


Article

Mitigating Distractor Challenges in Video Object Segmentation through Shape and Motion Cues

Jidong Peng¹, Yibing Zhao², Dingwei Zhang² and Yadang Chen^{2,*} ¹ Nanjing Research Institute of Electronic Engineering, Nanjing 210001, China; cetc28_mail@cetc.com.cn² School of Computer Science, Nanjing University of Information Science and Technology, Nanjing 210044, China; 20211249478@nuist.edu.cn (Y.Z.); 20211221049@nuist.edu.cn (D.Z.)

* Correspondence: adamchen@nuist.edu.cn

Abstract: The purpose of semi-supervised video object segmentation (VOS) is to predict and generate object masks in subsequent video frames after being provided with the initial frame's object mask. Currently, mainstream methods leverage historical frame information for enhancing the network's performance. However, this approach faces the following issues: (1) They often overlook important shape information, leading to decreased accuracy in segmenting object-edge areas. (2) They often use pixel-level motion estimation to guide the matching for addressing distractor objects. However, this brings heavy computation costs and struggle against occlusion or fast/blurred motion. For the first problem, this paper introduces an object shape extraction module that exploits both the high-level and low-level features to obtain object shape information, by which the shape information can be used to further refine the predicted masks. For the second problem, this paper introduces a novel object-level motion prediction module, in which it stores the representative motion features during the training stage, and predicts the object motion by retrieving them during the inference stage. We evaluate our method on benchmark datasets compared with recent state-of-the-art methods, and the results demonstrate the effectiveness of the proposed method.

Keywords: video object segmentation; object motion; spatio-temporal constraints; memory network



Citation: Peng, J.; Zhao, Y.; Zhang, D.; Chen Y. Mitigating Distractor Challenges in Video Object Segmentation through Shape and Motion Cues. *Appl. Sci.* **2024**, *14*, 2002. <https://doi.org/10.3390/app14052002>

Academic Editor: Andrea Prati

Received: 11 December 2023

Revised: 16 February 2024

Accepted: 26 February 2024

Published: 28 February 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Video object segmentation (VOS) is a fundamental task in video understanding, including many industrial information applications such as autonomous driving. In this paper, we will focus on video object segmentation under semi-supervised learning, i.e., inferring the object mask in the rest of the video frames given the first frame's object mask.

During recent years, match-based methods [1–11] have become the mainstream. This approach makes full use of the segmented historical frames, encoding them into embedded features stored in a memory bank to assist in the segmentation of subsequent frames. The most recent matching-based VOS methods focus on addressing the memory cost problem [5,6] and the distractor objects problem [7–9], both caused by the global matching mechanism. Though they have achieved good performance, some issues still exist in those methods: (1) The important shape information is often ignored, leading to a decrease in accuracy for the segmentation around object-edge regions. (2) For addressing distractor objects, pixel-level motion estimation is often involved along the video to restrict the global matching; this brings heavy computation costs and struggles against occlusion or fast/blurred motion.

For the first problem, this paper introduces a lightweight shape extraction module aimed at obtaining object shape information. Specifically, a shape extraction head is added following the low-level features outputted by the network to focus on extracting shape-related features of the object. Simultaneously, high-level features from the network are also used to denoise the activations in the shape extraction module, eliminating irrelevant

noise from the low-level features. Thus, we use the extracted shape information to further optimize the segmentation results.

For the second problem, we introduce a novel object-level motion prediction module beyond pixel level. To this end, we propose to use a motion memory bank to store the object-level representative motion features learned from the training stage, which can be used to predict the motion information during the inference stage. Specifically, a recurrent neural network [12] is used to extract object-level motion features during the training stage, then we retrieve the stored motion features for estimating the object motion at the inference stage in order to predict the position of the object in the next frame. This allows our method to achieve both high efficiency and strong robustness for handling distractor objects or fast/blurry motion.

Figure 1 illustrates the improvement in segmentation accuracy using our method. In summary, the main contributions of this work are as follows:

- We design a lightweight object shape extraction module that exploits both the high-level and low-level features to obtain object shape information. And the shape information is then used to further refine the predicted masks.
- We introduce a novel object-level motion prediction module that stores the representative motion features during the training stage, and predicts the object motion by retrieving them during the inference stage, by which our method achieves high efficiency and strong robustness.
- The proposed method performs well compared to recent state-of-the-art methods on benchmark datasets. An accuracy of 85.2 $\mathcal{J}\&\mathcal{F}$ was obtained on the DAVIS 2017 [13] validation set. An accuracy of 84.5 $\mathcal{J}\&\mathcal{F}$ was obtained on the YouTube-VOS [14] 2019 validation set.

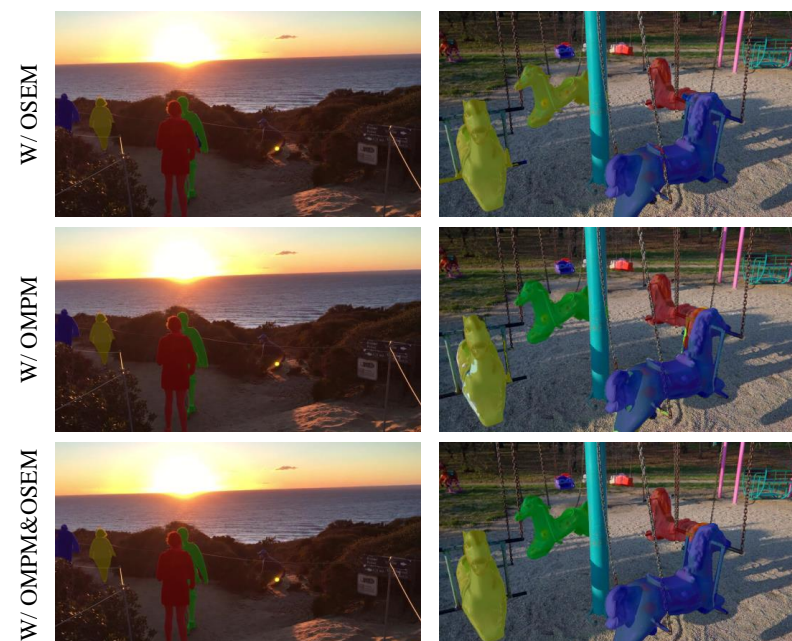


Figure 1. This figure shows the significant effects of the object motion prediction module (OMPM) and object shape extraction module (OSEM) that we propose.

2. Related Works

2.1. Semi-Supervised VOS

The latest video object segmentation (VOS) solutions can be roughly categorized into three types: (1) Methods based on online learning [15] employ online fine-tuning techniques to adjust pretrained parameters during inference, adapting to specific objects. While achieving some success, they are often sensitive to hyperparameters and inefficient. (2) Propagation-based methods [16] consider video object segmentation as the task of

predicting the current frame based on the previous frame's mask. However, they solely rely on the mask from the preceding frame, making them susceptible to occlusion and error accumulation. (3) Matching-based methods [1–9] utilize the first annotated frame and previously inferred frames to construct an explicit object model, enabling the matching and classification of query pixels. A significant milestone in this field is STM [1], which introduces external memory to explicitly and persistently store representations and masks of past frames, leveraging the information from previous frames for more accurate segmentation.

2.2. Matching-Based VOS Networks

Recent research in matching-based video object segmentation (VOS) [1–9,17–23] has primarily focused on improving network structures, including building more efficient memory [5,6], adopting local matching [7–9], and incorporating background context [3,4]. Among them, STM [1] is the pioneer in the field of video object segmentation, which first proposed the design of the memory bank, enabling VOS methods to effectively utilize the temporal information in video sequences. STCN [2] optimized the memory retrieval strategy based on STM, improving the performance. RDE [5] improved the memory update mechanism, avoiding the problem of memory expansion over time. Miles [24] borrowed the technique of knowledge distillation, making the video object segmentation method more suitable for running on mobile devices. However, there are still challenges that need to be addressed. On the one hand, existing methods often overlook crucial shape information when segmenting object edges, leading to reduced accuracy in edge-region segmentation. Inspired by [25], this paper proposes an object shape extraction module to enhance the network's accuracy in segmenting object edges. On the other hand, the common approach to addressing misalignment of similar objects primarily relies on pixel-level motion information. As a result, it often proves inaccurate in cases involving occlusion and fast-moving objects. Inspired by [20], this paper introduces an object motion prediction module, which relies primarily on object-level motion information not constrained by image features. This performs better in addressing challenges such as occlusion and rapid object movement compared to pixel-level motion.

2.3. VOS Network with Spatio-Temporal Constraint

There have been many previous attempts [8–11,18,26] to mitigate the distractor objects problem. Kernelized memory networks [9] alleviate the issue of misalignment by introducing Gaussian kernels to reduce the non-locality of STM [1]. KMMN [18] introduces a top-k guided and a kernel-guided memory matching module to achieve efficient memory matching. SCM [26] introduces a spatially constrained module that utilizes spatial prior values from previous predictions. This helps to eliminate appearance confusion and false predictions. RMNet [8] introduces a pixel-level motion estimation method to reduce match consumption and distractor objects. Previous methods often use pixel-level motion estimation to guide the matching for addressing distractor objects. However, this brings heavy computation costs and struggles against occlusion or fast/blurry motion. Unlike pixel-level motion prediction, the object-level motion prediction proposed in this paper can effectively deal with fast/blurry motion and occlusion scene challenges.

3. Methods

This section introduces the main components of our method, including the baseline network in Section 3.1, the object shape extraction module (OSEM) in Section 3.2, and the object motion prediction module (OMPM) in Section 3.3. The object shape extraction module can enhance the network's segmentation performance in the object overlap area, and the object motion prediction module can improve the network's robustness to distractor objects. Figure 1 shows the intuitive effects of these two modules. In the first row, using only the segmentation results from OSEM demonstrates high accuracy in edge-region segmentation but comes with the issue of incorrectly segmenting distractor objects. On the other hand, in the second row, relying solely on OMPM for segmentation noticeably

reduces the problem of distractor objects but leads to a decrease in accuracy in edge-region segmentation. In the third row, by simultaneously applying OPM and OSEM, we clearly observe the synergistic effects of both, substantially alleviating the issue of distractor objects and successfully enhancing the accuracy of edge-region segmentation.

3.1. Network Overview

3.1.1. Encoders

Following the STCN [2], the image encoder plays a crucial role in extracting image features from a query frame with dimensions $H \times W$. Additionally, we utilize a mask encoder to encode the mask, and then store it in the memory bank. Similar to the STCN [2], we select the res4 features of the ResNet50 [27] model as the backbone features for the image encoder, with a stride of 16, while excluding res5. For the mask encoder, we opt for the ResNet18 [27] model for implementation. Subsequently, we generate key and value embeddings using two projection headers, referred to as key ($\mathbf{k} \in \frac{H}{16} \times \frac{W}{16} \times \mathbb{R}^{C_k}$) and value ($\mathbf{v} \in \frac{H}{16} \times \frac{W}{16} \times \mathbb{R}^{C_v}$). In this article, we set C_k to 64 and C_v to 512.

3.1.2. Memory Reading

According to [2], utilizing both an image encoder and mask encoder alongside a single query frame and T memory frames enables the extraction of the query key \mathbf{k}^Q with dimensions $HW \times \mathbb{R}^{C_k}$, memory value \mathbf{v}^M with dimensions $THW \times \mathbb{R}^{C_v}$, and memory key \mathbf{k}^M with dimensions $THW \times \mathbb{R}^{C_k}$.

The affinity matrix, denoted as \mathbf{S}^{L2} , and its softmax-normalized counterpart \mathbf{W} are computed, where both matrices \mathbf{S}^{L2} and \mathbf{W} belong to the space $\mathbb{R}^{THW \times HW}$. The formulations for \mathbf{S}^{L2} and \mathbf{W} are outlined below:

$$\mathbf{S}_{ij}^{L2} = -\|\mathbf{k}_i^M - \mathbf{k}_j^Q\|_2^2, \quad (1)$$

$$\mathbf{W}_{ij} = \frac{\exp(\mathbf{S}_{ij}^{L2})}{\sum_n (\exp(\mathbf{S}_{nj}^{L2}))}, \quad (2)$$

In this context, \mathbf{S}^{L2} indicates the application of the L2 function for assessing similarity distances, while the feature vector located at position j is represented by k_j .

3.1.3. Decoder

Our decoder's architecture closely resembles that of the STM [1]. The features are step by step processed and upsampled in a procedure that increases the scale by a factor of 2. The last layer in the decoder predicts a mask with a stride of 4 and, when necessary, we employ bilinear upsampling to align with the resolution of the input image. In scenarios involving multiple objects, we adopt a processing strategy consistent with that described in [28].

3.2. Object Shape Extraction Module

Mainstream methods often disregard essential shape information when dealing with object–edge segmentation, resulting in decreased accuracy in segmenting object-edge regions. Inspired by [25], we design an object shape extraction module (OSEM) for video object segmentation. The module processes shape information in the form of object edges, using a gated convolutional layer [25] and a loss function to limit its processing to only boundary-related information. By integrating information read from the backbone network memory with shape information, we generate segmentation results that emphasize the edges of the object. Finally, we utilize shape information to further refine the segmentation results, generating the ultimate segmentation results. Parts (a) in Figure 2 display the schematic structure of our shape extraction module, while Figure 3 shows the shape

information outputted by different gated convolutional layers. It can be seen that the shape information becomes clearer and clearer through the shape extraction module.

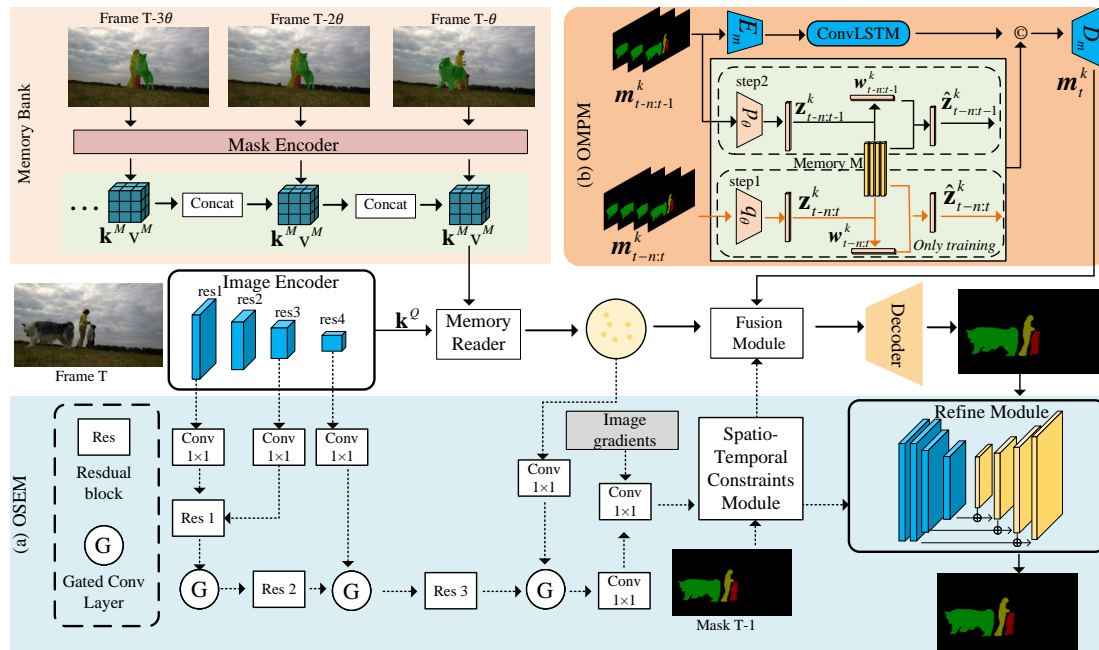


Figure 2. The main framework of this work. (a) The object shape extraction module takes as input the image gradients, the outputs of the first, third, and the last convolutional layers of the backbone network, and the memory readout results from the backbone network. It then generates boundary information for the object as output. Subsequently, we feed this boundary information into a spatio-temporal constraints module to obtain more accurate localized boundary information; Figure 4 illustrates the specific implementation of the spatio-temporal constraints module. This refined boundary information is then used to refine the masks generated by the backbone network. (b) The object mask m^k represents the positional information of the k -th object. Our training process is divided into two stages. During the training process, we have access to the ground truth labels $m^k_{t-n:t}$ for the object mask, and we only update the memory bank (M) in the first step. In the inference stage, we only perform the second step, keeping the memory bank unchanged. We utilize the masks obtained from the network inference for frames $t - n$ to $t - 1$ to predict the subsequent mask for frame t .

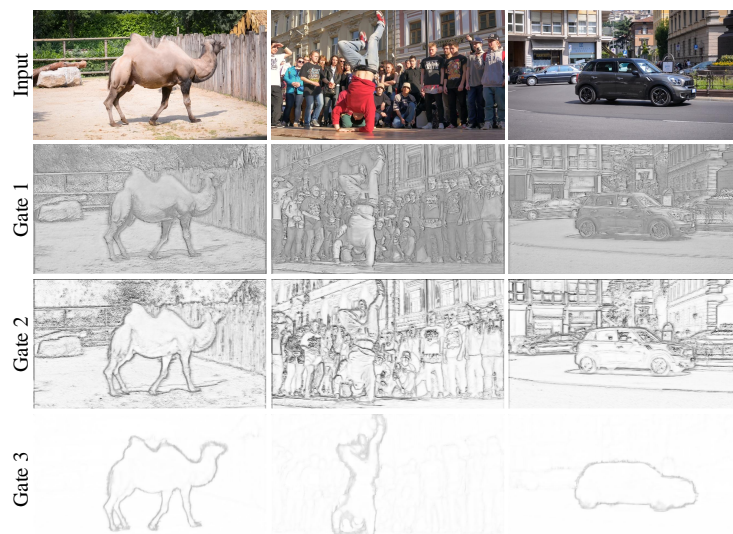


Figure 3. This figure shows the shape information during the work of the shape extraction module. The extracted shape information is progressively clearer. Gate represents our gated convolutional layer.

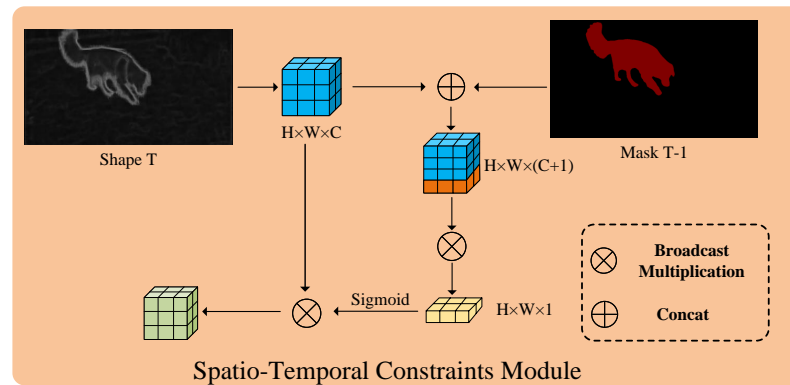


Figure 4. Implementation of shape spatio-temporal constraints module.

3.2.1. Utilizing Backbone Network Information

This module takes image gradients, the outputs of the first, third, and the last convolutional layers of the backbone network, as well as the memory reader results from the backbone network as inputs, and then generates the boundary information of the object as output. The architecture of this module consists of a series of residual blocks interleaved with gated convolutional layers (GCLs) [25]. The presence of GCLs [25] ensures that the OSEM processes only information related to the object's shape. Since we can obtain precise object shape information from the ground truth object segmentation mask, we can train the OSEM using cross-entropy loss.

3.2.2. Gated Convolutional Layer

We use a gated convolutional layer [25] (GCL) to facilitate the flow of information from the backbone network to the shape extraction module to enhance the impact of object segmentation and object boundaries on the task. We use high-level features from the network to denoise the activations in the shape extraction module, eliminating irrelevant noise from the low-level features. As shown in Figure 3, it can be observed that after the gated convolutional layer (GCL), shape noise unrelated to the object is gradually eliminated.

3.2.3. Shape Spatio-Temporal Constraints

Since the video object itself has a spatio-temporal constraint, i.e., the location of the object is approximately the same in two sequential frames, inspired by [26], this paper utilizes the mask information from the previous frame to make the network more focused on the shape of the object region. The prediction mask of the previous frame has a shape of $H \times W$ and is connected to the current shape map, which has a size of $H \times W \times 1$, forming a feature map of $H \times W \times 2$. Subsequently, a convolution layer with a size of 3×3 and the Sigmoid function generate the object's rough location prior with a size of $H \times W \times 1$. Finally, the shape map features are multiplied by the positional prior to obtain the spatially and temporally constrained shape map. Next, we integrate the spatially and temporally constrained shape map with the memory readout features in a residual manner. Figure 4 illustrates the specific implementation of this module.

To further enhance performance by leveraging shape information, especially at the boundaries, inspired by [26] we designed a shape-based refinement module. We used the feature maps before soft aggregation and the object boundary map as inputs to the refinement module. Next, we resampled them with a 3×3 convolutional layer and the ReLU function, followed by upsampling to the original resolution and once again merging them through soft aggregation.

3.3. Object Motion Prediction Module

Mainstream methods to address distractor objects mainly consider pixel-level motion. This often leads to errors, especially when dealing with object occlusion or fast motion. Inspired by [20], we introduce a novel object motion prediction module (OMPM) beyond

the pixel level. Part (b) in Figure 2 displays the structure of this module. Figure 5 shows the difference between our method and the pixel-level method. In video frames (I_t), where t denotes the t -th frame, we use binary masks to represent the positional information of objects. The binary mask is denoted as \mathbf{m}_t^k , where k represents the k -th object. This module learns the motion from \mathbf{m}_{t-n}^k to \mathbf{m}_{t-1}^k to predict the positional information of the object \mathbf{m}_t^k . In our scenario, given any number of frames, the module predicts one frame forward ($\mathbf{m}_t^k = OMPM(\mathbf{m}_{t-n:t-1}^k)$), where $OMPM$ represents the object motion prediction module.

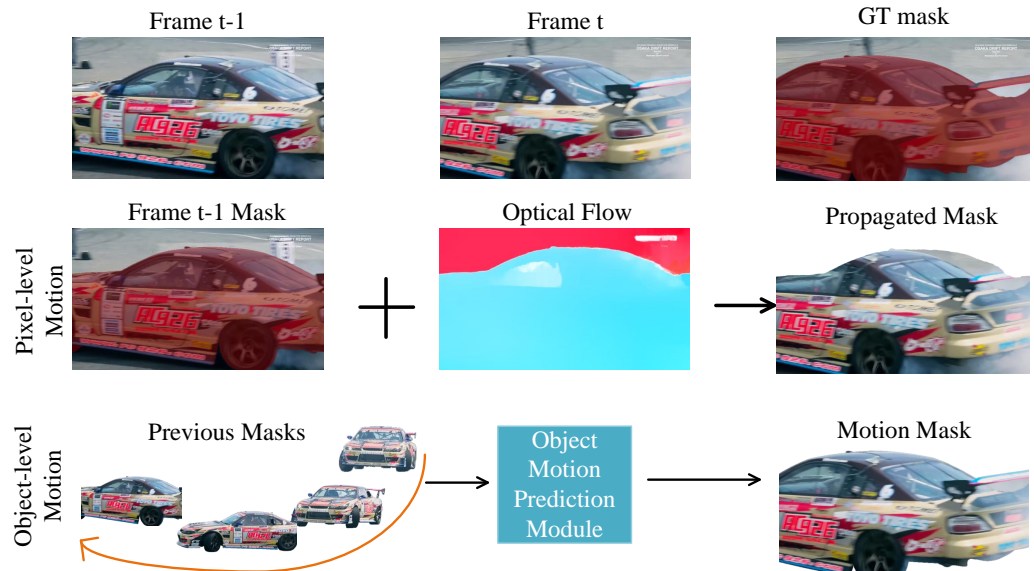


Figure 5. This figure shows a significant difference between pixel-level motion and object-level motion. In the first row, we present the previous frame’s image, the current frame’s image, and the ground truth mask, separately. In the second row, we show pixel-level motion, primarily based on the previous frame’s mask and precomputed optical flow information, to infer the current frame object’s position. However, this method is prone to failure when the object undergoes rapid motion. In contrast, the object-level motion depicted in the third row involves extracting the object motion trajectory directly from the previous object mask, predicting the accurate position of the object in the current frame without the need for precomputed optical flow information. This approach not only remains effective in situations involving occlusion and rapid motion but also eliminates the need for precomputed steps, making it more efficient.

Inspired by conditional variational autoencoder (CVAE) dynamics modeling [29], in the training process, an encoder is utilized to understand the motion signatures extracted from the Gaussian distribution of latent variables \mathbf{z}_t used for prediction. Meanwhile, we store key motion signatures in the memory bank \mathbf{M} . In the inference process, these stored patterns refine incomplete information from $(\mathbf{m}_{t-n:t-1}^k)$, incorporating object-level motion details to predict the next frame \mathbf{m}_t^k .

3.3.1. Motion Signature Memory Bank

During training, by accessing \mathbf{m}_t^k , we utilize q_ϕ to learn the inference of motion signature $\mathbf{z}_{t-n:t}^k \in \mathbb{R}^l$. The resulting $\mathbf{z}_{t-n:t}^k \in \mathbb{R}^l$ is stored in $\mathbf{M} \in \mathbb{R}^{c \times l}$, with similarity weights $w_{1:t}^k \in \mathbb{R}^c$ based on the standard method [30].

The softmax function computes $w_{1:t,i}^k$ for each memory vector $v_i \in \mathbb{R}^l$.

$$w_{1:t,i}^k = \frac{\exp\{S_{\cos}(\mathbf{v}_i, \mathbf{z}_{t-n:t}^k)\}}{\sum_{j=1}^c \exp\{S_{\cos}(\mathbf{v}_j, \mathbf{z}_{t-n:t}^k)\}}, \quad (3)$$

where w_t^k is a non-negative vector, enabling memory bank access. $S_{\cos}(a, b)$ represents the calculation of the cosine similarity between a and b . We can obtain memory features related to a latent variable \mathbf{z} of length l :

$$\hat{\mathbf{z}} = \mathbf{wM} = \sum_{i=1}^c w_i \mathbf{v}_i, \quad (4)$$

The memory bank parameters are updated through backpropagation. Equations (3) and (4) offer an explicit means to store and retrieve various motion signatures, including $\mathbf{z}_{t-n:t}^k$.

Using the trained memory bank, we take $\mathbf{m}_{t-n:t-1}^k$ as input and obtain the motion signature $\mathbf{z}_{t-n:t-1}^k$ through the encoder p_θ . Accessing the memory bank through Equations (3) and (4), we obtain the motion signature $\hat{\mathbf{z}}_{t-n:t-1}^k$ corresponding to $\mathbf{z}_{t-n:t-1}^k$. During inference, we do not update the memory bank.

3.3.2. Motion Prediction Using Memory

We infer the object frame \mathbf{m}_t^k through the utilization of a network based on a recurrent neural network (RNN). Specifically, taking $\mathbf{m}_{t-n:t-1}^k$ as input, we extract features $\mathbf{f}_{t-n:t-1}^k$ using the mask encoder E_m and Conv-LSTM. Subsequently, concatenating these extracted features with motion signatures $\hat{\mathbf{z}}_{t-n:t-1}^k$ retrieved from the memory bank, we then input them into the mask decoder D_m to infer the \mathbf{m}_t^k . Our training process is divided into two stages. During the training process, we have access to the ground truth labels $\mathbf{m}_{t-n:t}^k$ for the object mask, and we only update the memory bank (\mathbf{M}) in the first step. Other training details are consistent with [20].

3.3.3. Motion Fusion

We concatenate the features from the motion prediction module and the features extracted from memory. These features undergo processing through the utilization of two ResBlocks [27] and a CBAM [31] block for enhanced refinement. The CBAM block [31] can adaptively adjust the weights of features, thereby enhancing the model's expressive power and generalization ability. With the model fusion module, our model can better fuse the backbone features with the predicted motion information features so that we can utilize the information at different levels in the model, and thus, obtain more accurate prediction results.

4. Results

4.1. Datasets and Metrics

4.1.1. DAVIS

DAVIS 2016 [32] is a benchmark dataset for single-object video segmentation, comprising a total of 50 videos. Among these, 30 are used as the training set, and 20 are allocated to the validation set. DAVIS 2017 [13] serves as a benchmark dataset for video multi-object segmentation, including a total of 120 videos. Its validation set and test-dev set each consist of 30 videos.

4.1.2. YouTube-VOS

YouTube-VOS 2019 [14] is a benchmark for multi-object video segmentation, providing a total of 3924 videos, with 507 designated for the validation set and 3471 for the training set. The validation set includes categories not present in the training set, allowing for a further assessment of the model's generalization performance.

4.1.3. Metrics

For the DAVIS dataset [13], this paper employs the region similarity \mathcal{J} . To assess region-based segmentation similarity, specifically the count of mislabeled pixels, the Jaccard index \mathcal{J} is utilized. Given the output segmentation M and the corresponding ground

truth mask G , it is formally defined as $\mathcal{J} = \frac{|M \cap G|}{|M \cup G|}$. Contour accuracy, denoted as \mathcal{F} , is evaluated through contour-based precision and recall, P_c and R_c . This is formally defined as $\mathcal{F} = \frac{2 \cdot P_c \cdot R_c}{P_c + R_c}$. The $\mathcal{J} \& \mathcal{F}$, denoting the average of the region similarity metric (\mathcal{J}) and contour accuracy (\mathcal{F}) for the DAVIS dataset, is formally defined as $\mathcal{J} \& \mathcal{F} = \frac{\mathcal{J} + \mathcal{F}}{2}$. Unlike the DAVIS [13] dataset, in YouTube-VOS 2019 [14] classes are categorized into seen and unseen categories. The final result (\mathcal{G}) is the average of four metrics: \mathcal{J} for seen categories (\mathcal{J}_S), \mathcal{F} for seen categories (\mathcal{F}_S), \mathcal{J} for unseen categories (\mathcal{J}_U), and \mathcal{F} for unseen categories (\mathcal{F}_U).

4.2. Implementation Details

During the training phase, we opted for the PyTorch [33] framework and employed the Adam optimizer [34] for model optimization. Regarding the learning rate, we set the initial learning rate to 1×10^{-5} and adopted a warm-up strategy, starting with a smaller learning rate and gradually increasing it. This strategy aimed to promote more robust convergence of the model. Similar to prior research [2], we applied deformation operations to static images by transforming one image into three different ones, thereby generating virtual videos for pretraining [35]. For the main training on the YouTube-VOS [14] and DAVIS [13] datasets, we utilized the same data processing methods as described in Ref. [36]. We resized all input image masks to 384×384 with padding to preserve their aspect ratios. Similar to STM [1], batch normalization layers were frozen during the training process. We supervised the training of the shape extraction module by generating real mask edges from ground truth object masks. The training approach for the motion prediction module remained consistent with Ref. [20]. Additionally, we adopted the same training frame sampling method as MiVOS [37], initially using a smaller sampling interval, gradually increasing it to 25, and finally restoring it to the original value as training approached completion. Similar strategies were employed for the shape space constraint module's sampling of the previous frame mask. In the inference phase, we utilized the first k filters [37], setting k to 80 on the DAVIS 2017 [13] validation set and 20 on other datasets. Additionally, we employed kernelized memory reading [9], with a kernel size of 7. Memory storage operations were performed every five frames, and no temporary frames were stored. We used the same implementation as STCN [2] for our backbone network, and the gated convolutional layer implementation in the object shape extraction module was the same as in [25]. In our object motion prediction module, the mask encoder used two 2D-Conv layers, followed by 3-layer Conv-LSTMs. The encoders (p_θ and q_ϕ) employed three 3D-Conv layers. For the mask decoder, we used three 2D-DeConv layers. During both the training and inference phases, we leveraged automatic mixed precision in PyTorch [33] to accelerate the training and inference processes.

4.3. Comparison with Other Methods

4.3.1. Quantitative Comparison

Table 1 shows the comparison of our method on the DAVIS 2017 [13] validation set with other methods. On the DAVIS 2017 [13] validation set we obtained competitive results, achieving a $\mathcal{J} \& \mathcal{F}$ score of 85.2. Table 2 shows the comparison of our method with the other method on the YouTube-VOS 2019 [14] validation set, where the performance of our model improves by 0.8% compared to our baseline model [2], obtaining a $\mathcal{J} \& \mathcal{F}$ score of 83.5. Table 3 shows the comparison of our method with the other method on the DAVIS 2017 [13] test-dev set, where the performance of our model improves by 1.4% compared to our baseline model [2], obtaining a $\mathcal{J} \& \mathcal{F}$ score of 77.5. Table 4 shows the comparison of our method with other methods on the DAVIS 2016 [32] validation set. Although our method performed lower than our baseline method STCN [2] (accuracy lower by 0.2) on the DAVIS 2017 [13] validation set, it significantly surpassed the baseline method [2] on the DAVIS 2017 [13] test-dev set and the YouTube-VOS 2019 [14] validation set (with accuracies higher by 1.4 and 0.8, respectively). It is worth noting that the segmentation difficulty is relatively lower in the DAVIS 2017 [13] validation set compared to the DAVIS

2017 [13] test-dev set and the YouTube-VOS 2019 [14] validation set. In practice, we found that achieving high accuracy on the DAVIS 2017 [13] test-dev set and the YouTube-VOS 2019 [14] dataset better reflects the generalization performance of our method. For instance, in STCN [2], the authors increased accuracy on the DAVIS 2017 [13] test-dev set (76.1→77.8) and YouTube-VOS 2019 [14] validation set (82.7→84.2) substantially by using additional training data, but the accuracy on the DAVIS 2017 [13] validation set decreased (85.4→85.3).

Table 1. Comparison of different methods on the DAVIS 2017 validation set.

Method	OL	M	$\mathcal{J} \& \mathcal{F}$	\mathcal{J}	\mathcal{F}
OSMN [38]	✓		54.8	52.5	57.1
RGMR [28]			66.7	64.8	68.6
RVOS [39]			50.3	48.0	52.6
Track-Seg [16]			72.3	68.6	76.0
PReMVOS [40]	✓		77.8	73.9	81.7
TVOS [41]			72.3	69.9	74.7
GC [6]		✓	71.4	69.3	73.5
SwiftNet [42]		✓	81.1	78.3	83.9
STM [1]		✓	81.8	79.2	84.3
GraphMem [43]		✓	82.8	80.2	85.2
MiVOS [37]		✓	83.3	80.6	85.9
CFBI [3]			81.9	79.1	84.6
KMN [9]		✓	82.8	80.0	85.6
RMNet [8]		✓	83.5	81.0	86.0
LWL [44]		✓	81.6	79.1	84.1
CFBI+ [4]			82.9	80.1	85.7
LCM [7]		✓	83.5	80.5	86.5
STCN [2]		✓	85.4	82.2	88.6
RDE [5]		✓	84.2	80.8	87.5
SCE [45]		✓	84.2	80.8	87.5
Miles [24]		✓	83.7	80.2	87.1
InstMove [20]		✓	85.1	82.3	87.9
Ours		✓	85.2	82.3	88.0

OL stands for online fine-tuning-based approach. M stands for memory-based approach. The ✓ symbol represents "yes".

Table 2. Results on the YouTube-VOS 2019 validation set.

Method	M	\mathcal{G}	\mathcal{J}_s	\mathcal{F}_s	\mathcal{J}_u	\mathcal{F}_u
STM [1]	✓	79.2	79.6	83.6	73.0	80.6
MiVOS [37]	✓	82.4	80.6	84.7	78.2	85.9
CFBI [3]		81.0	80.6	85.1	75.2	83.0
SST [46]		81.8	80.9	-	76.6	-
STCN [2]	✓	82.7	81.1	85.4	78.2	85.9
RDE [5]	✓	81.9	81.1	85.5	76.2	84.8
SCE [45]	✓	81.6	81.3	85.7	75.0	83.4
Miles [24]	✓	82.3	81.6	86.0	76.3	85.2
InstMove [20]	✓	83.4	82.5	86.9	77.9	86.0
Ours	✓	83.5	82.4	87.0	78.3	86.3

M stands for memory-based approach. The ✓ symbol represents "yes".

Table 3. Results on the DAVIS 2017 test-dev set.

Method	OL	M	$\mathcal{J}\&\mathcal{F}$	\mathcal{J}	\mathcal{F}
OSMN [38]	✓		41.3	37.3	44.9
OnAVOS [47]	✓		56.5	53.4	59.6
FEELVOS [48]			57.8	55.2	60.5
PReMVOS [40]	✓		71.6	67.5	75.7
STM [1]		✓	72.2	69.3	75.2
RMNet [8]		✓	75.0	71.	78.1
CFBI [3]			76.6	73.0	80.1
KMN [9]		✓	77.2	74.1	80.3
CFBI+ [4]			78.0	74.4	81.6
LCM [7]		✓	78.1	74.4	81.8
STCN [2]		✓	76.1	72.6	79.6
Ours		✓	77.5	74.0	81.0

OL stands for online fine-tuning-based approach. M stands for memory-based approach. The ✓ symbol represents "yes".

Table 4. Results on the DAVIS 2016 validation set.

Method	OL	M	$\mathcal{J}\&\mathcal{F}$	\mathcal{J}	\mathcal{F}
OSMN [38]	✓		73.5	74.0	72.9
MaskTrack [35]			77.6	79.7	75.4
FEELVOS [48]			81.7	81.1	82.2
RGMP [28]			81.8	81.5	82.0
Track-Seg [16]			83.1	82.6	83.6
OnAVOS [47]	✓		85.5	86.1	84.9
PReMVOS [40]	✓		86.8	84.9	88.6
GC [6]		✓	86.8	87.6	85.7
RMNet [8]		✓	88.8	88.9	88.7
STM [1]		✓	89.3	88.7	89.9
CFBI [3]			89.4	88.3	90.5
CFBI+ [4]			89.9	88.7	91.1
MiVOS [37]		✓	90.0	88.9	91.9
SwiftNet [42]		✓	90.4	90.5	90.3
KMN [9]		✓	90.5	89.5	91.5
LCM [7]		✓	90.7	91.4	89.9
STCN [2]		✓	91.6	90.8	92.5
RDE [5]		✓	91.1	89.7	92.5
SCE [45]		✓	90.8	91.7	89.9
Miles [24]		✓	90.6	89.7	91.6
Ours		✓	91.6	90.6	92.7

OL stands for online fine-tuning-based approach. M stands for memory-based approach. The ✓ symbol represents "yes".

4.3.2. Qualitative Comparison

Figure 6 compares the segmentation results of our method and the baseline method in similar object scenes. We have compared our method with the state-of-the-art methods, including STCN [2], AOTT [19], RDE [5], and MiVOS [37]. STCN serves as both a memory-based method and our baseline approach. AOTT [19] is a representative method based on Transformer, while RDE [5] is a typical approach that occupies fixed-size memory. MiVOS [37] is an improved version of the milestone method STM [1] within the category of memory-based approaches. From Figure 6, it is clearly evident that our method outperforms other techniques in terms of segmentation accuracy, particularly when dealing with similar objects or handling fine details at the boundaries of overlapping objects. Specifically, MiVOS [37] and STCN [2] erroneously segmented the car on the left in the background as the object in the first scene, and incorrectly segmented the standing person on the left in the second scene. In contrast, our method excelled in both of these scenarios. AOTT [19] and STCN [2] mistakenly segmented the background horse and person as the object in the third scene. In the fourth scene, AOTT [19] failed in the segmentation of the deer head on the right, indicating inadequate precision in object-edge handling. Although STCN [2] performed

well in object-edge-region processing, segmentation failure occurred due to the complete confusion of two objects. On the other hand, our approach, aided by the object motion prediction module and object shape extraction module, not only accurately segments the edge regions of the object but also exhibits strong robustness against challenges such as occlusion. In the fifth and final scenes, RDE [5] and STCN [2] suffer from background-foreground confusion in the object-edge regions and object confusion errors. In contrast, our method, leveraging shape information effectively, achieves higher segmentation accuracy.



Figure 6. Qualitative comparison of our method with baseline methods (STCN [2], AOTT [19], MiVOS [37], RDE [5]). We use the red dotted box to mark where the segmentation method fails. It can be observed that our method, compared to the baseline approach, alleviates the issue of mis-segmentation of similar objects and achieves higher accuracy in the segmentation of edge regions.

4.4. Ablation Study

We present a comprehensive analysis of the ablation experiments conducted on the DAVIS 2017 validation dataset, focusing on the efficacy of individual modules within our proposed model configuration. Our primary objective is to investigate the impact of each module on model performance, with a particular emphasis on the object-edge extraction and object motion prediction modules introduced in this paper. Unless otherwise specified, our method only undergoes the main training phase during ablation studies.

Table 5 summarizes the results obtained from these experiments, revealing critical insights into the contributions of various model components. Notably, our findings demonstrate that both the object-edge extraction module and the object motion prediction module significantly enhance the overall performance of the model. Moreover, the most substantial performance improvement is achieved when both modules are incorporated simultaneously. Table 6 presents the ablation experiments we conducted on the shape spatio-temporal

constraints module and shape-based refinement module in the object shape extraction module. The experiments indicate that both modules contribute to the overall performance improvement of the model, with the best results achieved when both are used simultaneously.

To vividly illustrate the impact of the modules proposed in this article on performance, we present in Figure 1 the influence of these modules on segmentation accuracy. In the first scenario, employing the object shape extraction module results in more accurate segmentation of human edges, especially in regions where two individuals overlap. In the second scenario, utilizing the object motion prediction module significantly improves the segmentation accuracy of the carousel, effectively mitigating object misalignment issues. Figure 7 demonstrates the visual effects of the shape extraction module. It is worth noting that our shape extraction module can precisely capture the edges of the object without being influenced by background edges. This accurate shape information simultaneously enhances the refinement mask accuracy of our shape-based recovery module.

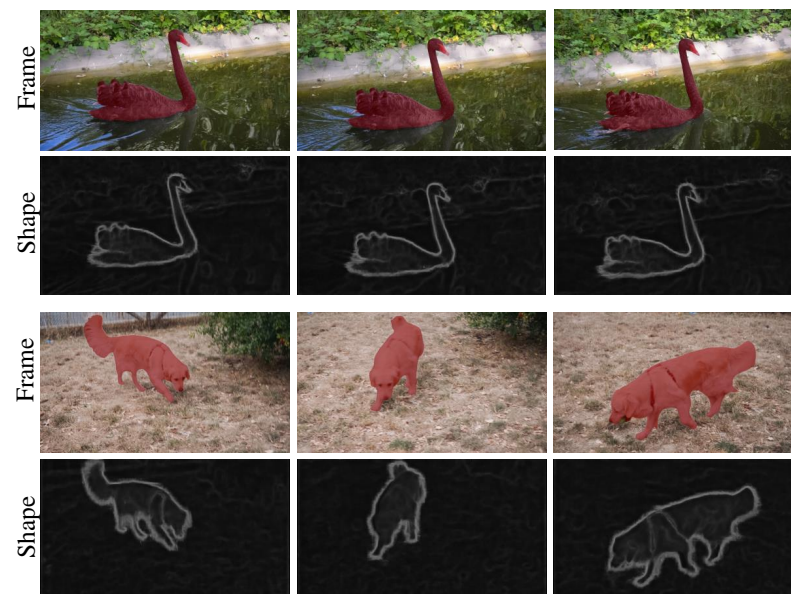


Figure 7. The visualization result of the shape map. It can be seen that our shape extraction module can accurately extract the object shape, and the extracted object shape can further improve the segmentation accuracy.

In Table 7, we compared the training results of using pretraining, main training, and a combination of both. Importantly, our study found that combining pretraining and main training methods significantly outperformed using either method alone in terms of model performance. Specifically, when conducting only pretraining, we achieved an accuracy of 76.9 $J&F$ score on the DAVIS 2017 [13] validation set. With only main training, we obtained an accuracy of 84.2 $J&F$ score, as main training aligns more closely with the data on the validation set. However, employing a strategy of pretraining followed by main training resulted in an accuracy of 85.2 $J&F$ score, a 1% improvement over the accuracy achieved with main training alone. We believe that pretraining allows our network to learn more generalized segmentation features.

Table 5. Ablation study.

Method	$J&F$	J	F
Without OSEM and OMPM	82.5	79.3	85.7
With OSEM	83.7	80.3	87.0
With OMPM	83.9	81.3	86.5
With OSEM and OMPM	84.2	81.3	87.1

OSEM stands for object shape extraction module. OMPM stands for object motion prediction module.

Table 6. Ablation study of object shape extraction module.

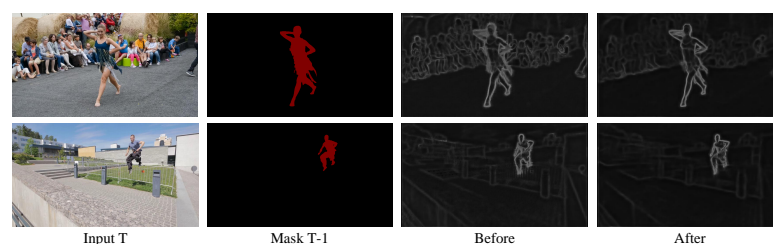
Method	$\mathcal{J}\&\mathcal{F}$	\mathcal{J}	\mathcal{F}
With shape spatio-temporal constraints	83.4	80.3	86.4
With shape-based refinement module	83.5	80.1	86.8
With both	83.7	80.3	87.0

Table 7. Effects of pretraining on static images/main training on the DAVIS 2017 [13] validation set.

Method	$\mathcal{J}\&\mathcal{F}$	\mathcal{J}	\mathcal{F}
Pretraining only	76.9	74.4	79.5
Main training only	84.2	81.3	87.1
Both	85.2	82.3	88.0

Figure 8 shows the visual display of the effectiveness of the spatio-temporal constraints module, in which “Input” denotes the input image, “Mask T-1” denotes the object mask of frame T-1, “Before” denotes the shape information before the spatial restriction module is processed, and “After” denotes the shape information after the processing. It can be seen that after the spatio-temporal constraints module is processed, it can effectively alleviate the background interference of the object shape. A visual example of the shape-based refinement module’s effects is shown in Figure 9. In this figure, “Input” denotes the input image, “Shape” denotes the object shape, “Before” denotes the mask before processing through the shape-based refinement module, and “After” denotes the mask after processing. The figure illustrates that by leveraging shape information effectively, our shape-based refinement module can significantly improve the segmentation accuracy at the edges of the object.

A comparison between the intermediate layer features of our approach and STCN [2] is presented in Figure 10. The points in the figure represent objects in a single frame of the video, with different colors indicating different objects. The first row shows the salsa video from DAVIS 2017 test-dev, featuring 10 similar objects and facing challenges such as occlusion and rapid movement, making segmentation difficult. The second row displays the sun-people video from DAVIS 2017 [13] test-dev, including four similar objects. Observing both scenarios, compared to STCN [2], our method’s extracted intermediate features exhibit larger feature distances between different objects, demonstrating higher discriminability for similar objects and effectively overcoming challenges such as occlusion and rapid movement. Figure 11 illustrates a comparison between the heatmap generated by our method and the baseline method STCN [2]. As shown in the figure, it can be observed that our method is less affected by background interference compared to the baseline STCN [2], focusing more on the object. Specifically, in the first and second scenarios, the attention of our baseline method is dispersed to irrelevant background, impacting its segmentation accuracy to some extent, whereas our method is less affected by background interference. In the second scenario, our baseline method [2] concentrates attention only on part of the object, while our method’s attention covers a broader region, resulting in higher segmentation accuracy, especially at the edges of the object.

**Figure 8.** The figure illustrates our ablation experiments on the spatio-temporal constraints module. It can be observed that the processing of the spatio-temporal constraints module effectively alleviates the interference of the background on the shape of the object.

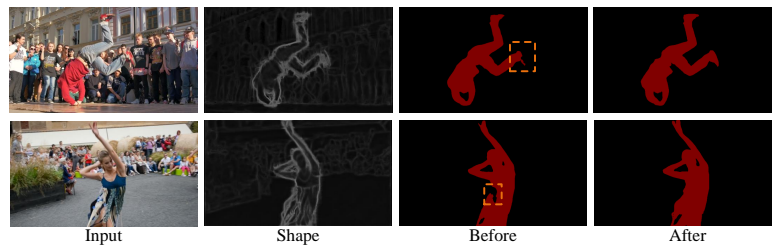


Figure 9. The figure illustrates the results of our ablation experiments on the shape-based refinement module. It can be observed that the shape-based refinement module effectively refines the mask, enhancing the segmentation accuracy in the object-edge regions. The orange box indicates the areas refined by our module.

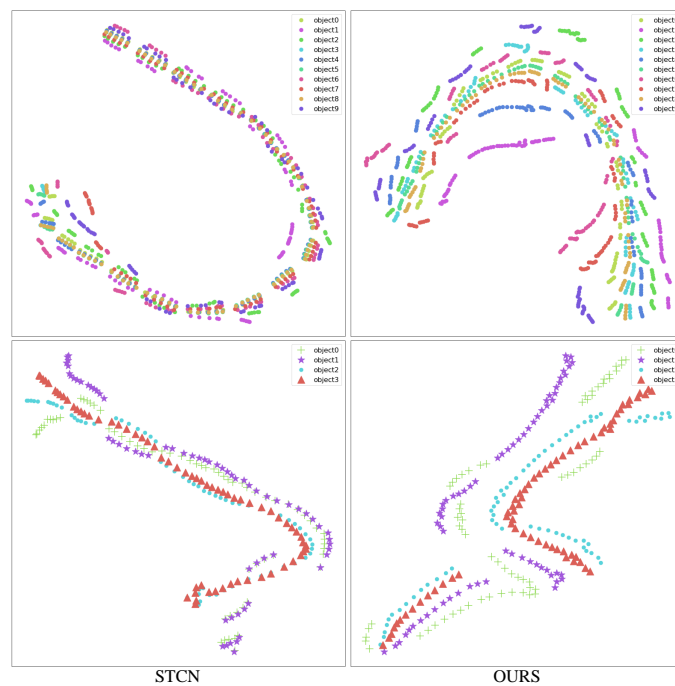


Figure 10. Comparison between the middle layer features of our method and the STCN [2]. The points in the figure represent individual frames in the video, and different colors indicate different objects. As can be seen from the figure, compared to STCN [2] the feature vectors of the distractor objects of our method are more discretized.

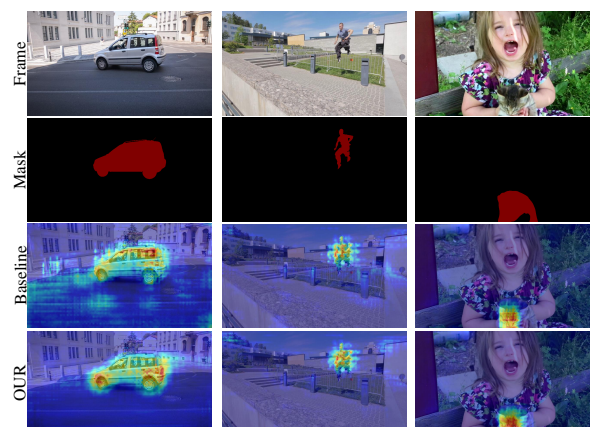


Figure 11. A comparison of the heatmap generated by our method with STCN [2]. As shown in the figure, it can be observed that our method pays more attention to the object region compared to STCN [2].

5. Discussion and Conclusions

This paper introduces a novel video object segmentation network. The network combines an object shape extraction module and an object motion prediction module, effectively solving the segmentation problem of multiple overlapping and confusing objects, and also improving the tracking ability of fast-moving objects. Our method shows excellent robustness to confusing objects in experiments, but the segmentation accuracy decreases in scenes with light changes. As shown in Figure 12, this is mainly because the change in lighting conditions causes a change in the object appearance, which affects the matching accuracy. We believe that future work should focus on the robustness of video object segmentation methods under different natural lighting conditions, and we suggest that more attempts can be made in the construction of datasets.

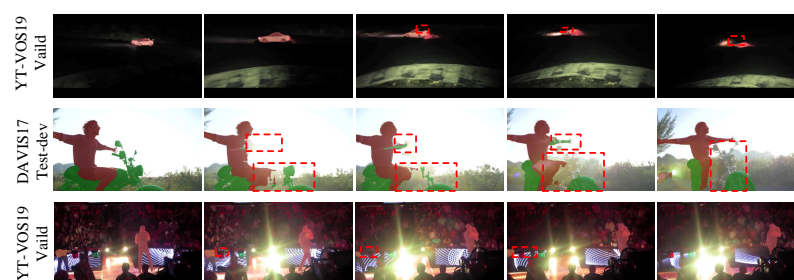


Figure 12. Limitation. We show three cases of failure due to natural factors. Our model fails to segment when the light changes. The red box indicates the areas where segmentation failed.

Author Contributions: Conceptualization, J.P., Y.Z. and D.Z.; methodology, J.P. and Y.Z.; software, J.P. and D.Z.; validation, J.P.; writing—original draft preparation, J.P.; writing—review and editing, J.P., Y.Z. and Y.C.; project administration, J.P. and Y.C.; funding acquisition, J.P. and Y.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded in part by the National Natural Science Foundation of China under grants 61802197 and 62072449.

Data Availability Statement: The data presented in this study are openly available in Refs. [13,14].

Acknowledgments: The authors thank the anonymous reviewers and the editors for their insightful comments and helpful suggestions for improving our manuscript.

Conflicts of Interest: The authors declare no conflicts of interest.

References

- Oh, S.W.; Lee, J.Y.; Xu, N.; Kim, S.J. Video object segmentation using space-time memory networks. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 9226–9235.
- Cheng, H.K.; Tai, Y.W.; Tang, C.K. Rethinking space-time networks with improved memory coverage for efficient video object segmentation. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 11781–11794.
- Yang, Z.; Wei, Y.; Yang, Y. Collaborative video object segmentation by foreground-background integration. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 332–348.
- Yang, Z.; Wei, Y.; Yang, Y. Collaborative video object segmentation by multi-scale foreground-background integration. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 4704–4712. [[CrossRef](#)] [[PubMed](#)]
- Li, M.; Hu, L.; Xiong, Z.; Zhang, B.; Pan, P.; Liu, D. Recurrent Dynamic Embedding for Video Object Segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 1332–1341.
- Li, Y.; Shen, Z.; Shan, Y. Fast video object segmentation using the global context module. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 735–750.
- 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, Virtual, 19–25 June 2021; pp. 4144–4154.
- 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, Virtual, 19–25 June 2021; pp. 1286–1295.

9. Seong, H.; Hyun, J.; Kim, E. Kernelized memory network for video object segmentation. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 629–645.
10. Chen, Y.; Zhang, D.; Yang, Z.X.; Wu, E. Robust and Efficient Memory Network for Video Object Segmentation. *arXiv* **2023**, arXiv:2304.11840.
11. Chen, Y.; Zhang, D.; Zheng, Y.; Yang, Z.X.; Wu, E.; Zhao, H. Boosting Video Object Segmentation via Robust and Efficient Memory Network. *IEEE Trans. Circuits Syst. Video Technol.* **2023**. [[CrossRef](#)]
12. Medsker, L.R.; Jain, L. Recurrent neural networks. *Des. Appl.* **2001**, *5*, 2.
13. Pont-Tuset, J.; Perazzi, F.; Caelles, S.; Arbeláez, P.; Sorkine-Hornung, A.; Van Gool, L. The 2017 davis challenge on video object segmentation. *arXiv* **2017**, arXiv:1704.00675.
14. Xu, N.; Yang, L.; Fan, Y.; Yue, D.; Liang, Y.; Yang, J.; Huang, T. Youtube-vos: A large-scale video object segmentation benchmark. *arXiv* **2018**, arXiv:1809.03327.
15. Voigtlaender, P.; Krause, M.; Osep, A.; Luiten, J.; Sekar, B.B.G.; Geiger, A.; Leibe, B. Mots: Multi-object tracking and segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 7942–7951.
16. Chen, X.; Li, Z.; Yuan, Y.; Yu, G.; Shen, J.; Qi, D. State-aware tracker for real-time video object segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 9384–9393.
17. Cheng, H.K.; Schwing, A.G. XMem: Long-Term Video Object Segmentation with an Atkinson-Shiffrin Memory Model. In Proceedings of the ECCV, Tel Aviv, Israel, 23–27 October 2022.
18. Seong, H.; Oh, S.W.; Lee, J.Y.; Lee, S.; Lee, S.; Kim, E. Hierarchical memory matching network for video object segmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 12889–12898.
19. Yang, Z.; Wei, Y.; Yang, Y. Associating objects with transformers for video object segmentation. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 2491–2502.
20. Liu, Q.; Wu, J.; Jiang, Y.; Bai, X.; Yuille, A.L.; Bai, S. InstMove: Instance Motion for Object-centric Video Segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 18–22 June 2023; pp. 6344–6354.
21. Chen, Y.; Hao, C.; Yang, Z.X.; Wu, E. Fast target-aware learning for few-shot video object segmentation. *Sci. China Inf. Sci.* **2022**, *65*, 182104. [[CrossRef](#)]
22. Ye, Q.; Huang, P.; Zhang, Z.; Zheng, Y.; Fu, L.; Yang, W. Multiview learning with robust double-sided twin SVM. *IEEE Trans. Cybern.* **2021**, *52*, 12745–12758. [[CrossRef](#)] [[PubMed](#)]
23. Fu, L.; Li, Z.; Ye, Q.; Yin, H.; Liu, Q.; Chen, X.; Fan, X.; Yang, W.; Yang, G. Learning Robust Discriminant Subspace Based on Joint L_2 , p - and L_2 , s -Norm Distance Metrics. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *33*, 130–144. [[CrossRef](#)] [[PubMed](#)]
24. Miles, R.; Yucel, M.K.; Manganelli, B.; Saà-Garriga, A. MobileVOS: Real-Time Video Object Segmentation Contrastive Learning meets Knowledge Distillation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 18–22 June 2023; pp. 10480–10490.
25. Takikawa, T.; Acuna, D.; Jampani, V.; Fidler, S. Gated-scnn: Gated shape cnns for semantic segmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 5229–5238.
26. Zhang, P.; Hu, L.; Zhang, B.; Pan, P.; Alibaba, D. Spatial consistent memory network for semi-supervised video object segmentation. In Proceedings of the CVPR Workshops, Seattle, WA, USA, 14–19 June 2020; Volume 6, p. 2.
27. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
28. Oh, S.W.; Lee, J.Y.; Sunkavalli, K.; Kim, S.J. Fast video object segmentation by reference-guided mask propagation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7376–7385.
29. Rempe, D.; Birdal, T.; Hertzmann, A.; Yang, J.; Sridhar, S.; Guibas, L.J. Humor: 3d human motion model for robust pose estimation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 11488–11499.
30. Lee, S.; Kim, H.G.; Choi, D.H.; Kim, H.I.; Ro, Y.M. Video prediction recalling long-term motion context via memory alignment learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 3054–3063.
31. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.
32. 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, Las Vegas, NV, USA, 27–30 June 2016; pp. 724–732.
33. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. Pytorch: An imperative style, high-performance deep learning library. *Adv. Neural Inf. Process. Syst.* **2019**, *32*.
34. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.

35. 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, Honolulu, HI, USA, 21–26 July 2017; pp. 2663–2672.
36. Cho, S.; Lee, H.; Lee, M.; Park, C.; Jang, S.; Kim, M.; Lee, S. Tackling background distraction in video object segmentation. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 446–462.
37. Cheng, H.K.; Tai, Y.W.; Tang, C.K. Modular interactive video object segmentation: Interaction-to-mask, propagation and difference-aware fusion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Virtual, 19–25 June 2021; pp. 5559–5568.
38. Yang, L.; Wang, Y.; Xiong, X.; Yang, J.; Katsaggelos, A.K. Efficient video object segmentation via network modulation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6499–6507.
39. 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, Long Beach, CA, USA, 15–20 June 2019; pp. 5277–5286.
40. Luiten, J.; Voigtlaender, P.; Leibe, B. Premvos: Proposal-generation, refinement and merging for video object segmentation. In Proceedings of the Asian Conference on Computer Vision, Perth, Australia, 2–6 December 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 565–580.
41. Zhang, Y.; Wu, Z.; Peng, H.; Lin, S. A transductive approach for video object segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 6949–6958.
42. Wang, H.; Jiang, X.; Ren, H.; Hu, Y.; Bai, S. Swiftnet: Real-time video object segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Virtual, 19–25 June 2021; pp. 1296–1305.
43. Lu, X.; Wang, W.; Danelljan, M.; Zhou, T.; Shen, J.; Gool, L.V. Video object segmentation with episodic graph memory networks. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 661–679.
44. Bhat, G.; Lawin, F.J.; Danelljan, M.; Robinson, A.; Felsberg, M.; Gool, L.V.; Timofte, R. Learning what to learn for video object segmentation. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 777–794.
45. Chen, Y.; Ji, C.; Yang, Z.X.; Wu, E. Spatial constraint for efficient semi-supervised video object segmentation. *Comput. Vis. Image Underst.* **2023**, *237*, 103843. [[CrossRef](#)]
46. Duke, B.; Ahmed, A.; Wolf, C.; Aarabi, P.; Taylor, G.W. Sstvos: Sparse spatiotemporal transformers for video object segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Virtual, 19–25 June 2021; pp. 5912–5921.
47. Voigtlaender, P.; Leibe, B. Online adaptation of convolutional neural networks for video object segmentation. *arXiv* **2017**, arXiv:1706.09364.
48. Voigtlaender, P.; Chai, Y.; Schrott, F.; Adam, H.; Leibe, B.; Chen, L.C. Feelvos: Fast end-to-end embedding learning for video object segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 9481–9490.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.