided single-shot video object segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp 3949–3957
24. Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL (2014) Microsoft coco: Common objects in context. In: European conference on computer vision, Springer, pp 740–755
25. Liu Y, Zhang D, Zhang Q, Han J (2021) Part-object relational visual saliency. IEEE Trans Pattern Anal Mach Intell
26. Liu Z, Lin Y, Cao Y, Hu H, Wei Y, Zhang Z, Lin S, Guo B (2021) Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 10012–10022
27. Lu X, Wang W, Danelljan M, Zhou T, Shen J, Van Gool L (2020) Video object segmentation with episodic graph memory networks. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16, Springer, pp 661–679
28. Luiten J, Voigtlaender P, Leibe B (2018) Premvos: Proposal-generation, refinement and merging for video object segmentation. In: Asian conference on computer vision, Springer, pp 565–580
29. Mao Y, Wang N, Zhou W, Li H (2021) Joint inductive and transductive learning for video object segmentation. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 9670–9679
30. Mei J, Wang M, Yang Y, Li Y, Liu Y (2023) Fast real-time video object segmentation with a tangled memory network. ACM Trans Intell Syst Technol 14(3):1–21
31. Oh SW, Lee JY, Sunkavalli K, Kim SJ (2018) Fast video object segmentation by reference-guided mask propagation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 7376–7385
32. Oh SW, Lee JY, Xu N, Kim SJ (2019) Video object segmentation using space-time memory networks. In: Proceedings of the IEEE/CVF International conference on computer vision, pp 9226–9235
33. Perazzi F, Khoreva A, Benenson R, Schiele B, Sorkine-Hornung A (2017) Learning video object segmentation from static images. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2663–2672
34. Perazzi F, Pont-Tuset J, McWilliams B, Van Gool L, Gross M, Sorkine-Hornung A (2016) A benchmark dataset and evaluation methodology for video object segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 724–732
35. Pont-Tuset J, Perazzi F, Caelles S, Arbeláez P, Sorkine-Hornung A, Van Gool L (2017) The 2017 davis challenge on video object segmentation. arXiv preprint arXiv:1704.00675
36. Robinson A, Lawin FJ, Danelljan M, Khan FS, Felsberg M (2020) Learning fast and robust target models for video object segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 7406–7415
37. Seong H, Hyun J, Kim E (2020) Kernelized memory network for video object segmentation. In: European Conference on Computer Vision, Springer, pp 629–645
38. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I (2017) Attention is all you need. Adv Neural Inf Process Syst 30
39. Voigtlaender P, Chai Y, Schroff F, Adam H, Leibe B, Chen LC (2019) Feelvos: Fast end-to-end embedding learning for video object segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 9481–9490
40. Voigtlaender P, Leibe B (2017) Online adaptation of convolutional neural networks for the 2017 davis challenge on video object segmentation. In: The 2017 DAVIS challenge on video object segmentation-CVPR Workshops, vol. 5
41. Voigtlaender P, Luiten J, Torr PH, Leibe B (2020) Siam r-cnn: Visual tracking by re-detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 6578–6588
42. Wang H, Jiang X, Ren H, Hu Y, Bai S (2021) Swiftnet: Real-time video object segmentation. In: Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition, pp 1296–1305
43. Wang H, Liu W, Xing W (2022) A temporal attention based appearance model for video object segmentation. Appl Intell 52(2):2290–2300
44. Wang H, Zhu Y, Adam H, Yuille A, Chen LC (2021) Max-deeplab: End-to-end panoptic segmentation with mask transformers. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 5463–5474
45. Wang M, Mei J, Liu L, Tian G, Liu Y, Pan Z (2022) Delving deeper into mask utilization in video object segmentation. IEEE Trans Image Process 31:6255–6266
46. Wang Q, Zhang L, Bertinetto L, Hu W, Torr PH (2019) Fast online object tracking and segmentation: A unifying approach. In: Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition, pp 1328–1338
47. Wang Z, Xu J, Liu L, Zhu F, Shao L (2019) Ranet: Ranking attention network for fast video object segmentation. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 3978–3987
48. Xie H, Yao H, Zhou S, Zhang S, Sun W (2021) Efficient regional memory network for video object segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 1286–1295
49. Xu N, Yang L, Fan Y, Yang J, Yue D, Liang Y, Price B, Cohen S, Huang T (2018) Youtube-vos: Sequence-to-sequence video object segmentation. In: Proceedings of the European conference on computer vision (ECCV), pp 585–601
50. Xu Y, Wang Z, Li Z, Yuan Y, Yu G (2020) Siamfc++: Towards robust and accurate visual tracking with target estimation guidelines. In: Proceedings of the AAAI conference on artificial intelligence, pp 12549–12556
51. Yan B, Peng H, Fu J, Wang D, Lu H (2021) Learning spatio-temporal transformer for visual tracking. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 10448–10457
52. Yang J, Ge H, Su S, Liu G (2022) Transformer-based two-source motion model for multi-object tracking. Appl Intell pp 1–13
53. Yang Z, Wei Y, Yang Y (2020) Collaborative video object segmentation by foreground-background integration. In: European Conference on Computer Vision, Springer, pp 332–348
54. Yang Z, Wei Y, Yang Y (2021) Associating objects with transformers for video object segmentation. Adv Neural Inf Process Syst 34

55. Zhang XY, Huang YP, Mi Y, Pei YT, Zou Q, Wang S (2021) Video sketch: A middle-level representation for action recognition. Appl Intell 51(4):2589–2608
56. Zhu W, Li J, Lu J, Zhou J (2021) Separable structure modeling for semi-supervised video object segmentation. IEEE Trans Circuits Syst Video Technol

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.