Computational Visual Media

https://doi.org/10.1007/s41095-022-0282-8 Vol. 9, No. x, month year, xx–xx

Research Article

Global Video Object Segmentation with Spatial Constraint Module

Yadang Chen[1], Duolin Wang[1], Zhiguo Chen[1](✉), Zhi-Xin Yang[2], and Enhua Wu[3,4].

Abstract We present a lightweight and efficient semi-supervised video object segmentation network based on the space-time memory framework. To some extent, our method solves the two difficulties encountered in traditional video object segmentation: one is that the single frame calculation time is too long, and the other is that the current frame's segmentation should use more information from past frames. The algorithm uses a global context (GC) module to achieve high-performance, real-time segmentation. The GC module can effectively integrate multi-frame image information without increased memory and can process each frame in real time. Moreover, the prediction mask of the previous frame is helpful for the segmentation of the current frame, so we input it into a spatial constraints module (SCM), which constrains the areas of segments in the current frame. The SCM effectively alleviates mismatching of similar targets yet consumes few additional resources. We added a refinement module to the decoder to improve boundary segmentation. Our model achieves state-of-the-art results on various datasets, scoring 80.1 on YouTube-VOS 2018 and a $\mathcal{J}\&\mathcal{F}$ score of 78.0 on DAVIS 2017, while taking 0.05 seconds per frame on the DAVIS 2016 validation dataset.

Keywords Video Object Segmentation, Semantic Segmentation, Global Context Module, Spatial Constraint

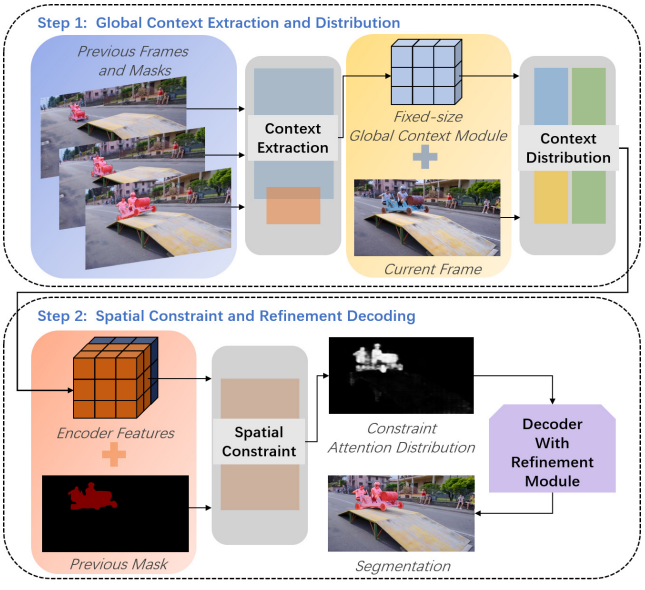


Fig. 1 Our GVOS-SCM solution has three key steps> (i) Context extraction: extract each frame's information into a fixed-size updater (see Eqns. 2, 3). (ii) Context distribution: match the current frame's semantic information with that in the updater at pixel level (see Eqn. 4). (iii) Spatial constraint enforcement: the mask of the previous frame is input into the spatial constraint module (see Eqn. 5).

1 Introduction

Video object segmentation, which aims to draw a detailed object mask on video frames, is widely applicable to various fields such as autopilots, video editing, and video synthesis. It originates from video object tracking [1]. Approaches can be divided into unsupervised methods [2, 3] that input only the video, and semi-supervised methods [4–7] that require a user to provide initial labels. In our work, we consider the second approach. The reason for doing so is that defining what constitutes an 'interesting object' is often application-specific, and the same video could have multiple valid solutions. Thus, cues regarding which objects are of interest can be concretely indicated by labels specifying this on a few key frames.

Existing deep learning-based algorithms for semi-supervised video object segmentation can be classified as propagation-based methods, matching-based

1 Engineering Research Center of Digital Forensics, Ministry of Education, School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing, 210044, China. Email:(✉)chenzhiguo@nuist.edu.cn

2 State Key Laboratory of Internet of Things for Smart City, Department of Electromechanical Engineering, University of Macau, Macau, 999078, China.

3 State Key Laboratory of Computer Science, Institute of Software, University of Chinese Academy of Sciences, Beijing, 100190, China.

4 Faculty of Science and Technology, University of Macau, Macau, 999078, China.

Manuscript received: 2022-01-01; accepted: 2022-01-01

methods, hybrid methods, and space-time memory based methods. Propagation-based methods [4, 8–11] utilize the target's temporal coherence, and rely on the mask from previous frames. For example, MaskTrack [11] combines the segmentation mask of the previous frame with the current frame to form the mask of the current frame. However, these methods suffer from occlusion problems and error drift. Matching-based methods [5, 12–14] uses the first frame of a given video as a reference frame and detect the segmented object independently in each frame. These methods are more robust and reduce the impact of occlusion, but do not take full advantage of spatiotemporal information. Hybrid methods [6, 15–17] integrate the above two methods, employing the previous frame and the first frame to segment the current frame, to integrate the advantages of the two types of method. Accordingly, the performance and accuracy of some hybrid algorithms are improve on the former two classes.

Since hybrid methods can significantly improve video target segmentation, it is natural to ask whether we can use more frames to learn richer contextual information. A recent paper uses this idea in the design of a new Space-Time Memory (STM) network [7, 18, 19]. In order to use information from more frames, STM stores key-value pairs extracted from past frames into a memory pool and then matches information extracted from the current frame with the information in the memory pool, at the pixel level. This algorithm has better robustness and good segmentation performance, even in the case of occlusion and appearance variation.

Although STM based methods achieve state-of-the-art precision, they suffer from excessive memory consumption, especially on long videos. When the STM module learns new information from a new frame, the module adds it to the memory. Over time, more and more memory is used, and may even result in memory exhaustion. To solve this problem, the author reduces the number of frames read and updates the memory every five frames. However, linearly increasing memory is still used over time, and the solution does not make the best use of the information in each frame.

In our work, inspired by [7], we employ a global context module (see Fig. 1) that retrieves the segmentation information in a more efficient way. As the learned video frames advance, the module automatically updates the information. Unlike the linear memory growth of [18], the size of the global context module is fixed and does not increase over time. There is no chance of memory exhaustion, and we can learn variations in the object through time.

When similar objects enter the field of view, the model sometimes makes incorrect predictions. Furthermore, the model performs poorly when the shape of the object changes dramatically. For counter these problems, we employ a spatial constraint module, inspired by [20] (see Fig. 1). It uses a mask from the previous image as a rough constraint to guide the model in removing confusing instances of similar appearance. In addition, we use ASPP [21] modules to handle scale changes in the video. Finally, we use a refinement module [22–25] after the decoder to further improve the segmentation results near the target boundary.

2 Related Work

2.1 Detection-based Methods

Detection-based methods rely on fine-tuning using the first-frame ground truth. They assume that a powerful frame-level target detector can be constructed, which can segment video frame by frame. OSVOS [12] is a representative algorithm, which uses a pre-trained convolution network for foreground-background segmentation, and first-frame ground truth for fine-tuning. OnAVOS [13] and OSVOS-S [5] introduce an online adaptation mechanism based on OSVOS. PML [26] proposes an embedding network with triplet loss and a nearest neighbour classifier. Most detection-based methods require online training, so the fine-tuning time will greatly affect the performance of the model, making it incapable of providing real-time results. A model based on fine-tuning from the first frame will be more robust to occlusion and other problems. However, due to the loss of available temporal information, such methods can fail if the shape of the target changes so drastically that the detector cannot recognize the target.

2.2 Propagation-based Methods

Propagation-based methods rely on the temporal coherence of the video, since most videos are smoothly varying. Thus we only need an adjustment to the mask of the previous frame to get the mask for the current frame. MaskTrack [11] is a typical propagation-based approach that inputs the mask of the previous frame and the current frame into the model, and outputs the mask of the current frame. Lucid [27] extends this method by introducing an elaborate data augmentation mechanism. Joint-task [2] and learning-correspondence [28] approaches first learn a visual representation, then uses KNN [29] to train the model to learn a feature mapping representation to perform cycle consistency tracking. The advantage of this approach is that it can overcome rapid large changes in appearance, but it cannot overcome occlusion, drift and other problems.

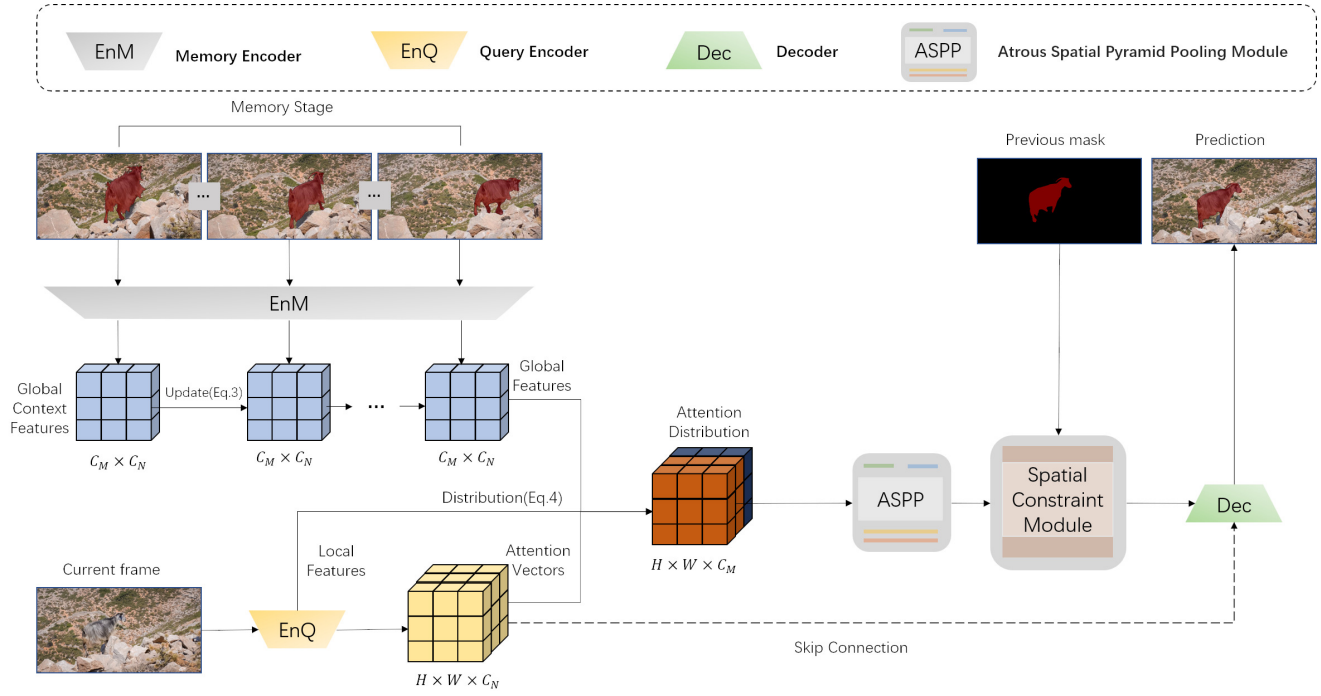


Fig. 2 Pipeline. Past frames are input into the network for encoding and sent to a fixed-size global context module. The module's content is updated automatically as frames advance. The encoder generates a set of attention vectors for the current frame to retrieve relevant information in the global context, to form global features. The encoder also generates local features. The global and local features are concatenated and passed to the constraint module, whose result is passed to the decoder to produce the segmentation result for the frame.

2.3 Space-time Memory Based Methods

STM [18] is a semi-supervised video object segmentation method. Although traditional propagation-based methods and matching-based methods achieve good results, they still do not use as many frames as possible in the video sequence, so much semantic information is lost. Inspired by the non-local method of [30], STM uses a novel attention module, which allows multiple frames from the video to pass through the encoder module, stores the information in the memory module, and then matches the information in the current frame with the information in the memory module, at the pixel level, to determine whether each pixel belongs to the foreground object. It does not need to limit the number of frames.

As STM does not rely on the assumption of video smoothness when learning spatial semantic information between distant pixels, it is possible to train the network first with static pictures with masks. Previous work [7, 19] has also used this strategy to generate 3-frame composite video clips by applying random affine transformations to static pictures with different parameters. We use image datasets annotated with object masks to train our network, and by doing so, we can produce a model that is robust to a variety of object appearance and category transformations.

STM also has various drawbacks, such as incorrect matching to similar-looking objects, imprecise edge processing for target objects, and poor segmentation quality when the object appearance changes too much. There are thus many improvement schemes for STM. For example, to alleviate mismatching, KMN [19] improves STM's memory reading module by using a 2D Gaussian kernel. AFB-URR [31] reduces memory consumption. STCN [32] and LCM [33] target improved segmentation accuracy. RMNet [34] uses optical flow [35, 36] to constrain the extent of the segmentation.

Due to the limited memory capacity, adding information to the memory module continually will lead to memory exhaustion. STM addresses this by saving image information every five frames, but this violates the original intent of matching all frames before the current frame, one by one. To alleviate the increasing memory consumption during STM usage, we employ a global context module. Every time we read a new number of frames, the global context module automatically updates its content without increasing resource consumption.

To sum up, the advantage of STM-based methods is that their network model elegantly uses as many frames as possible, and learns more context information than traditional methods, thus accurately predicting the mask of the current frame.

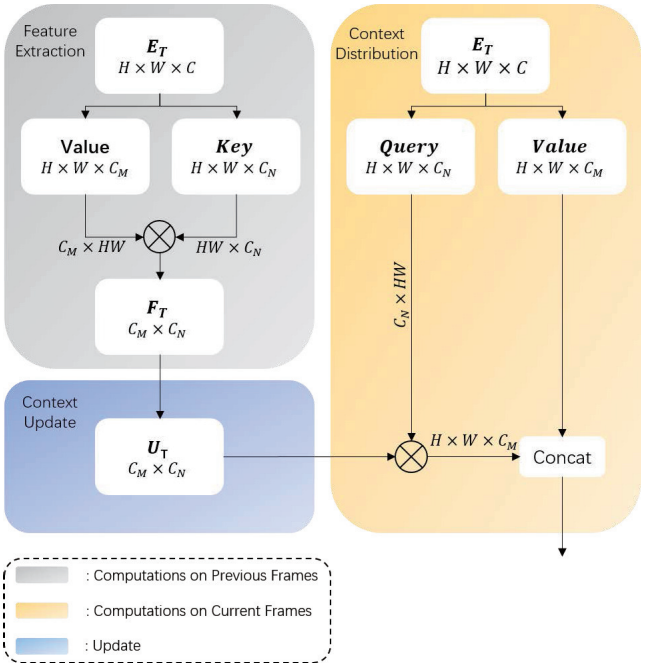


Fig. 3 Context distribution module.

3 Approach

In this section, we introduce a new efficient video object segmentation (VOS) framework based on STM methods. We first overview our framework in Sec. 3.1. In Sec. 3.2, we describe the principle of operation of the global context module, then we introduce the spatial constraint module in Sec. 3.3, We finally describe the boundary-aware refinement module in Sec. 3.4.

3.1 Overview

Fig. 2 overviews our architecture. The main structure is an encoder-decoder. The global context (GC) module is based on the output of the memory encoder and query encoder. Two different encoders are used to generate features at $H \times W$ resolution with C channels. The GC module has two functions: context extraction and updating, and context distribution. First, we use the memory encoder to extract semantic information from previous frames and their masks, and put it into a fixed-size updater. Next, we use the query encoder to encode the current frame to get a local feature embedding. We match the local features of the current frame with those in the updater at pixel level, then use an atrous spatial pyramid pooling module to get richer semantic information. The feature map is then spatially constrained to the target object through the spatial constraint module (SCM) to reduce errors due to similar objects. Finally, our prediction map is obtained through the decoder via a boundary-aware refinement module (BAM).

3.2 Global Context Module

3.2.1 STM versus GCM

Many recent VOS methods use attention mechanisms, with encouraging results. As a formulation, we may define query embedding of the current frame as $Q_r \in R^{HW \times C}$, key embedding of the memory frames as $K_y \in R^{THW \times C}$ and value embedding of the memory frames as$V_l \in R^{THW \times C}$, where H, W, C, T denote height, width, number of channels and temporal extent. Space time memory propagation is formulated as:

$$\begin{aligned} \mathrm{STM}(Q_r, K_y, V_l) &= \mathrm{CorF}(Q_r, K_y)V_l \\ &= \mathrm{softmax}(Q_r K_y^{tr}/\sqrt{C})V_l, \end{aligned} \tag{1}$$

where a distribution map is computed by the correlation function CorF. After multiplying Q_r and K_y^{tr}, softmax is applied to the resulting feature map, converting its values to the range [0,1], and then the value embedding V_l is propagated into each location of the current frame.

In the STM, the key-value pair vectors for each frame are stored in the memory module. As time advances, the number of video frames increases, and these vectors are concatenated, so K_y and V_l become larger and larger: computing STM requires more effort with greater video resolution or video duration.

In order to overcome the problem of excessive consumption of system resources, we employ the global context module, which works differently from the STM. The global context module automatically updates the information, without increasing its size, while having almost the same representation ability as STM.

3.2.2 Context Extraction and Update

. The global context module evolved from STM module, so their architectures are very similar. As Fig. 3 shows, we have two different encoders. One memory encoder encodes previous frames and their masks to generate the keys and values, of size $H \times W \times C_N$ and $H \times W \times C_M$ respectively, where C_N and C_M are the numbers of channels used. Another query encoder encodes the current frame, also generating queries and values.

STM keeps concatenating keys and values, so its memory pool gets ever bigger. The innovation of the GCM over STM is to combine keys and values generated by past frames into a fixed-size updater, and to update data automatically as new frames arrive. We call the step doing this the global summary step.

In this step, the STM method treats the key-value pair vectors generated by the encoder as $H \times W$ locations, where each location is a vector of $C_N(C_M)$ dimensions, while the

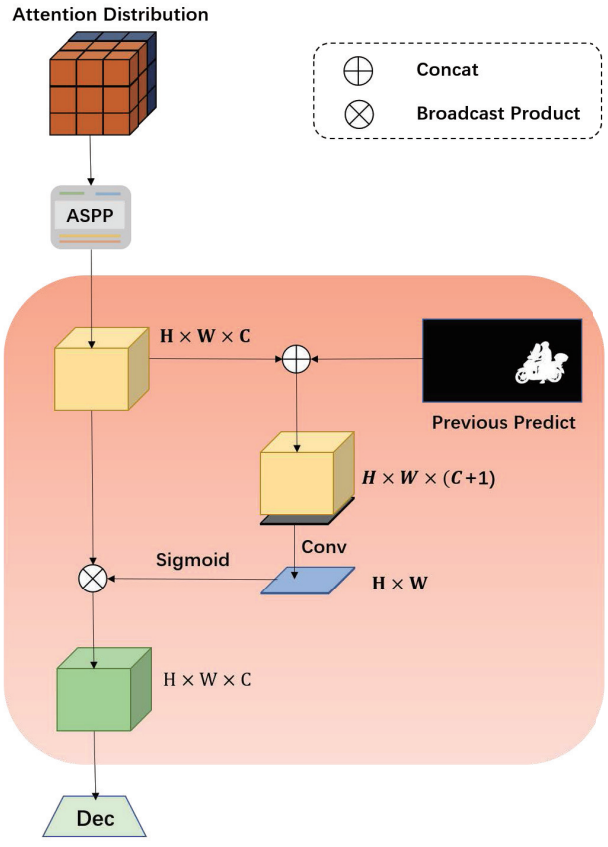


Fig. 4 Spatial constraint module.

GCM treats the key-value pairs as $C_N(C_M)$ one-channel feature maps and then considers them as several weight matrices related to the key-value pair vectors. The GCM first computes the context matrix of the current frame from the key-value pair vectors generated by the context extraction process, using

$$F_t = K_y(E_t)^T V_l(E_t), \tag{2}$$

where E_t denotes the output of the encoder at time t, F_t is the feature matrix of this frame, and K_y, V_l are functions that generate keys and values. We then include the resulting information in F_t in the global context matrix. Since the matrix has fixed size, we do so without using additional resources. The update for the global context module is performed follows:

$$U_t = \frac{1}{t}F_t + \frac{t-1}{t}U_{t-1}, \tag{3}$$

where U_t denotes the global context module. The weights ensure each F_p for $1 \leqslant p \leqslant t$ to contributes equally to U_t .

3.2.3 *Context Distribution*

We match the query and value information extracted from the current frame to the information stored in the global context module at the pixel level, which we call context distribution. In this process, we multiply the query with size $H \times W \times C_N$ by GT which has size $C_M \times C_N$ to get a matrix of size $H \times W \times C_M$, and then concatenate the matrix with the value produced by the current frame to get the output of GC module. This may be written:

$$I_T = Q_r(E_t)U_{t-1}, \tag{4}$$

where I_t represents the distributed global features for frame t, and Q is the function generating the queries.

The global context module summarizes the areas of semantic interest in the query position of the current frame for context features in past frames. The STM does this by first identifying such areas by query-key matching, then summarizing their values by weighted sum. The GCM achieves the same goal more effectively as the global context vector is already a global summary of all previously semantically similar regions in the framework. Query location only needs to determine the appropriate weight of the global context vector to generate a vector that summarizes all regions of interest.

3.3 Spatial Constraint Module

We employ a spatial constraint module (SCM, see Fig. 4) to ensure spatial consistency between adjacent frames, and reduces error due to similarity of of appearance, avoiding false predictions caused by similar instances of the same category. The prediction mask of the previous frame is a 0-1 mask of shape $H \times W$, which is cascaded with the current frame embedding ($H \times W \times C$) to obtain a feature map of shape $H \times W \times (C+1)$. A convolution layer with a 3×3 kernel and a sigmoid function are used to generate a spatial prior, which is a gate map of shape $H \times W$. The prior is multiplied by the current frame embedding. The SCM can be expressed as

$$S_T = \frac{1}{1+\exp\left(f_n\left(E_T \oplus P_{T-1}\right)\right)} \otimes E_T, \tag{5}$$

where E_T represents the encoder feature map of T frame, P_{T-1} represents the predicted mask of the previous frame, f_n denotes the convolution function, and $\oplus$ and $\otimes$ represent concatenation and element-wise product, respectively. Example attention maps generated by the SCM are shown in Fig. 5.

3.4 Boundary-Aware Refinement Module

3.4.1 *Architecture*

The spatial constraints module greatly reduces problems due to occlusion, but the target object may also change as the video progresses. SCM is not good enough alone to ensure high segmentation accuracy, so we use several methods to improve the segmentation accuracy our architecture. After the context

Fig. 5 Attention maps generated by the spatial constraint module

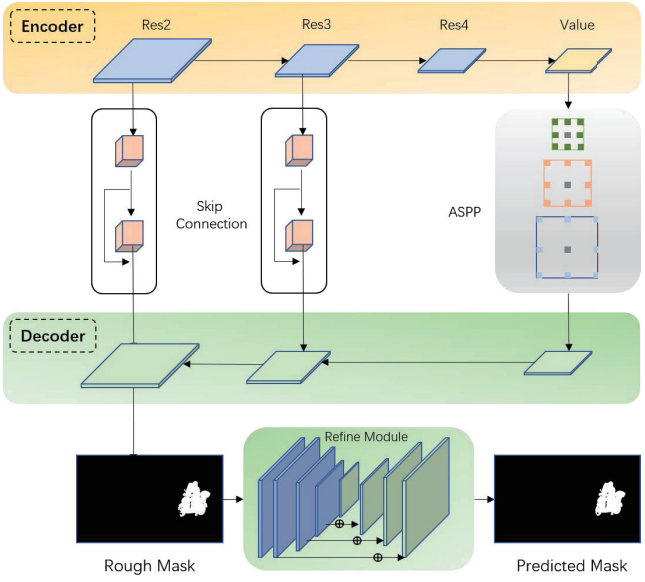


Fig. 6 Decoder.

distribution operation, we employ an atrous spatial pyramid pooling (ASPP) module, to obtain semantic information at different scales.

To improve the segmentation boundary, we apply a refinement module before soft aggregation. Refinement modules are usually designed as encoder-decoder modules, as shown in Fig. 6, with residual connections to avoid loss of precision while learning deeper information about the frame.

$$S_{\text{refined}} = S_{\text{coarse}} + S_{\text{residual}}. \tag{6}$$

We employ a novel residual refinement module (RRM) to refine both region and boundary drawbacks in coarse maps. As Fig. 6 shows, all of our convolution cores are of size 3×3. A batch normalization, a ReLu activation function, and a maxpool function are used after each convolution during the encoding phase. In the decoding phase, we use bilinear interpolation up-sampling; after each up-sampling is completed, we use 3×3 convolution and skip the convolution of the encoder and decoder. Similarly, a batch normalization and a ReLu activation function are used after the convolution operation. The loss function of the RRM is hybrid loss, which will be described later. The output of this RRM module is used as input to soft aggregation [16], which merges the multi-object prediction; the loss function for soft aggregation is cross entropy loss.

3.4.2 *Hybrid Loss*

Accuracy of boundaries is one of the difficulties in image segmentation. To solve this issue, we employ the concept of hybrid loss. We combine three losses corresponding to three levels:

$$\ell^{(k)} = \ell^{(k)}_{\text{bce}} + \ell^{(k)}_{\text{ssim}} + \ell^{(k)}_{\text{iou}}, \tag{7}$$

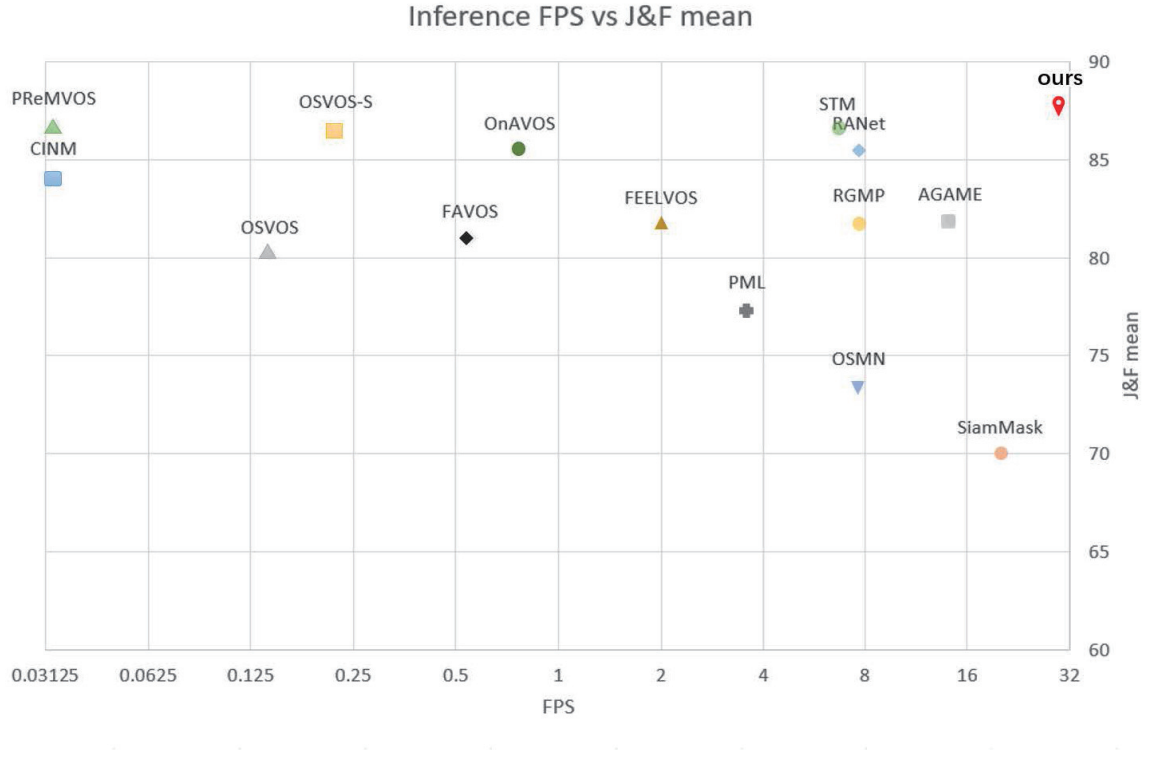


Fig. 7 Speed (FPS) versus accuracy ($\mathcal{J}\&\mathcal{F}$ mean) comparison using the DAVIS 2016 validation set at 480p resolution.

Table 1 Comparison of Accuracy of Model with Full Training to Models Using Pre-training Only or Main Training Only.

Variant	YouTube-VOS	DAVIS-2017	
	Overall	$\mathcal{J}$	$\mathcal{F}$
Pre-training only	70.1	72.5	73.9
Main-training only	65.2	50.1	52.6
Full training	**80.5**	85.2	84.3

Table 2 Accuracy and speed of models using STM and GCM on the DAVIS 2017 dataset.

Variant	$\mathcal{J}(\%)$	$\mathcal{F}(\%)$	$\mathcal{J}\&\mathcal{F}(\%)$	Time (s)
Memory Read	85.5	86.6	81.1	0.15
GCM	85.2	84.3	80.1	0.05

where $l^{(k)}$ is the loss of the k-th side output, $\ell_{\text{bce}}^{(k)}$,$\ell_{\text{ssim}}^{(k)}$ and $\ell_{\text{iou}}^{(k)}$ denote BCE loss, SSIM loss and IoU loss, respectively.

BCE is the most basic and commonly used binary cross-entropy. BCE loss is computed pixel-wise, for fairness:

$$\ell_{\text{bce}} = -\sum_{(r,c)} [G(r,c)\log(S(r,c)) + (1-G(r,c))\log(1-S(r,c))], \tag{8}$$

where r and c represent pixel coordinates, G is the ground truth mask, and S is the predicted value of the object.

SSIM is the structural similarity index. It is designed to assess picture quality, capture structure information, and learn structure relationships between a target and ground truth. SSIM loss acts on a patch-level, and the key is that it considers boundaries. SSIM loss is defined as:

$$\ell_{\text{ssim}} = 1 - \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{9}$$

where x, y sets represent areas of size $N \times N$ extracted from the predicted probability map S and ground truth. μ_x, μ_y, σ_x^2, and σ_y^2 are the mean and variance of x and y, respectively. σ_{xy} is the covariance of x and y.

The third loss is the IoU loss, which is acts at a map-level:

$$\ell_{\text{iou}} = 1 - \frac{\sum_{r=1}^{H}\sum_{c=1}^{W} S(r,c)G(r,c)}{\sum_{r=1}^{H}\sum_{c=1}^{W}[S(r,c) + G(r,c) - S(r,c)G(r,c)]}, \tag{10}$$

where r and c represent pixel coordinates, G is the ground truth mask, and S is the predicted value of the object.

4 Experiments

This section describes implementation details of our framework and experiments carried out the on the DAVIS 2016 [37], DAVIS 2017 [38], and YouTube-VOS 2018 [39] datasets. Evaluation metrics used for object segmentation are average region similarity ($\mathcal{J}$ mean), the average contour accuracy ($\mathcal{F}$ mean), and the average of the two ($\mathcal{J}\&\mathcal{F}$ mean). As Fig. 7 shows, our network model achieves a very good balance between speed and accuracy relative to other methods.

Fig. 8 Results with those from RGMP [16], FEELVOS [6], and RaNet [15].

Table 3 Memory usage for our framework and STM, at various times t.

Method	t	Memory	Method	t	Memory
STM	0	4MB	Ours	any	**1 MB**
	10	40MB			
	100	394MB			

Table 4 Accuracy of Models with and without SCM.

Variant	$\mathcal{J}(\%)$	$\mathcal{F}(\%)$	$\mathcal{J}\&\mathcal{F}(\%)$	Time (s)
SCM	85.2	83.9	80.1	0.05
W/O SCM	79.4	80.1	75.5	0.05

4.1 Datasets

The *DAVIS 2016 & 2017* [37, 38] datasets are intended to benchmark pixel-perfect labelling. Their goal is to provide realistic video scenes including camera jitter, background clutter, occlusion, and other complications. DAVIS 2016 [37] is a single-target dataset containing 50 video sequences, 30 of which are for training and 20 for validation. DAVIS 2017 [38] is a multi-target dataset. Each frame contains several different annotated targets. It includes 150 video sequences, 376 target instances, and 10459 frames.

The *YouTube-VOS 2018* dataset [39] is by far the largest video object segmentation dataset, comprising 4,453 YouTube video clips and 94 target instances, which allows comprehensive evaluation and comparison of video object segmentation methods.

4.2 Implementation Details

Our model is first pre-trained on the video clips simulated using an image dataset, then trained on the video dataset.

4.2.1 Pre-training on Image Datasets

Training with a static image database compensates for the lack of frames in the video database, and avoids over-fitting caused by a lack of training data. This method assumes no temporal relationship between images, and uses static picture datasets to train the video object segmentation models. Previous work used static images to train their networks, and we took a similar approach. The specific implementation applies random affine transformations [11] to various images. A video sequence composed of three frames is generated and used to train our network, making our network more robust and easier to adapt to different segmentation targets. We pre-trained our model on the CoCo dataset [46].

4.2.2 Main Training on Video Datasets

We used real video data for the main training stage, using DAVIS 2016 [37], DAVIS 2017 [38], and YouTube-VOS 2018 [39] datasets according to different training objectives. We randomly used three frames in the correct temporal order from the same video sequence as training samples. In order to learn appearance changes in objects over a long period, we randomly skipped frames during the sampling process. As training progressed, the number of frames skipped increased from 0 to 25.

Table 5 Comparison using DAVIS 2016 and DAVIS 2017 validation sets. Results for online (OL) and non-online methods are sorted by $\mathcal{J}\&\mathcal{F}$ mean. +YV indicates use of YouTube-VOS for training. The three best scores are indicated in red, blue and yellow, respectively (same for other tables).

Method	OL	Time (s)	DAVIS-2016			DAVIS-2017		
			$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$ Mean	$\mathcal{F}$ Mean	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$ Mean	$\mathcal{F}$ Mean
OSVOS [12]	✓	7	80.2	79.8	80.6	60.3	56.6	63.9
Lucid [27]	✓	-	83.0	83.9	82.0	-	-	-
CINM [40]	✓	>30	84.2	83.4	85.0	70.6	67.2	74.0
OnAVOS [13]	✓	13	85.5	86.1	84.9	65.4	61.6	69.1
OSVOS-S [5]	✓	4.5	86.6	85.6	87.5	-	-	-
PReMVOS [41]	✓	>30	**86.8**	84.9	**88.6**	**77.8**	**73.9**	**81.8**
DyeNet [10]	✓	2.32	-	86.2	-	-	-	-
GEM [42]		-	64.6	69.6	59.6	-	-	-
SiamMask [43]		**0.03**	70.0	71.7	67.8	56.4	71.7	67.8
OSMN [17]		0.13	73.5	74.0	72.9	54.8	52.5	57.1
PML [26]		0.28	77.4	75.5	79.3	-	-	-
VidMatch [44]		0.32	-	81.0	-	-	56.5	-
FAVOS [4]		1.8	81.0	82.4	79.5	58.2	54.6	61.8
FEELVOS [6]		0.5	81.7	80.3	83.1	69.1	65.9	72.3
RGMP [16]		0.13	81.8	81.5	82.0	66.7	64.8	68.6
AGAME [45]		0.07	81.9	81.5	82.2	70.0	67.2	72.7
RANet [15]		0.13	85.5	85.5	85.4	65.7	63.2	68.2
STM [18]		0.15	86.5	84.8	88.1	71.6	69.2	74.0
GC [7]		**0.04**	86.6	**87.6**	85.7	71.4	69.3	73.5
OURS		**0.05**	**88.7**	**89.5**	**87.9**	**74.2**	**72.5**	**75.8**
OURS (+YV)		**0.05**	**90.1**	**91.0**	**89.2**	**78.0**	**75.4**	**80.5**

4.2.3 Other Training Details

We randomly clipped input frames to a size of 384×384. We used the Adam [47] optimizer with a fixed learning rate of 10^{-5}. We froze the batch normalization layer during training. The mini-batch size was 4. Both pre- and main-training used random affine transformations, but the main training process was less random. The sampling intervals increased by 5 after every 20 epochs, both for Davis and YouTube-VOS.

4.3 Ablation Study

We performed ablation experiments using the DAVIS 2017 dataset to see how each module of our network contributes to the final results.

4.3.1 Pre-training and Main Training

An interesting result from our experiments is that when we only do pre-training, the video segmentation capability of the model is better than when the model only undergoes main training, which indicates that the size of the training set has a significant influence on the resulting network. When omitting pre-training, the overall accuracy on the YouTube dataset for the main training-only model decreased by 15% (see Table 1): our model is severely over-fitting. These experiments show that the rich static image resources used in pre-training can help enhance our network's robustness, so we use both pre- and main-training strategies for the model to achieve the best results.

4.3.2 Global Context Module

The GCM uses a fixed-size updater so that as the number of video frames increases, the model memory usage does not: the network can learn information from each frame. Results of a comparison to STM's update module using the DAVIS 2017 dataset are shown in Table 2 with STM using the same scheme of reading all frames as GC. It can be seen that GCM's speed of processing video is significantly better, while accuracy is not greatly affected.The $\mathcal{J}$ mean and $\mathcal{J}$ mean obtained by STM are 0.3% and 2.3% higher than by GCM, respectively. The improvement is minimal, but GC runs three times faster than STM. Table 3 shows the memory consumption of the two methods. As t increases, STM's resource consumption increases linearly, while GCM's resource consumption remains at a very low level.

4.3.3 Spatial Constraint Module

The spatial constraint module is used to reduce mismatching of target objects with similar appearance. A comparison was performed with and without the module using the DAVIS 2017 dataset. It shows that the module can significantly prevent mismatching yet has little effect on computational efficiency, as shown in Table 4. In a multi-object video set, the target is more susceptible to interference from similar objects, and the

improvement provided by the SCM becomes very obvious: when SCM is used, $\mathcal{J}$ and $\mathcal{F}$ are improved by 5.8% and 3.8%, respectively, while SCM does not affect speed. As Fig. 5 shows, the SCM uses a mask from the previous frame to focus the current frame on the target object, greatly reducing mismatching.

4.4 Comparisons to State-of-the-Art Methods

4.4.1 DAVIS 2016 (Single Object)

. The first comparison used the verification set from the DAVIS 2016 benchmark, with single-object videos. We directly cite results for other representative works from the DAVIS 2016 benchmark website, including for the recent STM [18] and RANet [15]. Results are given in Table 5. We can see that using the online learning method returns higher scores.

Fig. 7 draws a scatter diagram for various methods according to speed and accuracy. It can be seen that the accuracy of methods based on online learning is very high, but the online learning process is time-consuming, and the calculation time is prolonged. Offline learning methods have high calculation speed, but lower accuracy. Recent methods such as STM achieve a balance between accuracy and speed, running at 6.7 FPS. Our framework improves upon STM, and its speed reaches 25 FPS. It is noteworthy that the videos in DAVIS are very short, mostly not exceeding 100 frames. As the time taken by STM increases linearly with number of frames, as video length increases, STM will become slower and slower, while our framework can maintain high computing speed for any video length. In general, our method achieves the highest speed, and its $\mathcal{J}$ mean score is also among the best.

As Fig. 9, columns 1, 3 show, even when the target object undergoes severe deformation, our method can segment the object accurately and is unaffected by occlusion.

4.4.2 DAVIS 2017 (Multiple Object)

DAVIS17 is a multi-object segmentation database, in which many objects interfere and obscure each other. Multi-object scenarios are more challenging than single-target scenarios. In Table 4, we compare our framework with several existing mainstream frameworks and see that online learning-based methods perform equally well in multi-target scenarios. However, the computation time for online learning methods is prolonged. For offline learning methods, our framework is more accurate and faster than STM.

The spatial constraints module gives our network model a distinct advantage in multi-target classification tasks. In Fig. 9, rows 2, 4, 5, our method correctly identifies different entities.

4.4.3 YouTube-VOS

One of the features of the YouTube-VOS dataset is that there are some unseen targets in the validation set. Table 6 compares different methods using this dataset. STM again achieved high scores in this test. Our framework significantly improves upon STM, achieving high scores on seen and unseen object segmentation.

4.5 Qualitative Results

Fig. 8 shows visual examples of the segmented results of our framework and other frameworks. Our spatial constraints module can effectively handle many challenging situations, such as object confusion, size changes, and appearance transformations. Our refinement module can help to segment the edges of the target object. In the first row, RGMP [16], FEELVOS [6], and RaNet [15] all identify two dogs as the same entity, while our method accurately identifies two entities. In the second row, the RaNet method again has a problem of misidentification. In the third row, the RGMP and FEELVOS methods do not recognize the target object. In the last row, all three methods have mismatching problems.

However, there is still room for improvement in our framework. As Fig. 10 shows, when an object is severely deformed, it may lead to in inaccurate results (see row 1, columns 4, 5). When the target object does not appear in a long sequence of frames, this may cause segmentation to fail (see row 2, column 3, 4). Thanks to the robustness of our network, the number of frames in which segmentation fails usually does not exceed two (see row 2, column 5). Mismatches can also occur when several split objects are very close together and there are interactions between them (see row 3, columns 2, 4). Since the spatial constraints module uses the mask from the previous frame's segmentation result, our network may also treat an occlusion as a object target if there is occlusion in the current frame (see row 4, columns 3, 4, 5);

To sum up, some imperfect segmentation results under extreme conditions, generally our framework provides very good segmentation results even with target occlusion, target confusion, complex object appearance; our network also achieves a very good balance between accuracy and speed.

5 Conclusions

We have designed a new video object segmentation framework. Fast video frame information acquisition and updating are achieved through the GCM based on the STM approach; it captures object segmentation information in processed frames through a fixed size updater. We also use a

Table 6 Comparison using the YouTube-VOS validation set.

Method	Overall	$\mathcal{J}$ seen	$\mathcal{J}$ unseen	$\mathcal{F}$ seen	$\mathcal{F}$ unseen
RVOS [48]	56.8	63.6	45.5	67.2	51.0
OSVOS [12]	58.8	59.8	54.2	60.5	60.7
S2S(OL) [49]	64.4	71.0	55.5	70.0	61.2
VSBMM [50]	64.5	70.0	62.5	66.2	59.3
PReMVOS [41]	66.9	71.4	56.5	75.9	63.7
AGAME [45]	66.1	67.8	60.8	-	-
BoLTVOS [51]	71.1	**71.6**	64.3	-	-
AGSS [52]	71.3	71.3	65.5	**76.2**	73.1
STM [18]	**79.4**	**79.7**	**72.8**	**84.2**	**80.9**
GC [7]	**73.2**	**72.6**	**68.9**	75.6	**75.7**
GVOS-SCM (ours)	**80.1**	**80.1**	**76.6**	**83.2**	**81.4**

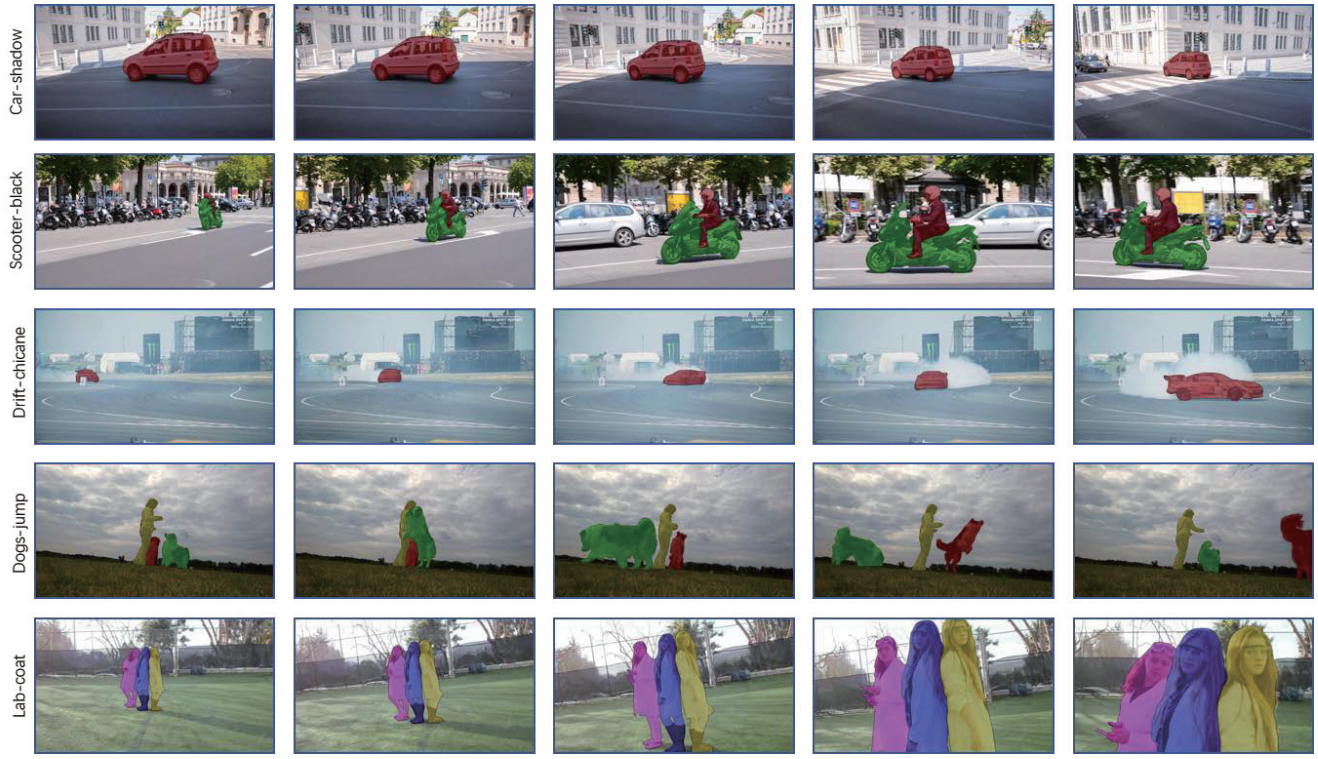


Fig. 9 Further results from our method using the DAVIS 2017 validation set.

spatial constraint module, which helps our network to achieve outstanding results in multi-target problems. Finally, we use a refinement module to help our network provide a more refined segmentation boundary for the target object. As experiments on benchmark datasets show, our method outperforms STM, in terms of both accuracy and speed. Furthermore, because of the GCM, our network cannot run out of memory over time. Overall, our solution is efficient and compatible, and we hope it will set a strong baseline for other real-time video object segmentation solutions in the future.

Acknowledgements

This work was partially supported by the National Natural Science Foundation of China (Grants 61802197, 62072449, 61632003), the Science and Technology Development Fund, Macau SAR (Grants 0018/2019/AKP and SKL-IOTSC(UM)-2021-2023), the Guangdong Science and Technology Department (Grant 2018B030324002), and the Zhuhai Science and Technology Innovation Bureau Zhuhai-Hong Kong-Macau Special Cooperation Project (Grant ZH22017002200001PWC)

Declaration of competing interests

The authors have no competing interests to declare that are relevant to the content of this article.

Fig. 10 Imperfect segmentation results

References

[1] Chen D, Tang F, Dong W, Yao H, Xu C. SiamCPN: Visual tracking with the Siamese center-prediction network. *Computational Visual Media*, 2021, 7(2): 253–265.

[2] Li X, Liu S, De Mello S, Wang X, Kautz J, Yang MH. Joint-task self-supervised learning for temporal correspondence. *arXiv preprint arXiv:1909.11895*, 2019.

[3] Zhang FL, Barnes C, Zhang HT, Zhao J, Salas G. Coherent video generation for multiple hand-held cameras with dynamic foreground. *Computational Visual Media*, 2020, 6(3): 291–306.

[4] Cheng J, Tsai YH, Hung WC, Wang S, Yang MH. Fast and accurate online video object segmentation via tracking parts. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, 7415–7424.

[5] Maninis KK, Caelles S, Chen Y, Pont-Tuset J, Leal-Taixé L, Cremers D, Van Gool L. Video object segmentation without temporal information. *IEEE transactions on pattern analysis and machine intelligence*, 2018, 41(6): 1515–1530.

[6] Voigtlaender P, Chai Y, Schroff F, Adam H, Leibe B, Chen LC. Feelvos: Fast end-to-end embedding learning for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, 9481–9490.

[7] Li Y, Shen Z, Shan Y. Fast Video Object Segmentation using the Global Context Module. *arXiv preprint arXiv:2001.11243*, 2020.

[8] Hu YT, Huang JB, Schwing AG. Maskrnn: Instance level video object segmentation. *arXiv preprint arXiv:1803.11187*, 2018.

[9] Khoreva A, Benenson R, Ilg E, Brox T, Schiele B. Lucid data dreaming for object tracking. In *The DAVIS challenge on video object segmentation*, 2017, 1–6.

[10] Li X, Loy CC. Video object segmentation with joint re-identification and attention-aware mask propagation. In *Proceedings of the European conference on computer vision (ECCV)*, 2018, 90–105.

[11] Perazzi F, Khoreva A, Benenson R, Schiele B, Sorkine-Hornung A. Learning video object segmentation from static images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, 2663–2672.

[12] Caelles S, Maninis KK, Pont-Tuset J, Leal-Taixé L, Cremers D, Van Gool L. One-shot video object segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, 221–230.

[13] Voigtlaender P, Leibe B. Online adaptation of convolutional neural networks for video object segmentation. *arXiv preprint arXiv:1706.09364*, 2017.

[14] Shin Yoon J, Rameau F, Kim J, Lee S, Shin S, So Kweon I. Pixel-level matching for video object segmentation using convolutional neural networks. In *Proceedings of the IEEE international conference on computer vision*, 2017, 2167–2176.

[15] Wang Z, Xu J, Liu L, Zhu F, Shao L. Ranet: Ranking attention network for fast video object segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, 3978–3987.

[16] Oh SW, Lee JY, Sunkavalli K, Kim SJ. Fast video object segmentation by reference-guided mask propagation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, 7376–7385.

[17] Yang L, Wang Y, Xiong X, Yang J, Katsaggelos AK. Efficient video object segmentation via network modulation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, 6499–6507.

[18] Oh SW, Lee JY, Xu N, Kim SJ. Video object segmentation using space-time memory networks. In *Proceedings of the*

IEEE/CVF International Conference on Computer Vision, 2019, 9226–9235.

[19] Seong H, Hyun J, Kim E. Kernelized Memory Network for Video Object Segmentation. *arXiv preprint arXiv:2007.08270*, 2020.

[20] Zhang P, Hu L, Zhang B, Pan P, Yang Z, Ding Y, Wei Y, Yang Y, Seong H, Hyun J, et al.. Spatial Constrained Memory Network for Semi-supervised Video Object Segmentation. In *The 2020 DAVIS Challenge on Video Object Segmentation-CVPR Workshops*, 2020, 1–4.

[21] Chen LC, Papandreou G, Schroff F, Adam H. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[22] Liu P, Fu H, Ma H. An end-to-end convolutional network for joint detecting and denoising adversarial perturbations in vehicle classification. *Computational Visual Media*, 2021, 7(2): 217–227.

[23] Huo Y, Yoon Se. A survey on deep learning-based Monte Carlo denoising. *Computational Visual Media*, 2021: 1–17.

[24] Danon D, Arar M, Cohen-Or D, Shamir A. Image resizing by reconstruction from deep features. *Computational Visual Media*, 2021: 1–14.

[25] Liu X, Li C, Wong TT. Boundary-aware texture region segmentation from manga. *Computational Visual Media*, 2017, 3(1): 61–71.

[26] Chen Y, Pont-Tuset J, Montes A, Van Gool L. Blazingly fast video object segmentation with pixel-wise metric learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, 1189–1198.

[27] Khoreva A, Benenson R, Ilg E, Brox T, Schiele B. Lucid data dreaming for video object segmentation. *International Journal of Computer Vision*, 2019, 127(9): 1175–1197.

[28] Wang X, Jabri A, Efros AA. Learning correspondence from the cycle-consistency of time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, 2566–2576.

[29] Zhang ML, Zhou ZH. ML-KNN: A lazy learning approach to multi-label learning. *Pattern recognition*, 2007, 40(7): 2038–2048.

[30] Wang X, Girshick R, Gupta A, He K. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, 7794–7803.

[31] Liang Y, Li X, Jafari N, Chen Q. Video object segmentation with adaptive feature bank and uncertain-region refinement. *arXiv preprint arXiv:2010.07958*, 2020.

[32] Cheng HK, Tai YW, Tang CK. Rethinking Space-Time Networks with Improved Memory Coverage for Efficient Video Object Segmentation. *arXiv preprint arXiv:2106.05210*, 2021.

[33] Hu L, Zhang P, Zhang B, Pan P, Xu Y, Jin R. Learning Position and Target Consistency for Memory-based Video Object Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, 4144–4154.

[34] Xie H, Yao H, Zhou S, Zhang S, Sun W. Efficient Regional Memory Network for Video Object Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, 1286–1295.

[35] Tang L, Chen K, Wu C, Hong Y, Jia K, Yang ZX. Improving semantic analysis on point clouds via auxiliary supervision of local geometric priors. *IEEE Transactions on Cybernetics*, 2020.

[36] Yang ZX, Tang L, Zhang K, Wong PK. Multi-view cnn feature aggregation with elm auto-encoder for 3d shape recognition. *Cognitive Computation*, 2018, 10(6): 908–921.

[37] Perazzi F, Pont-Tuset J, McWilliams B, Van Gool L, Gross M, Sorkine-Hornung A. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, 724–732.

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

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

[40] Bao L, Wu B, Liu W. CNN in MRF: Video object segmentation via inference in a CNN-based higher-order spatio-temporal MRF. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, 5977–5986.

[41] Luiten J, Voigtlaender P, Leibe B. PReMVOS: Proposal-generation, Refinement and Merging for Video Object Segmentation. *arXiv preprint arXiv:1807.09190*, 2018.

[42] Li Y, Wen L, Chang MC, Lyu S. Graph-to-Graph Energy Minimization for Video Object Segmentation. In *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2019, 1–8.

[43] Wang Q, Zhang L, Bertinetto L, Hu W, Torr PH. Fast online object tracking and segmentation: A unifying approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, 1328–1338.

[44] Hu YT, Huang JB, Schwing AG. Videomatch: Matching based video object segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, 2018, 54–70.

[45] Johnander J, Danelljan M, Brissman E, Khan FS, Felsberg M. A generative appearance model for end-to-end video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, 8953–8962.

[46] Lin TY, Maire M, Belongie S, Bourdev L, Girshick R, Hays J, Perona P, Ramanan D, Zitnick CL, Dollár P. Microsoft COCO: Common Objects in Context. *arXiv preprint arXiv:1405.0312*, 2014.

[47] Kingma DP, Ba J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[48] Ventura C, Bellver M, Girbau A, Salvador A, Marques F, Giro-i Nieto X. Rvos: End-to-end recurrent network for video object

segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, 5277–5286.

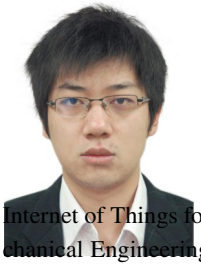
[49] Xu N, Yang L, Fan Y, Yang J, Yue D, Liang Y, Price B, Cohen S, Huang T. Youtube-vos: Sequence-to-sequence video object segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, 2018, 585–601.

[50] Wehrwein S, Szeliski R. Video segmentation with background motion models. In *BMVC*, volume 245, 2017, 246.

[51] Voigtlaender P, Luiten J, Leibe B. Boltvos: Box-level tracking for video object segmentation. *arXiv preprint arXiv:1904.04552*, 2019.

[52] Lin H, Qi X, Jia J. Agss-vos: Attention guided single-shot video object segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, 3949–3957.

Author biographies



Yadang Chen received a Ph.D. in Soft Engineering from the University of Macau in 2016. He is currently an associate professor in Nanjing University of Information Science and Technology. From 2019-2020, he was a Postdoctor in the State Key Laboratory of the Internet of Things for Smart Cities and Department of Electromechanical Engineering, University of Macau. From 2017-2018, he was a Visiting Scholar with Michigan State University. His main research interests include video segmentation, video enhancement, video editing, and augmented reality.

Duolin Wang is now a master's student in Software Engineering at Nanjing University of Information Science and Technology, China. His research interests include deep learning and computer vision.

Zhiguo Chen received M.S. and Ph.D. degrees from the division of Internet and Multimedia Engineering at Konkuk University, Seoul, Korea, in 2014 and 2019, respectively. He is an Associate Professor with the School of Computer Science, Nanjing University of Information Science & Technology. His research interests include artificial intelligence, information security and cloud computing.

Zhi-Xin Yang obtained his Ph.D. in Industrial Engineering and Engineering Management from Hong Kong University of Science and Technology. He is currently an Associate Professor in the State Key Laboratory of Internet of Things for Smart Cities, Faculty of Science and Technology, and the Director of Research Service and Knowledge Transfer Office both at the University of Macau. His current research interests include fault diagnosis and prognosis, machine learning, and computer vision-based robotics.

Enhua Wu completed his B.Sc. studies in Tsinghua University, and received his Ph.D. degree from Department of Computer Science, University of Manchester, UK in 1984. He has been working at the State Key Lab. of Computer Science, Institute of Software, Chinese Academy of Sciences, since 1985, as a director of the Research Dept of Fundamental Theory and Advanced Technology, IOS until 2001. He has also been a full professor of the University of Macau (UM) since 1997, where he is now the Associate Dean of the Faculty of Science and Technology.