

SUPPLEMENTARY MATERIAL FOR MOTION-INDUCTIVE SELF-SUPERVISED OBJECT DISCOVERY IN VIDEOS

Anonymous authors

Paper under double-blind review

A MORE IMPLEMENTATION DETAILS

In this section, we list the architecture details and training settings. Codes and models will be released publicly.

A.1 VISUAL ENCODER

The visual encoder contains the first three stages of Swin-Tiny V2. We tabulate the workflow in Table 1.

stage	operation	output sizes
input	-	$3 \times 192 \times 384$
PatchEmbed	4×4 , stride 4, 96	$96 \times 48 \times 96$
SwinBlock1	$\begin{bmatrix} h = 3 \\ ws = 12 \end{bmatrix} \times 2$	$96 \times 48 \times 96$
DownSample1	192	$192 \times 24 \times 48$
SwinBlock2	$\begin{bmatrix} h = 6 \\ ws = 12 \end{bmatrix} \times 2$	$192 \times 24 \times 48$
DownSample2	384	$384 \times 12 \times 24$
SwinBlock3	$\begin{bmatrix} h = 12 \\ ws = 12 \end{bmatrix} \times 6$	$384 \times 12 \times 24$

Table 1: Architecture of visual encoder. h stands for the number of attention heads while ws refers to window size.

A.2 FRAME COMPARATOR

The frame comparator possesses two deformable convolutional layers, with ReLU operation in between and three Transformer Encoder blocks. We tabulate the workflow in Table 2.

stage	operation	output sizes
input	-	$768 \times 12 \times 24$
DeformConv1	3×3 , 768	$768 \times 12 \times 24$
ReLU	-	$768 \times 12 \times 24$
DeformConv2	3×3 , 384	$384 \times 12 \times 24$
Transformer	$\begin{bmatrix} h = 8 \\ 384 \end{bmatrix} \times 3$	$384 \times 12 \times 24$

Table 2: Architecture of frame comparator. h represents the number of attention heads

A.3 FLOW DECODER

The frame comparator consists of three stages of SwinV2 block added with linear expanding layers. We tabulate the workflow in Table 3.

stage	operation	output sizes
input	-	$384 \times 12 \times 24$
SwinBlock1	$\begin{bmatrix} h = 12 \\ ws = 12 \end{bmatrix} \times 2$	$384 \times 12 \times 24$
PatchExpand1	768	$192 \times 24 \times 48$
SwinBlock2	$\begin{bmatrix} h = 6 \\ ws = 12 \end{bmatrix} \times 2$	$192 \times 24 \times 48$
PatchExpand2	384	$96 \times 48 \times 96$
SwinBlock3	$\begin{bmatrix} h = 3 \\ ws = 12 \end{bmatrix} \times 2$	$96 \times 48 \times 96$
PatchExpand3	1536	$96 \times 192 \times 384$
outConv	5×5 , stride 1, 4	$4 \times 192 \times 384$

Table 3: Architecture of flow decoder. h stands for the number of attention heads while ws refers to window size.

A.4 TRAINING DETAILS

For all datasets, we train with a batch size of 2. To train more efficiently, we sample three flow pairs ($i \rightarrow j$) for a given frame i ; one is static replication $i = j$, another two are motion pair $i \neq j$. We apply temporal consistency first on the masks from two motion pairs, then pull the static mask to the average of two dynamic masks closer. We linearly warm up the learning rate for the first 1k iterations. Besides, for every 100k iterations, we decay the learning rate by half and increase the scale of temporal consistency λ_c and entropy regularisation λ_e by the factor of 5. In the default setting, we train for about 3 days on 8 standard Tesla V100 GPUs with 32GB memory each.

A.5 TEST-TIME ADAPTATION

Inspired by OCLR (Xie et al., 2022), we adopt test-time adaptation based on RGB sequence to enhance appearance consistency. In detail, we follow existing works on self-supervised tracking (Lai et al., 2020; Jabri et al., 2020) to propagate object masks across time span. The whole process consists of three steps. First, we extract RGB features of each frame with a DINO-pretrained ViT encoder. Then, we select key frames for object mask propagation. Finally, we calculate the affinity matrix between frames and perform mask propagation.

DINO Feature Extraction. Given a video sequence $v = \{x_1, \dots, x_T\}$, $v \in \mathbb{R}^{T \times H \times W \times 3}$, we use DINO pretrained ViT-Small encoder with patch size 8×8 to extract features:

$$\{f_1, \dots, f_T\} = \{\Phi(x_1), \dots, \Phi(x_T)\}, \quad f_t \in \mathbb{R}^{h \times w \times 384}, \quad (1)$$

where $h = H/8$ and $w = W/8$. The extracted features will be used in the mask propagation step.

Key Frame Selection. Given video v , our model predicts object mask of each frame as $m = \{\alpha_1, \dots, \alpha_T\}$, $m \in \mathbb{R}^{T \times H \times W \times 1}$. Since the video frames are continuous along the temporal dimension, it is practical to propagate object masks between neighboring frames. The propagation operation is the same as Jabri et al. (2020), the only difference is that we have no ground-truth mask for reference. Therefore, we need to design a mechanism to select object masks of high confidence. To do this, we measure the temporal coherence of predicted object masks for key frame selection. Specifically, for each timestamp $t \in \{3, \dots, T-2\}$, we can calculate four propagated masks as:

$$\hat{\alpha}_t = [\text{Mask-prop}(\alpha_{t-2}), \text{Mask-prop}(\alpha_{t-1}), \text{Mask-prop}(\alpha_{t+1}), \text{Mask-prop}(\alpha_{t+2})], \quad (2)$$

where ‘Mask-prop’ denotes the propagation operation. Then we calculate the average IoU between the original mask and propagated masks as the confidence score:

$$s_t = \frac{\text{IoU}(\hat{\alpha}_t[0], \alpha_t) + \text{IoU}(\hat{\alpha}_t[1], \alpha_t) + \text{IoU}(\hat{\alpha}_t[2], \alpha_t) + \text{IoU}(\hat{\alpha}_t[3], \alpha_t)}{4}. \quad (3)$$

Sequence	\mathcal{J} (Mean) \uparrow	
	w/o post proc.	test-time adap.
dog	80.7	87.4
cows	87.2	88.8
goat	47.5	80.6
camel	85.6	86.1
libby	72.5	77.7
parkour	72.9	87.9
soapbox	84.6	86.5
blackswan	48.9	46.9
bmx-trees	50.1	55.8
kite-surf	55.9	62.6
car-shadow	87.9	86.9
breakdance	82.6	76.0
dance-twirl	82.5	85.4
scooter-black	80.2	80.3
drift-chicane	78.6	82.2
motocross-jump	68.4	88.9
horsejump-high	78.0	84.3
drift-straight	69.2	80.0
car-roundabout	87.7	83.9
paragliding-launch	62.1	62.8
frame avg.	73.9	79.2

Table 4: Sequence-wise results on DAVIS2016.

The calculated s_t measures the coherency between α_t and its neighbors. Our empirical studies show that it serves as a reliable signal for key frame selection.

Object Mask Propagation. We select timestamps with Top- $k\%$ confidence score as the key reference frames ($k = 15$ on DAVIS2016, $k = 25$ on SegTrackv2, $k = 10$ on FBMS-59). Then we iteratively propagate the object masks with a neighbor temporal window size $n = 7$.

B RESULTS BREAKDOWN

We include a specific result breakdown in this section. We show the per-sequence results on DAVIS2016 in Table 4, SegTrackv2 in Table 5 and FBMS-59 in Table 6.

C MORE QUALITATIVE RESULTS

We show more qualitative results of SegTrackv2 and FBMS-59 in Figure 1 and Figure 2.

Sequence	\mathcal{J} (Mean) \uparrow	
	w/o post proc.	test-time adap.
drift	41.7	40.7
birdfall	38.2	61.5
girl	76.5	82.3
cheetah	18.4	30.1
worm	52.5	74.3
parachute	90.2	92.0
monkeydog	14.3	31.1
hummingbird	61.2	58.8
soldier	66.3	58.8
bmx	73.7	78.8
frog	80.5	76.3
penguin	63.5	62.7
monkey	46.8	77.4
bird of paradise	85.3	85.4
frame avg.	62.2	69.4

Table 5: Sequence-wise results on SegTrackv2.

Sequence	\mathcal{J} (Mean) \uparrow	
	w/o post proc.	test-time adap.
camel01	25.8	66.9
cars1	66.1	88.3
cars10	31.6	33.9
cars4	72.9	83.2
cars5	81.2	82.5
cats01	80.6	79.2
cats03	62.0	63.4
cats06	40.1	38.7
dogs01	70.6	61.2
dogs02	62.8	82.2
farm01	86.7	88.9
giraffes01	38.6	52.2
goats01	44.8	45.3
horses02	64.4	77.6
horses04	68.5	73.5
horses05	43.7	49.0
lion01	60.1	71.5
marple12	74.7	80.3
marple2	65.0	71.4
marple4	79.1	91.6
marple6	76.0	85.1
marple7	72.5	55.2
marple9	87.5	97.9
people03	76.5	48.5
people1	72.4	80.8
people2	80.9	83.0
rabbits02	50.2	58.9
rabbits03	39.5	55.7
rabbits04	47.0	53.0
tennis	63.1	71.1
frame avg.	61.3	66.9

Table 6: Sequence-wise results on FBMS-59.

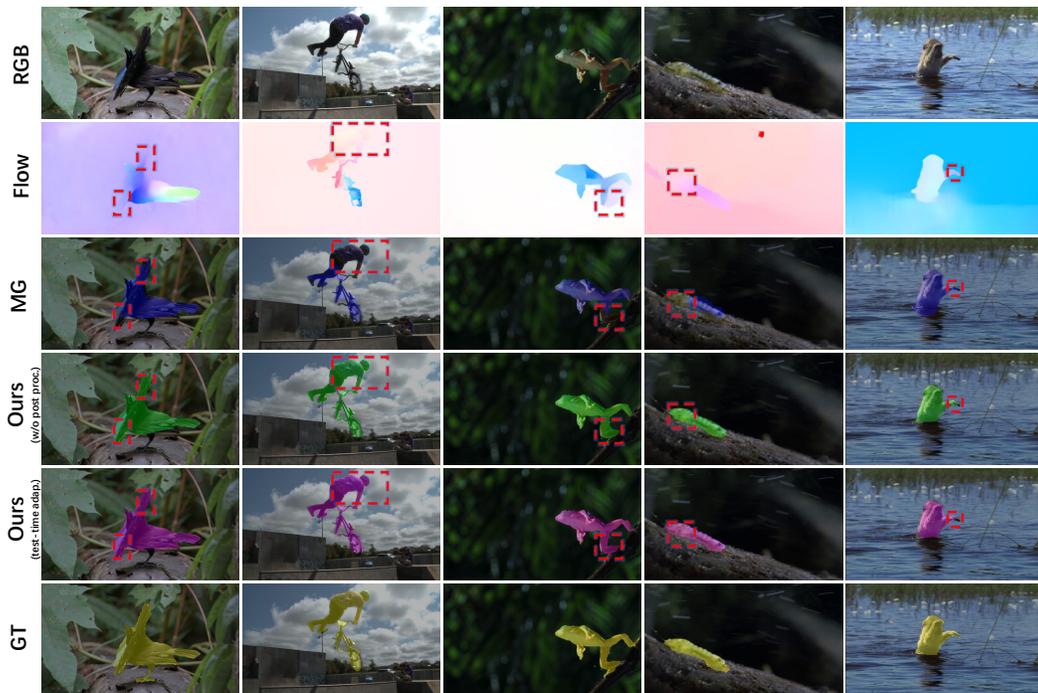


Figure 1: Qualitative results on SegTrackv2. MG refers to Yang et al. (2021). Red boxes outline the corresponding difference.

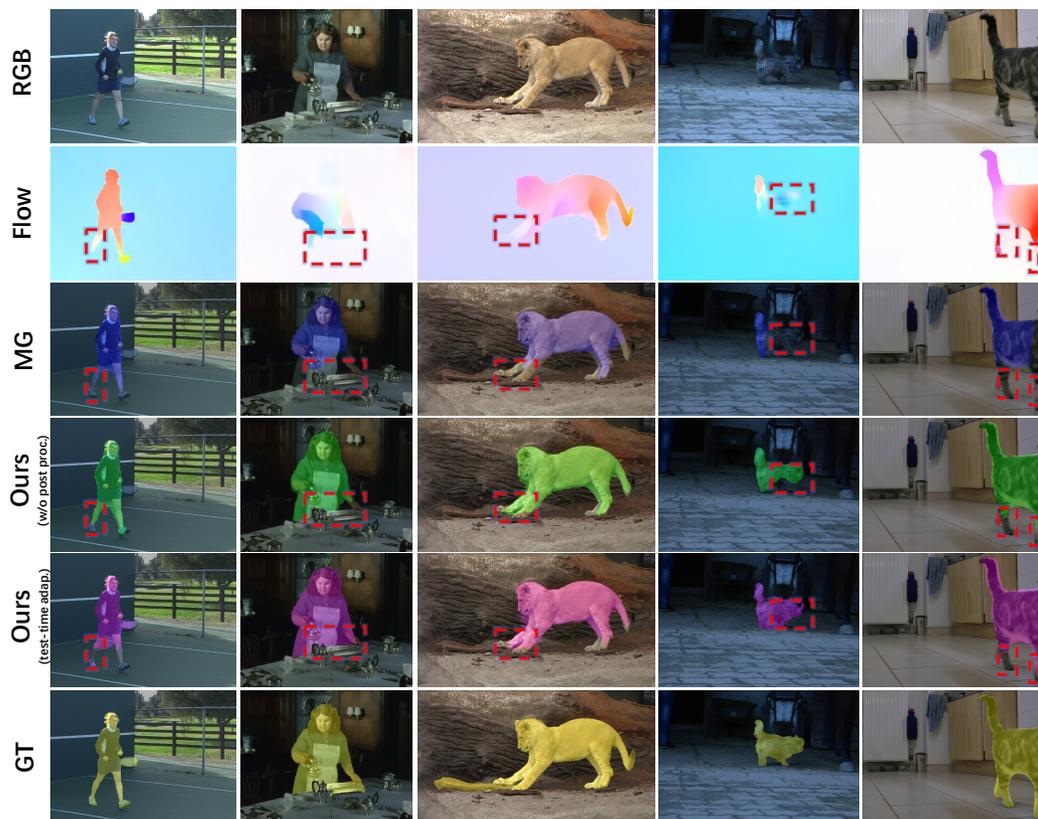


Figure 2: Qualitative results on FBMS-59. MG refers to Yang et al. (2021). Red boxes highlight the corresponding difference.

REFERENCES

- Allan Jabri, Andrew Owens, and Alexei Efros. Space-time correspondence as a contrastive random walk. *Advances in neural information processing systems*, 33:19545–19560, 2020.
- Zihang Lai, Erika Lu, and Weidi Xie. Mast: A memory-augmented self-supervised tracker. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6479–6488, 2020.
- Junyu Xie, Weidi Xie, and Andrew Zisserman. Segmenting moving objects via an object-centric layered representation. *arXiv preprint arXiv:2207.02206*, 2022.
- Charig Yang, Hala Lamdouar, Erika Lu, Andrew Zisserman, and Weidi Xie. Self-supervised video object segmentation by motion grouping. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7177–7188, October 2021.