Recurrent Dynamic Embedding for Video Object Segmentation Supplementary Material

Mingxing Li[1], Li Hu[2], Zhiwei Xiong[1], Bang Zhang[2], Pan Pan[2], Dong Liu[1]
[1]University of Science and Technology of China
[2]Alibaba DAMO Academy, Alibaba Group

1. Details of Memory Bank Update with EMA

We summarize the EMA based methods first select proposal embeddings and then merge the proposal embeddings with the old memory bank in an EMA way. As shown in Eq. 1 (note Eq. 1 is the process to pixel level), GCNet [5] uses simple averaging ($\lambda = 0.5$ in Eq. 1). AFB-URR [6] uses EMA ($\lambda = 0 \sim 1$ in Eq. 1) when the new feature is close to an existing one. And SwiftNet [9] uses the most similar embedding implemented by *argmax* ($\lambda = 0$ in Eq. 1) when the pixel location is triggered.

$$\mathbf{k}_t^{IE}(p) = (1-\lambda)\mathbf{k}^{Q}(q) + \lambda \mathbf{k}_{t-\theta}^{IE}(p) \tag{1}$$

2. Ablation Study

2.1. Weights of Loss Function

In our experiments, the overall loss function, which is computed as follows:

$$Loss = L_{Seg} + \mathbb{1}[t = 3, 5]\mu L_{UG} + \gamma L_{MC}. \tag{2}$$

The default setting of the weights is $\mu = \gamma = 10$, we fix one item and change the other item to show the ablation of the weights in Table 1.

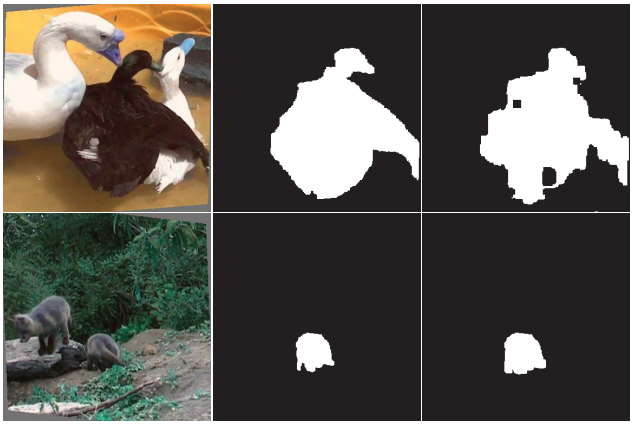

Figure 1. An illustration of perturbations for the GT frame. From left to right are the input image, GT, and perturbated GT.

Weight	0	5	10	15
L_{UG}	82.9	83.8	**84.2**	83.6
L_{MC}	83.5	83.7	**84.2**	83.9

Table 1. Ablation of weights of the loss function on the DAVIS 2017 validation set. We evaluate $\mathcal{J}\&\mathcal{F}$ for the different weights of the unbiased guidance loss L_{UG} and the mask consistency loss L_{MC}, when the weight of the other is fixed ($L_{UG} = L_{MC} = 10$ by default).

2.2. Perturbation Levels

As shown in Figure 1, we perform perturbation transform such as the random dilation and eroding on the first frame with different perturbation levels. We adopt intersection over union (IOU) between the perturbated mask and the GT mask to indicate the perturbation levels. In our experiments, our method generates a perturbated mask with the perturbation level randomly sampled from an interval [*lower bound*, 1]. We fix the weight of the mask consistency loss L_{MC} to 10 and show ablation of *lower bound* in Table 2. Too high perturbation level (low IOU) makes the training of the network more difficult, which is not conducive to network convergence.

lower bound	0.55	0.65	0.75	0.85	0.95
$\mathcal{J}\&\mathcal{F}$	82.2	82.7	83.6	**84.2**	83.2

Table 2. Ablation of the perturbation levels. We fix the weight of the mask consistency loss L_{MC} to 10.

2.3. Sampling Interval

On the validation set of DAVIS 2017 and YouTube-VOS 2019, Figure 2 shows ablation of sampling interval θ. We fix the sampling interval $\theta = 3$ on the DAVIS datasets and achieve the new state-of-the-art performance. And we set the sampling interval $\theta = 4$ on YouTube-VOS 2019 to fit the motion pattern on YouTube-VOS. We will extend the adaptive mechanism of the update interval for SAM in future work. As the sampling interval increases, the increase rate of FPS on DAVIS 2017 becomes slower. The analysis

of this phenomenon can be found in Sec. 2.4.

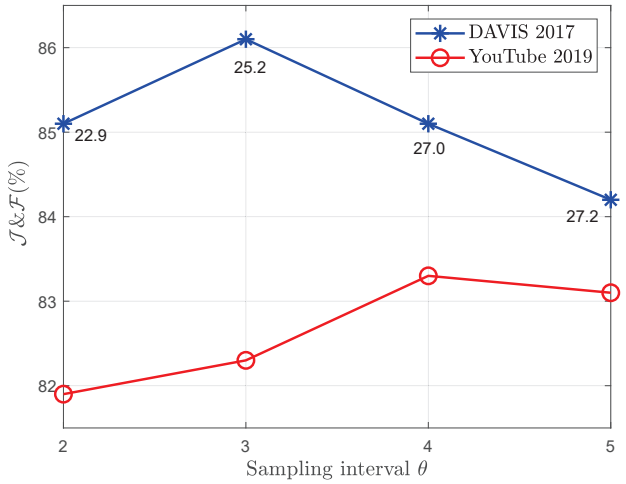

Figure 2. Ablation of sampling interval θ on the validation set of DAVIS 2017 and YouTube-VOS 2019. On DAVIS 2017, we additionally annotate FPS of different sampling intervals.

2.4. Analysis of Inference Time

We find as the sampling interval increases, the increase rate of FPS on DAVIS 2017 becomes slower. As shown in Figure 3, we explore the average inference time of different components of our method on the DAVIS 2017 validation set. The main reason is the most time-consuming part is not SAM but the matching operation and decoder.

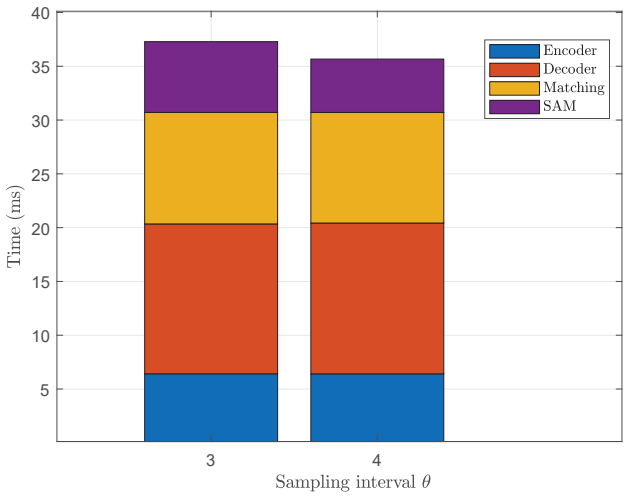

Figure 3. Ablation of average inference time of different components on the DAVIS 2017 validation set.

2.5. Analysis of RDE

We have demonstrated the superiority of our recurrent dynamic embedding (RDE) in experiments. While how to analyze the quality of embedding is not intuitive. Inspired by [10], we employ PCA [8] to project the value embedding of the STM pattern memory bank and RDE into RGB space for visualization. For the DIY car in "soapbox", as the length of the video increases, the embedding of the STM pattern memory bank gradually blurs. It is because a large amount of information is introduced losslessly, which may be not conducive to reading the most important information for the network. Our RDE originates the embedding of the latest frame and the historical information (previous RDE). It is based on a weak Markovian assumption, which means except for the previous RDE and the latest frame, the segmentation of the query frame is independent of past states.

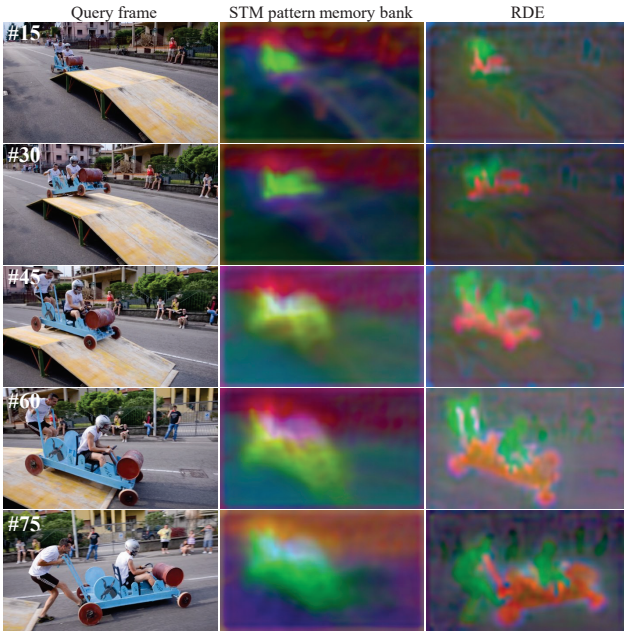

Figure 4. The value embedding of the DIY car processed by PCA [8]. $\#n$ denotes the n-th frame. In the STM [7] pattern memory bank of STCN [3], as the video length increases, the new embedding of the DIY car is continuously concatenated into the STM pattern memory bank, which inevitably introduces lots of noise. Our recurrent dynamic embedding (RDE) originates the embedding of the latest frame and the historical information (previous RDE) to support the most helpful information to the segmentation of the query frame.

2.6. Enhancing Part of SAM

We show the ablation of the enhancing part of SAM without the BL30K [2] pre-training. We utilize a simple atrous spatial pyramid pooling (ASPP) [1] in the enhancing part. Without atrous spatial pyramid pooling (ASPP) [1], $\mathcal{J}\&\mathcal{F}$ drops 1.3% which verifies the importance of the enhancing part. We further evaluate to enrich the representation of the enhancing part by introducing several residual blocks (ResBlock) [4]. We find adding more residual blocks

	Ablation Settings	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$
Architecture	w/o enhancing part	82.9	79.8	86.0
	w/ ASPP	**84.2**	**80.8**	**87.5**
	+ResBlock × 1	83.5	80.4	86.6
	+ResBlock × 2	83.2	79.8	86.6
	+ResBlock × 3	82.6	79.6	85.6

Table 3. Ablation of the enchaining part of SAM without the BL30K [2] pre-training.

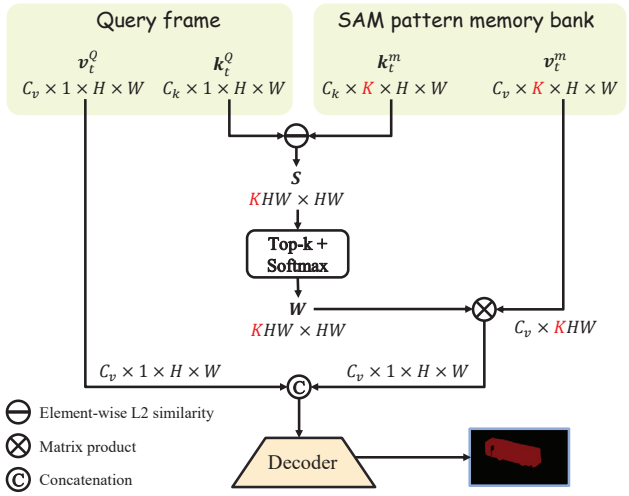

Figure 5. An illustration of the memory reading from our SAM pattern memory bank. Constant K is 2 in our experiments owing to the concatenation of previous RDE and the embedding of the latest frame.

cannot improve the performance but drop. It indicates SAM does not require a deep network structure.

3. Memory Reading Details

For the SAM pattern memory bank m at time t, we keep target-agnostic key $\mathbf{k}_t^m \in \mathbb{R}^{N \times C_k \times H \times W}$ and target-specific value $\mathbf{v}_t^m \in \mathbb{R}^{N \times O \times C_v \times H \times W}$, where N denotes the size of a batch, O denotes the total number of the objects and $H \times W$ denotes the spatial size of the embeddings. First, we flatten $N \times O$ to NO for the value $\mathbf{v}_t^m$ to the efficient calculation. As shown in Figure 5, we ignore the batch axis for brevity. Given the query frame t, we read the information from the SAM pattern memory bank as shown in Figure 5.

4. Inference Details

During the inference, the pseudo-code of our method is in Algorithm 1. Every several frames (e.g., 3), RDE is updated by SAM and old RDE is discarded. The whole process is concise and extensible.

Algorithm 1 Inference on DAVIS, PyTorch-like

```
# frame size: H x W
# object number: C (including background)
# frame_range: the index list of frames except for
    the first frame (N-1)
# prob: output tensor (C, N, 1, H, W)
# mem_every: update interval

# Obtain key, value of the first frame
first_k, first_v = ImageE(0)

# SAM initialization with the embeddings of the
    first frame.
# 1. mem_gt for two repeated gt frame
# 2. mem_temp for the latest frame
# 3. mem_rde for SAM
mem_bank.initialize()

for i in frame_range:
   # Extract key (qk16), value (qv16) and middle
       features (qf16, qf8, qf4) for frame i
   qk16, qv16, qf16, qf8, qf4 = ImageE(i)

   # Segment the i-th frame with the SAM pattern
       memory bank
   out_mask = segment_with_query(mem_bank, qf8, qf4,
       qk16, qv16)

   # Aggregate like STM
   prob[:,i] = aggregate(out_mask)

   if (i % mem_every) == 0:
         key_i = qk16

         # Encode query frame i
         value_i = MaskE(i, qf16, out_mask[1:])

         # Concatenate previous RDE and the
             embedding of the latest frame
         mem_cross = cat(mem_rde, mem_temp)

         # Recurrently use SAM
         mem_cross = SAM(mem_cross)

         # Update the SAM pattern memory bank
         mem_rde.update(mem_cross)
         # Update the temporary key and value
         mem_temp.update([key_i, value_i])
```

5. Visualization Results

As shown in Figure 6 and 7, we show the qualitative results on DAVIS 2017 test set compared with STCN [3]. For complex scenes and similar-looking instances, our method has relatively fewer errors than STCN. Figure 8 and 9 show qualitative results on YouTube-VOS 2019 validation set, which also indicates the superiority of our method.

References

[1] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 2

[2] Ho Kei Cheng, Yu-Wing Tai, and Chi-Keung Tang. Modular interactive video object segmentation: Interaction-to-mask, propagation and difference-aware fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5559–5568, 2021. 2, 3

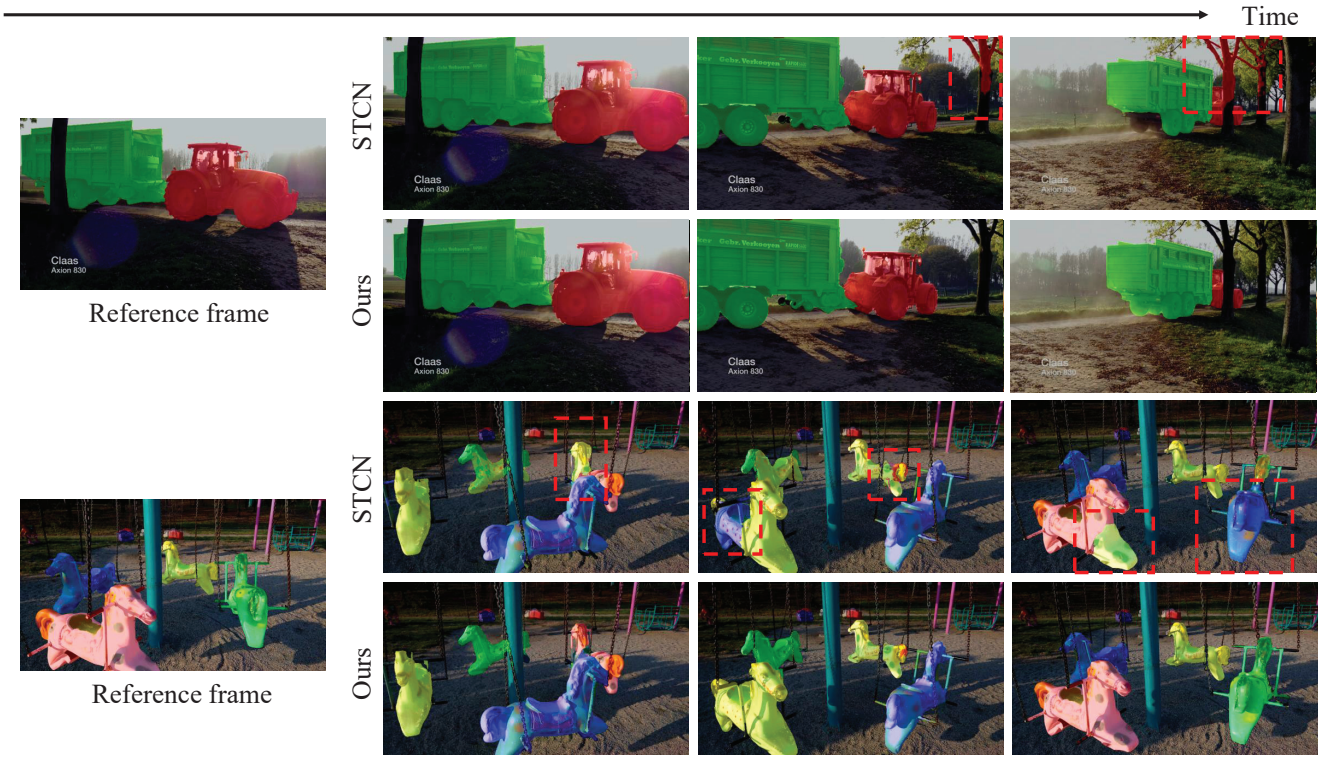

Figure 6. Qualitative results of STCN [3] and our method on DAVIS 2017 test set. We mark errors with red dotted boxes for the best view.

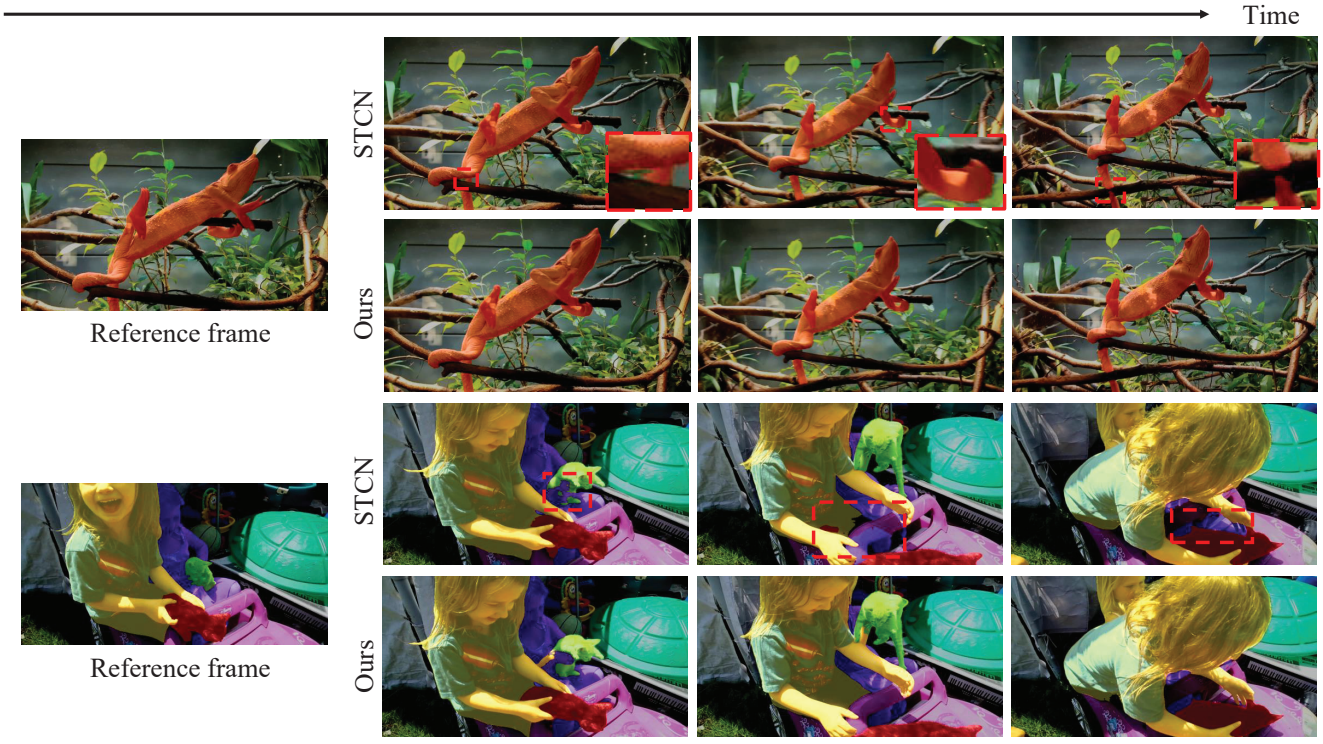

Figure 7. Qualitative results of STCN [3] and our method on DAVIS 2017 test set. We mark errors with red dotted boxes for the best view.

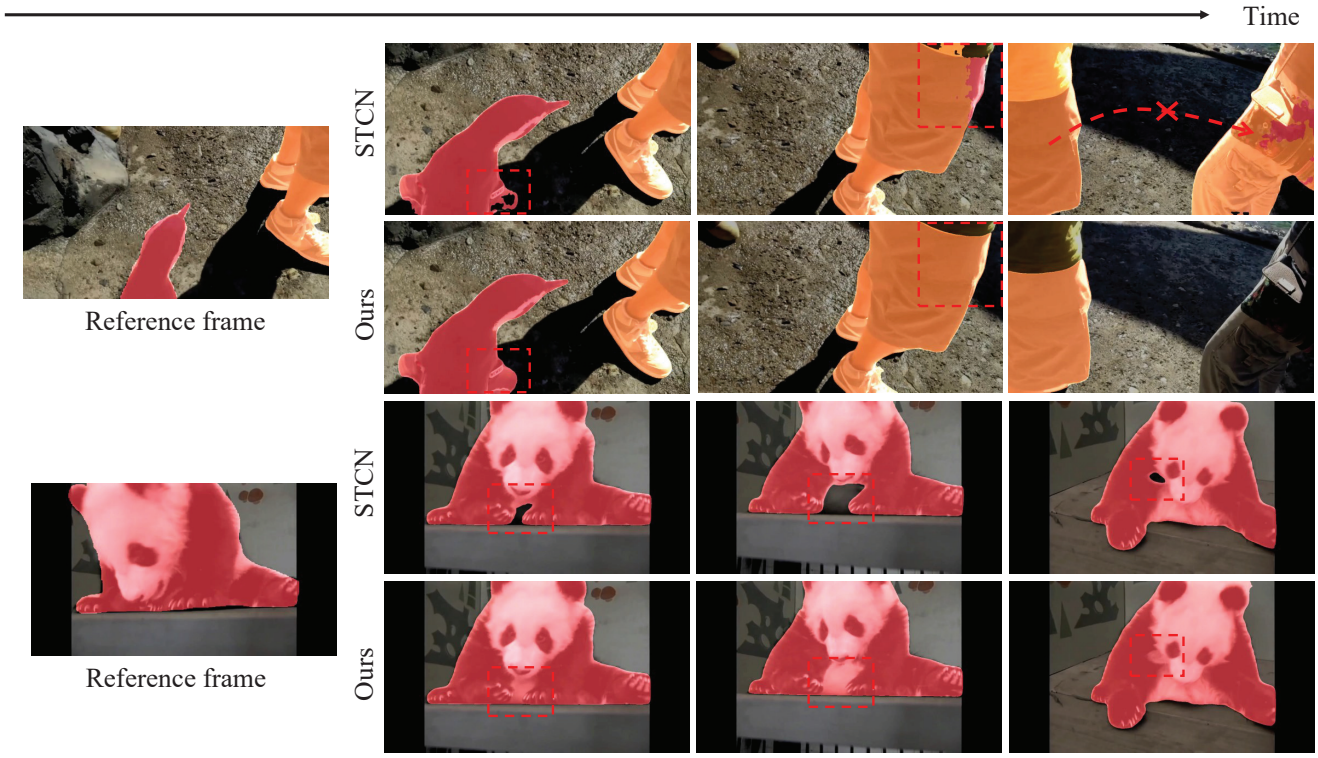

Figure 8. Qualitative results of STCN [3] and our method on YouTube-VOS 2019. We mark errors with red dotted boxes for the best view.

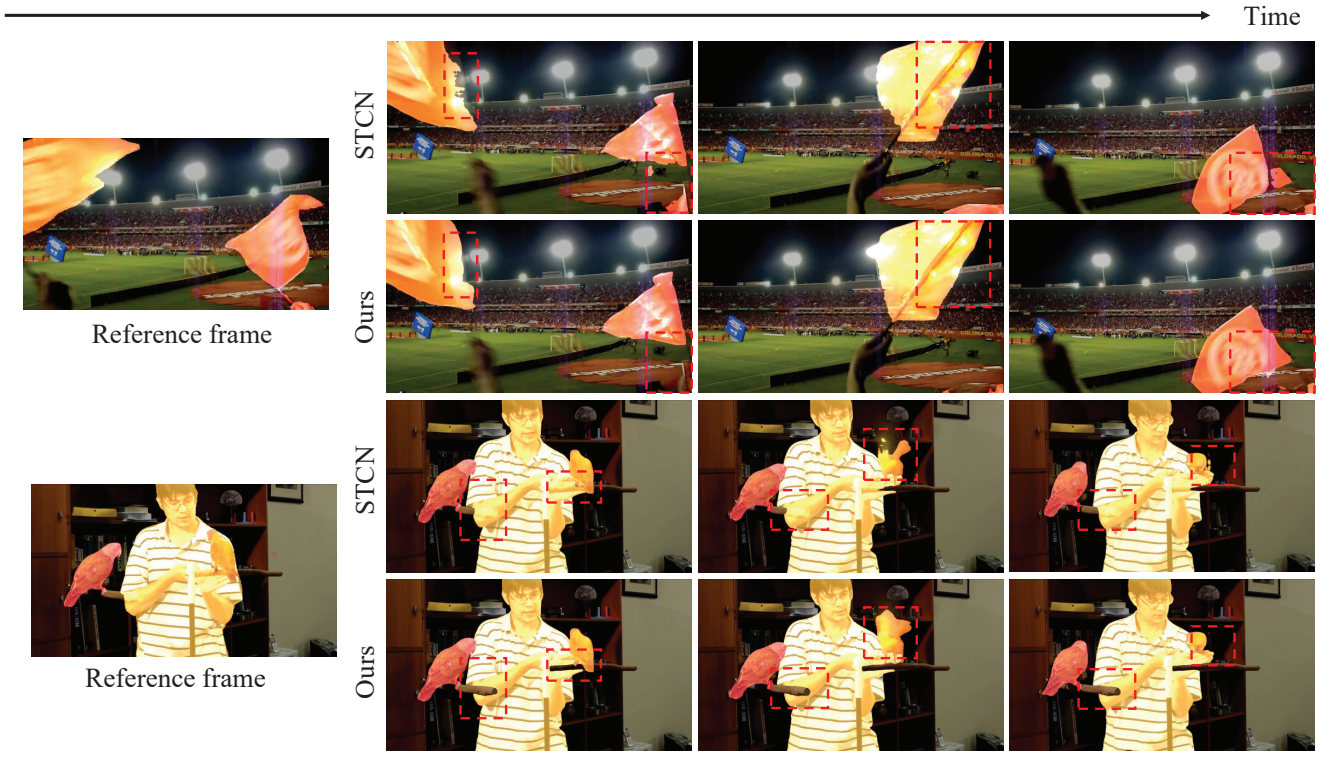

Figure 9. Qualitative results of STCN [3] and our method on YouTube-VOS 2019. We mark errors with red dotted boxes for the best view.

[3] Ho Kei Cheng, Yu-Wing Tai, and Chi-Keung Tang. Rethinking space-time networks with improved memory coverage for efficient video object segmentation. *arXiv preprint arXiv:2106.05210*, 2021. 2, 3, 4, 5

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[5] Yu Li, Zhuoran Shen, and Ying Shan. Fast video object segmentation using the global context module. In *European Conference on Computer Vision*, pages 735–750. Springer, 2020. 1

[6] Yongqing Liang, Xin Li, Navid Jafari, and Qin Chen. Video object segmentation with adaptive feature bank and uncertain-region refinement. *arXiv preprint arXiv:2010.07958*, 2020. 1

[7] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9226–9235, 2019. 2

[8] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572, 1901. 2

[9] Haochen Wang, Xiaolong Jiang, Haibing Ren, Yao Hu, and Song Bai. Swiftnet: Real-time video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1296–1305, 2021. 1

[10] Adrian Wolny, Qin Yu, Constantin Pape, and Anna Kreshuk. Sparse object-level supervision for instance segmentation with pixel embeddings. *arXiv preprint arXiv:2103.14572*, 2021. 2