Supplementary Material for Point-VOS: Pointing Up Video Object Segmentation

Abstract

In this supplementary, we provide the experimental results of the additional simulations in Sec. 1, the details for annotating the Point-VOS Oops validation set in Sec. 2, the statistics for Point-VOS datasets in Sec. 3, the implementation details in Sec. 4 and the additional qualitative results in Sec. 5.

1. Additional Simulations

Farthest Point Sampling Strategy. In Sec. 3.1 of the main paper, we ran a number of point simulation experiments on DAVIS [13] and YT-VOS [22] to analyse the effect of using point annotations both during training and testing. For these experiments, the simulated points are sampled randomly from the available ground truth segmentation masks for each frame.

In addition to sampling the points randomly, we also consider using the farthest-point sampling (FPS) technique. The FPS algorithm starts from some random initial point in the given input point set, and then iteratively selects a single point that has the largest distance from all the previously sampled ones. For our point simulations, instead of starting from a random point, we always start from a point that best represents the center of the input mask. To sample this center, we first generate Euclidean distance transforms from the ground-truth segmentation masks for each foreground object and the common background. We then sample the point that has the largest distance from each of these distance transforms and further use these as the starting points for the FPS algorithm. The FPS algorithm is then separately applied on the points that represent each of the foreground objects and the background starting from the corresponding center point.

Similar to the point simulation experiments presented in Sec. 3.1, we again use Point-STCN to train multiple models on different number of simulated points. Here again, we do not apply any temporal sparsity. Also, note that in both random and FPS point sampling strategies, we run each experiment 5 times and report the mean score. In Fig. 1, we show the results for the FPS point sampling strategy on the DAVIS validation set. It can be seen that the performance

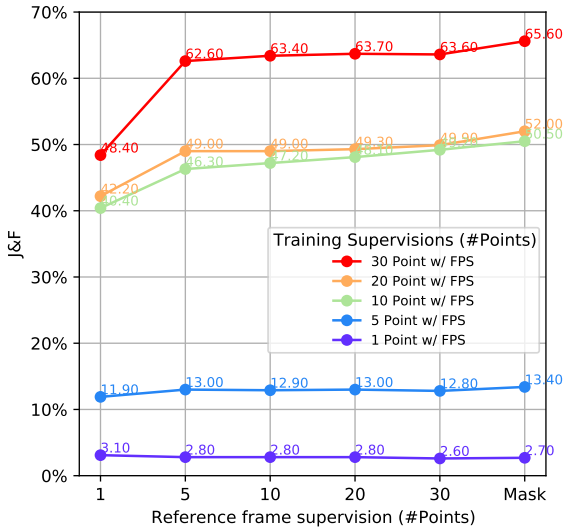


Figure 1. **FPS point sampling results on the DAVIS validation set.** We vary the number of sampled points per object for training supervision and the number of sampled points on the reference frame.

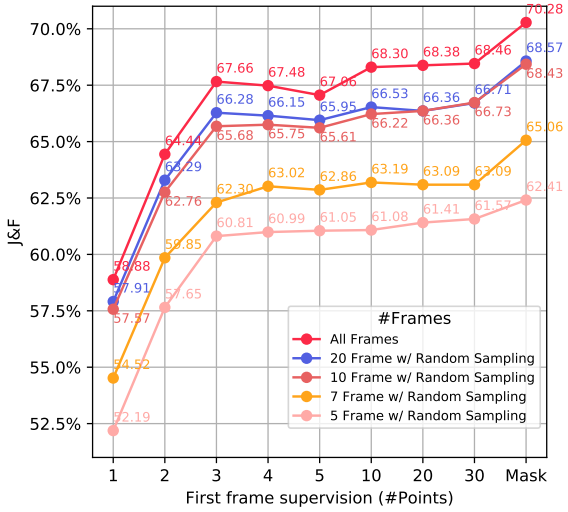


Figure 2. **Results on the DAVIS validation set for randomly sub-sampling frames.** We vary the number of randomly sampled frames.

of Point-STCN is much worse when we use FPS instead of the random sampling strategy (see Fig. 2 in the main paper), *e.g.*, for FPS we achieve the best result by training

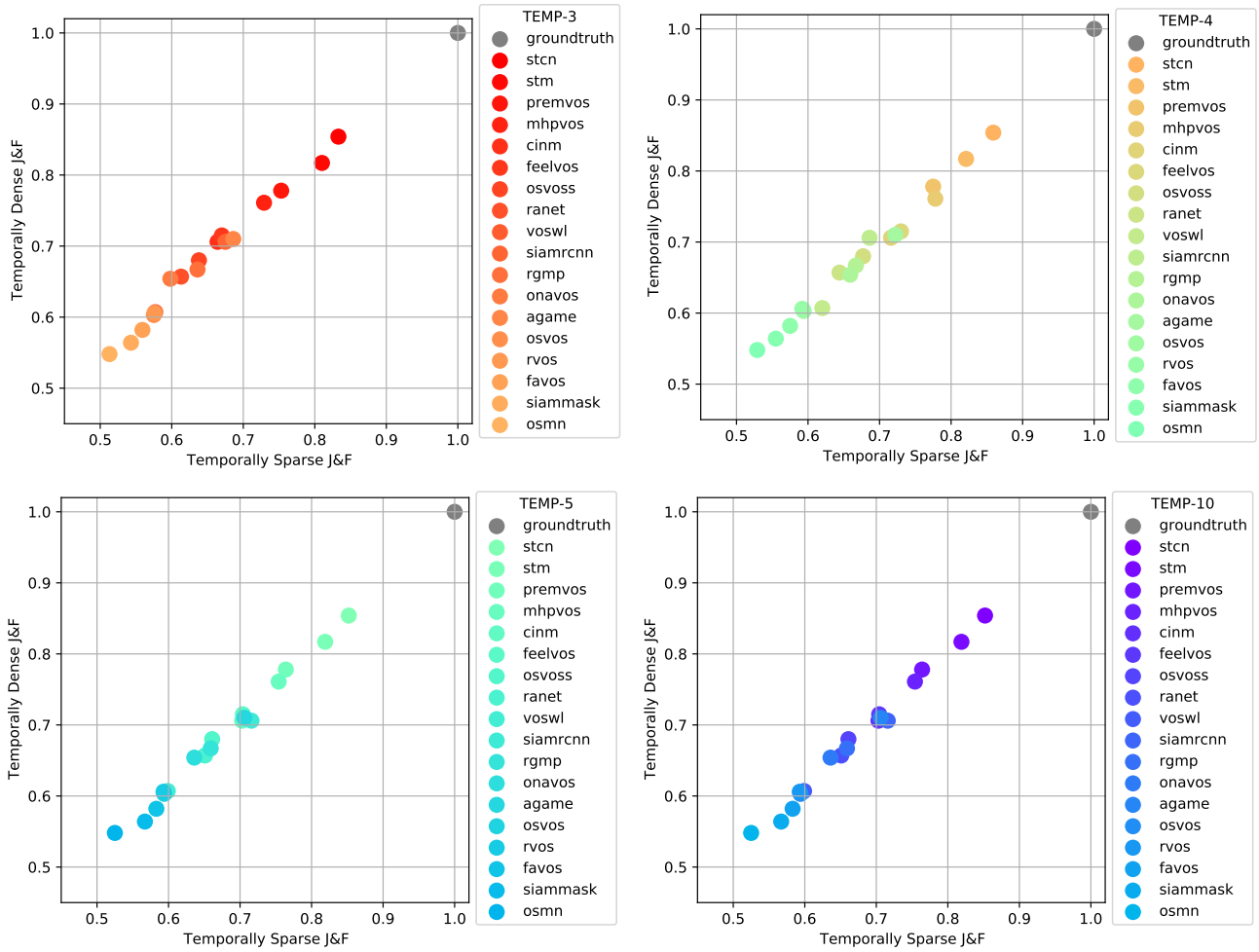


Figure 3. **Temporally dense $\mathcal{J}\&\mathcal{F}$ *vs*. temporally sparse $\mathcal{J}\&\mathcal{F}$ results.** We get 18 methods (colored dots) from the leaderboard of the DAVIS benchmark and evaluate them on the 4 different temporally sparse DAVIS validation sets. TEMP-3 shows the results evaluated on 3 sub-sampled frames, TEMP-4 on 4 sub-sampled frames, TEMP-5 on 5 sub-sampled frames, and TEMP-10 on 10 sub-sampled frames.

with 30 points, which is almost on par with using 10 points as training supervision in the random point sampling strategy. Thus, we decided to use the random point sampling strategy for our point annotations.

Randomly Sub-sampling Frames. Along with the evenly-spaced sub-sampling strategy explained in Sec. 3.1 of the main paper, we also try to sub-sample frames randomly. Similarly to evenly-spaced sub-sampling, starting from all frames, we randomly sub-sample up to 20 frames for each video. For training supervision, we keep again the setup of 10 randomly sampled points per frame per object. Also, we run each experiment 3 times for both evenly-spaced and random sub-sampling strategies and report the mean score.

We demonstrate the results for the random sub-sampling strategy on the DAVIS validation set in Fig. 2. We obtain very similar results for both strategies. We cannot observe a notable difference compared to Fig. 3 in the main paper, so we decided to make use of the evenly-spaced sub-sampling strategy.

Evaluating on temporally sparse videos. To annotate the ground-truth segmentation masks for the evaluation of the Point-VOS Oops (PV-Oops) benchmark, we also make the following key design decision. As the consecutive frames are extremely correlated and redundant, we question whether evaluating the results on a sparse subset of frames is sufficient. In that way, we would diminish the annotation effort while annotating the validation set as well, increasing cost and time-efficiency.

To this end, we run analysis experiments on DAVIS benchmark results. First, we generate temporally sparse validation sets from the DAVIS validation set by sub-sampling 3, 4, 5, and 10 frames evenly spaced, *i.e.*, we obtain 4 temporally sparse validation sets consisting of sub-sampled 3, 4, 5, or 10 frames. Then,we get the methods [1–6, 9–12, 15–18, 20, 21, 23, 24] from the DAVIS benchmark leaderboard, evaluate them on a sparse set of frames. Fi-

nally, we compare these results with the results on the temporally dense validation set, *i.e.*, the original DAVIS validation set with all frames. As seen in Fig. 3, the results are extremely correlated for all temporally sparse validation sets. In other words, even with only 3 ground-truth frames per object for evaluation, the ranking between methods does not change in almost all cases (except when their performance is extremely close to each other).

2. Annotating Point-VOS Oops Validation Set

We start annotating the Point-VOS Oops (PV-Oops) validation set by first annotating the reference frame with points. We generate point-wise annotations on the sub-sampled (evenly-spaced) 10 frames from each video and ask human annotators to verify them in the same way as for the training point annotations. Then, we check each video to decide the reference frame. In each video, we assign the first frame that contains at least 7 foreground points as the reference frame and remove all frames before the reference frame. In case, we cannot find a frame in the video with at least 7 foreground points, we eliminate the video. We also check whether we have enough frames after the reference frame. If there is no frame after the reference frame with at least 3 foreground points and 1 background point, we also drop the video.

Afterwards, we annotate the ground-truth segmentation masks for the evaluation of the PV-Oops benchmark. Informed by the simulation experiment for evaluating on a sparse subset of frames (see Sec. 1), we decided to annotate temporally sparse segmentation masks for the evaluation of the PV-Oops benchmark with 3 ground-truth frames.

While annotating 3 ground-truth frames, we start by first annotating the frame with the mouse trace segment for each video. Note that the mouse trace comes from the location-output questions of VidLN [19] for the PV-Oops dataset. In the original VidLN location-output task (which we do not consider in our work), a mask in the frame with the mouse trace is approximately evaluated by comparing it to the mouse trace. By annotating a segmentation mask for this frame, we make sure that our annotations can be used to replace the original VidLN evaluation, that compares the predicted mask with the mouse trace, with a more precise evaluation, that compares the predicted mask with the annotated mask.

After annotating the frame with mouse trace, we check each video and eliminate the videos, if the frame with the mouse trace is temporally before the reference frame, or exactly on the reference frame. From the remaining videos, we sub-sample (evenly-spaced) 3 frames from the frames coming after the reference frame with point annotations, and we check whether the frame with the mouse trace is in the 3 sub-sampled frames. If the frame with the mouse trace is in the 3 sub-sampled frames, we keep the other 2 sub-sampled frames and annotate them with ground-truth masks. If the frame with the mouse trace is not in the 3 sub-sampled frames, we drop the frame that is temporally closest to the mouse trace frame and send the other 2 frames to annotation.

	Dataset	Videos	Annotations	Objects	Positive Points	Negative Points	Ambiguous Points
train	Point-VOS Oops	7.4K	93K	12K	541K	1.2M	18K
	Point-VOS Kinetics	23.9K	965K	120K	5.2M	12.6M	253K
	Point-VOS DAVIS	60	600	145	9.7K	6K	—
	Point-VOS YouTube	3471	34.6K	6.4K	472K	346K	—
val	Point-VOS Oops	991	3.5K	991	7.3K	9.9K	91
	Point-VOS DAVIS	30	1.9K	61	558	300	—
	Point-VOS YouTube	507	614	1K	9.8K	6K	—

Table 1. **Statistics for the Point-VOS datasets.** Annotations here means summing up frames containing at least one annotated object. Note that for Point-VOS DAVIS and Point-VOS YouTube, we sampled the points from the ground truth masks, while for all other datasets, we annotated new points.

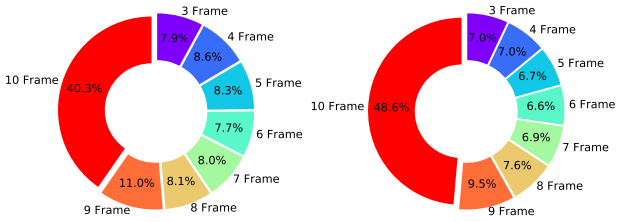
3. Point-VOS Datasets Statistics

Overview. In Tab. 1, we present the detailed statistics for the training and validation splits of the Point-VOS datasets.

Point-VOS Oops (PV-Oops) and Point-VOS Kinetics (PV-Kinetics) are the datasets that we annotated with new points. In total, we collected $19.7M$ points where $5.8M$ points are annotated as positive points and $13.9M$ points as negative points. Also, $271K$ points are annotated as ambiguous points. We do not use any ambiguous annotations in our experiments.

In PV-Oops, there are $541K$ positive points and $1.2M$ negative points in the training split, and also $7.3K$ positive points and $9.9K$ negative points in the validation split. In PV-Kinetics, there are $5.2M$ positive points and $12.6M$ negative points.

In addition to the PV-Oops and PV-Kinetics datasets, we also generated the Point-VOS versions of the DAVIS and YouTube-VOS (YT-VOS) datasets. For Point-VOS DAVIS (PV-DAVIS) and Point-VOS YouTube (PV-YT), we sample the spatially temporally sparse points from the ground truth masks. Since the original DAVIS and YT-VOS datasets are massively smaller than PV-Oops and PV-Kinetics, the total positive and negative points are also very much less in PV-DAVIS and PV-YT. There are $9.7K$ positive points and $6K$ negative points in the PV-DAVIS training split, 558 positive and 300 negative points in the PV-DAVIS validation split. PV-YT contains $472K$ positive and $346K$ negative points in the training split, and $9.8K$ positive and $6K$ negative points in the validation split. Note that there are fewer annotations in both PV-DAVIS and PV-YT compared to the original DAVIS and YT-VOS datasets as we sub-sample 10 frames.



(a) PV-Oops (b) PV-Kinetics

Figure 4. **The distribution of frames for PV-Oops and PV-Kinetics.** The distribution of frames means summing up frames in each video, which contains at least one positive point annotation.

Frame Distribution. In addition to the detailed statistics, we also analyze the distribution of frames in the training splits of PV-Oops and PV-Kinetics. During the annotation process, we provided 10 frames to the human annotators for annotations. Here, the distribution of frames means, we check each video after the annotation process and sum up the frames in each video, which have at least one positive point annotation.

Fig. 4 shows the frame distribution for PV-Oops (see Fig. 4a) and PV-Kinetics (see Fig. 4b). As seen, more than 40% of the videos in both PV-Oops and PV-Kinetics have all frames with positive point annotations (see red slice). Also, more than 30% of the videos in both PV-Oops and PV-Kinetics contain more than 5 frames with positive point annotations (see chameleon, green, caramel macchiato and orange slices).

Point Distribution. Finally, we analyze the distribution of the positive and negative points in the training splits of PV-Oops and PV-Kinetics. Here, the distribution of points means reporting the total number of videos in the different ranges of the number of point annotations.

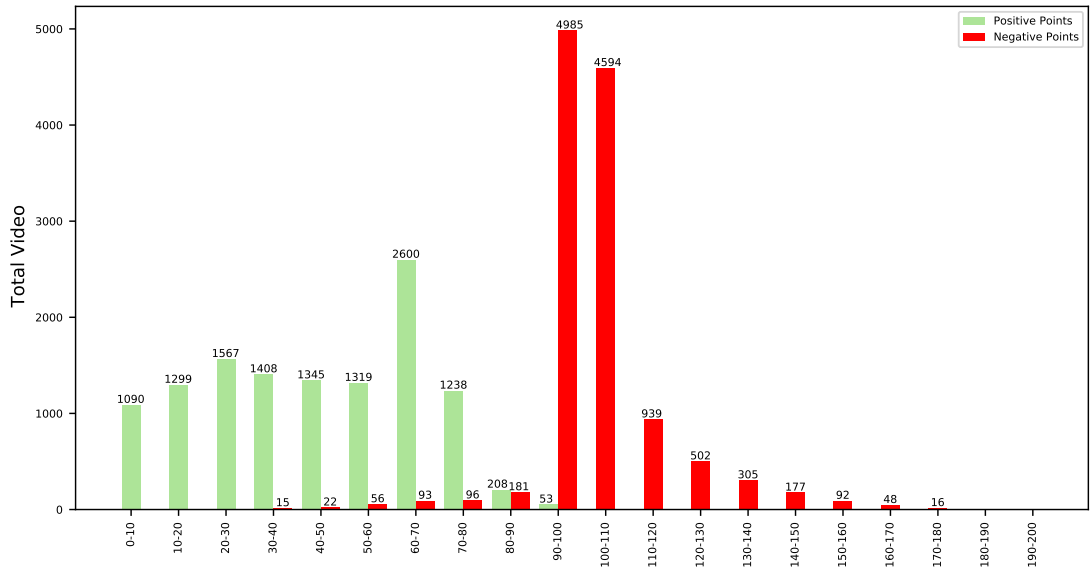
We show the distribution of points in Fig. 5 for PV-Oops (see Fig. 5a) and PV-Kinetics (see Fig. 5b). As seen, we observe similar point distributions in both PV-Oops and PV-Kinetics. As the size of the objects varies, the distribution of the positive points has more probability mass on the left than the distribution of the negative points in both PV-Oops and PV-Kinetics. Since we fixed the number of background points to 10 points for annotating, the distribution of the negative points has probability mass at the center for both PV-Oops and PV-Kinetics.

4. Implementation Details

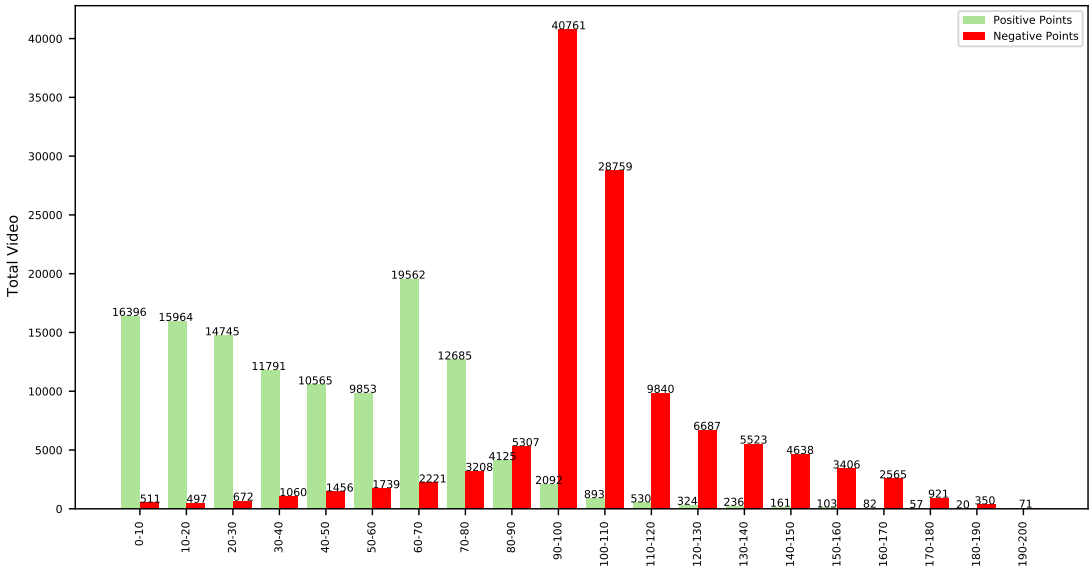
Point-STCN. A major advantage of using point annotations is that it can be used to train existing VOS models without making drastic changes to either the inherent model or the training strategy. We show this by easily adapting STCN to work with our point annotations while keeping most of the network structure intact. Specifically, we make the following modifications to STCN: (i) The value encoder of STCN now takes a set of sparse points (that we represent as a sparse segmentation mask) for each of the reference foreground objects in the first frame mask instead of the dense pixel-level masks. To leverage these point annotations, similar to the original STCN pre-processing pipeline, we apply augmentations like affine transformations and convert the points into a mask that has only non-zero elements on the locations of the points. We concatenate the point masks with the input image which is then processed by the value encoder. (ii) Instead of using a bootstrapped cross-entropy loss on the predicted dense posterior probabilities, we use a point-wise cross entropy loss where the loss is applied to only the output vectors at sparse point locations that are annotated in the ground-truth. We use bilinear interpolation on the output probability map to approximate the predictions on the precise point locations. During training, we use both the positive and the negative points for the loss computation. For each training sample, we sample 3 frames from a video. One of those frames is considered the reference frame which we used for initialization. The two other frames are considered the target frames, on which we calculate the loss. Only the positive (foreground) points are used as initialization in the reference (first) frame during both training and testing, while both positive and negative points are used to calculate the loss on the target frames.

DynaMITe Adaptation. DynaMITe [14] was originally designed to process user interactions in the form of user clicks. Since, for our point annotation scheme, only a mouse trace is available on the reference frames for each foreground object, we adapt DynaMITe to work with a trace as input for generating a reference mask. Those reference masks are later fed to STCN for propagation (see Sec. 3.2 of the main paper). To adapt DynaMITe, we first sample the image features that correspond to each of the pixel-locations covered by the input mouse trace, and perform a *global average pooling* operation to generate a single feature vector. This feature vector is then projected linearly to generate a query that corresponds with the trace, similar to the click features in DynaMITe. This query is then used by the *Interactive Transformer* module in DynaMITe to generate the output mask for the object of interest.

Training Details. We train Point-STCN with points and STCN with pseudo-masks on Point-VOS DAVIS (PV-DAVIS) and Point-VOS YouTube-VOS (PV-YT) jointly for a total of 38K iterations. The learning rate is reduced after 30K steps. On Point-VOS Oops (PV-Oops), Point-STCN and STCN are trained in total 60K iterations, and the learning rate is reduced after 50K steps. On Point-VOS Kinetics (PV-Kinetics), and also joint training on PV-Oops and PV-Kinetics, we train Point-STCN and STCN in total 190K iterations and reduce the learning rate after 150K steps.



(a) PV-Oops



(b) PV-Kinetics

Figure 5. **The distribution of the positive and negative points in PV-Oops and PV-Kinetics.** The x-axis represents the different ranges for the number of points, and the y-axis represents the total number of videos. Also, we show the precise numbers for the total videos at the top of the bars.

Following the original STCN setup, when training jointly on PV-DAVIS and PV-YT, we build a combined dataset by repeating the PV-DAVIS dataset 5 times and PV-YT 1 time, to compensate for the smaller size of PV-DAVIS. Similarly, when training jointly on PV-Oops and PV-Kinetics, we build a combined dataset by repeating PV-Oops 5 times and PV-Kinetics 1 time in order to compensate for the smaller size of PV-Oops.

Moreover, for each training of Point-STCN and STCN, we use Adam [7] and start with a learning rate of 10^{-5} and reduce it to 10^{-6} after a certain number of training steps as indicated above. We set the weight decay to 10^{-7} and the batch size to 4. We conduct all STCN and Point-STCN trainings with 8 V100 GPUs, and all inference experiments

on a single 3090 GPU.

For training ReferFormer, we closely follow the setup used by VidLN [19].

Hybrid Task. In Sec. 4.1 of the main paper, we introduced the *Hybrid* task (a task in between VOS and Point-VOS). In the VOS task, dense segmentation masks are used both during training and for test-time initialization, while, in the Point-VOS task, spatially temporally sparse point annotations are used in both cases. For the Hybrid task, spatially and temporally dense masks are used during training, while only points are used on the reference frame at test-time. This means that the Hybrid task follows the setup from VOS at training time, while it follows the setup from Point-VOS at test-time.

In the Hybrid setup, we make use of dense masks to train Hybrid-STCN while we initialize the reference frame with sparse points. Recall that STCN uses 3 frames during training, from which one is the reference frame and two are the target frames. In the Hybrid setup, we initialize STCN with points in the reference frame and apply a full mask loss in the target frames. At test-time, Hybrid-STCN can then be initialized with points and achieves better results than Point-STCN, as we use more supervision during training.

5. Additional Qualitative Results

In Fig. 6 and Fig. 7, we provide the additional example point annotations for Point-VOS Oops (PV-Oops) and Point-VOS Kinetics (PV-Kinetics). We successfully annotated multi-modal points for different and challenging scenes, and also the objects from a large vocabulary.

In Fig. 8 and Fig. 9, we also show the examples of ambiguous point annotations from PV-Oops and PV-Kinetics, *i.e*. the point annotations where the human annotators indicated that they were unsure. We observe that we have ambiguous point annotations in particular cases for both PV-Oops and PV-Kinetics, *e.g*., if the given point is in a challenging lighting condition or at the border.

In Fig. 10, Fig. 12, and Fig. 14, we present the tracking results of Point-STCN (trained with points) on Point-VOS Oops (PV-Oops), Point-VOS DAVIS (PV-DAVIS) and Point-VOS YouTube (PV-YT), respectively. Also, in Fig. 11, Fig. 13 and Fig. 15, we demonstrate the results of STCN [3] (trained with pseudo-masks) on PV-Oops, PV-DAVIS and PV-YT, respectively.

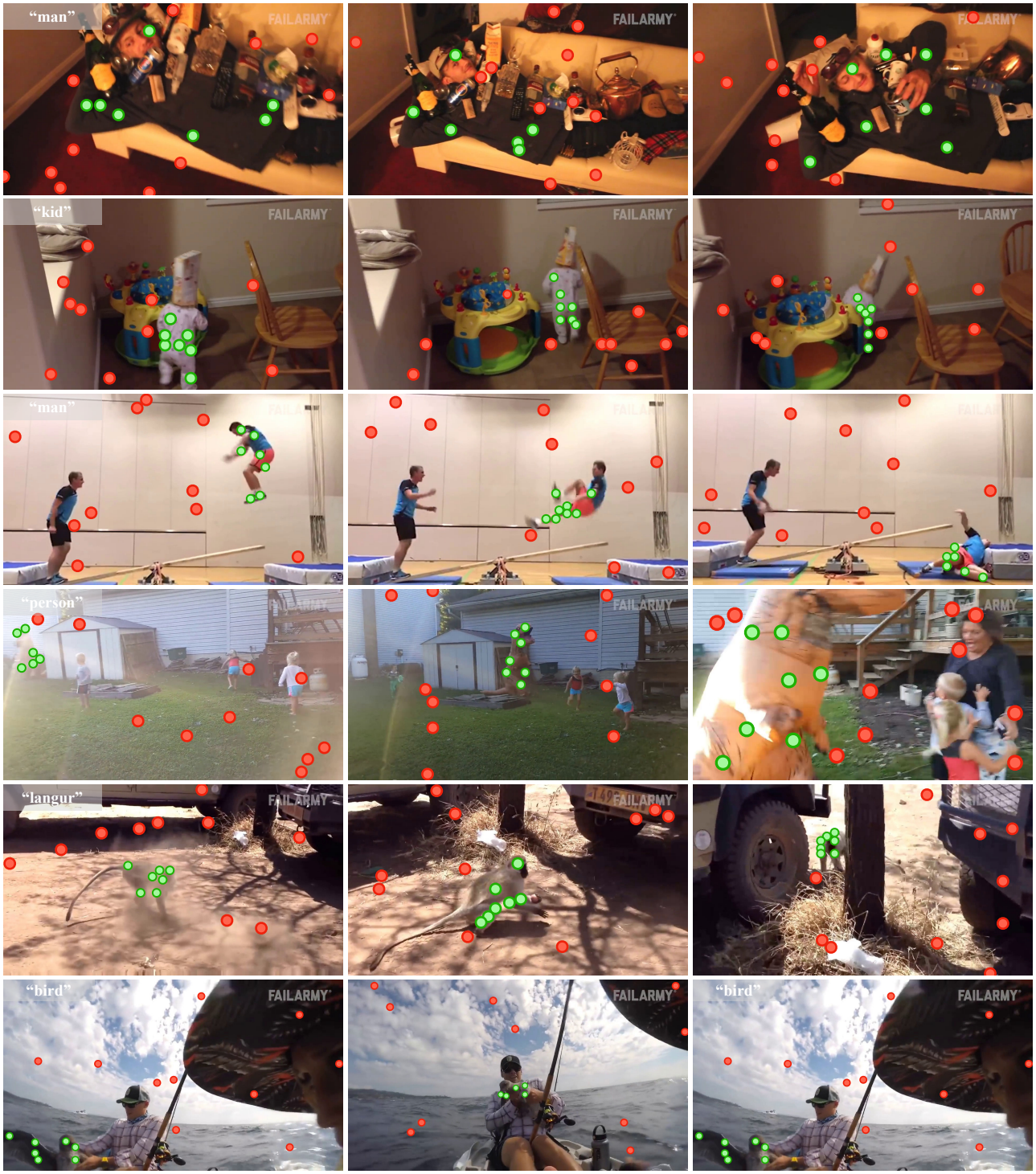


Figure 6. **Additional example point annotations for Point-VOS Oops.** We are able to have multi-modal point annotations in cluttered scenes (*first row*), fast motion (*third row*), challenging lighting conditions (*fourth row*), and motion blur (*fifth row*). Green dots represent positive points and red dots negative points.

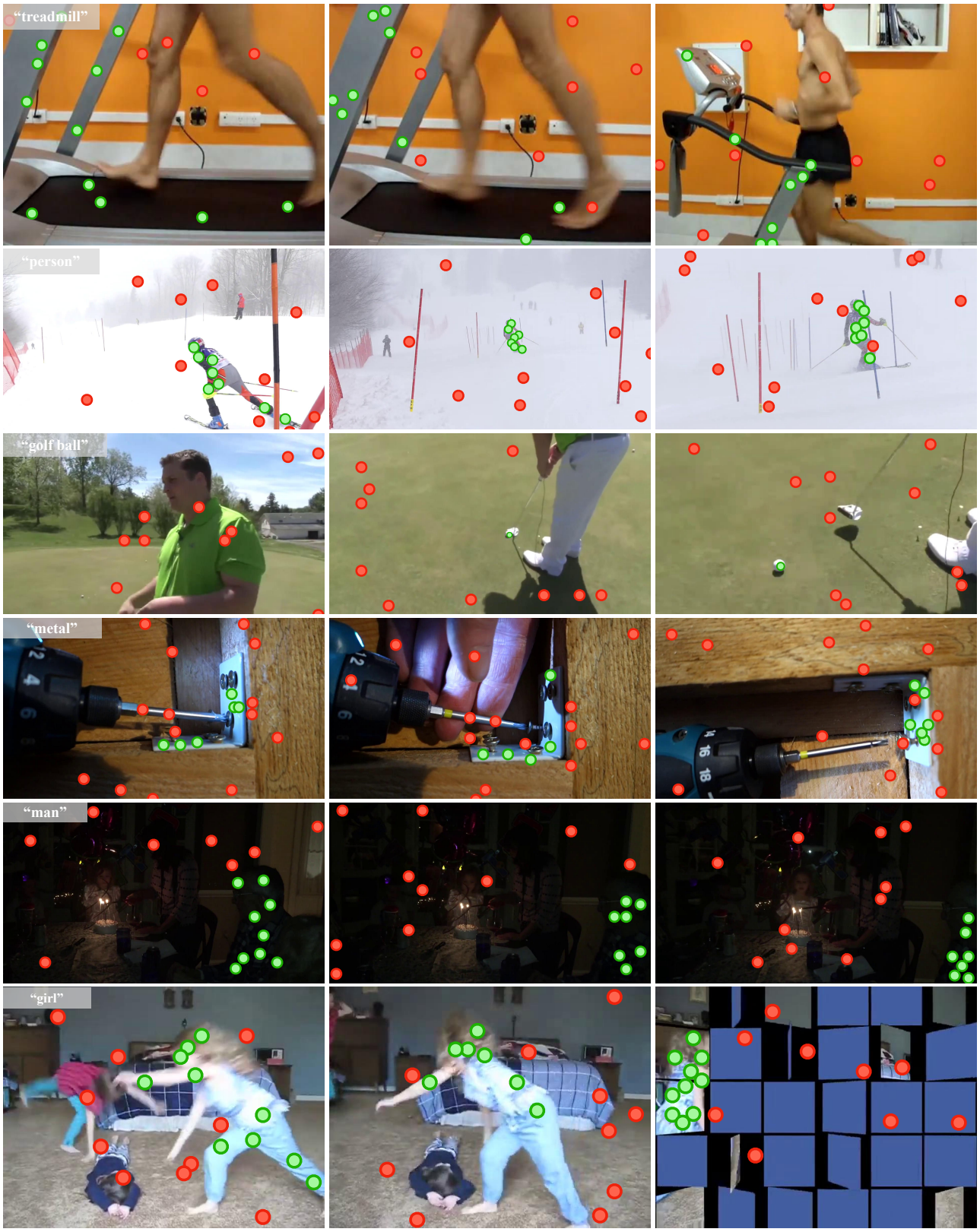


Figure 7. **Additional example point annotations for Point-VOS Kinetics.** We can provide multi-modal point-wise annotations for objects from a large vocabulary (*first and fourth rows*), scenes with fast motion (*second row*), small objects (*third row*), and also scenes with difficult lighting conditions (*fourth row*) and motion blur (*fifth row*). Green dots represent positive points and red dots negative points.

Figure 8. **Example ambiguous point annotations from Point-VOS Oops.** We observe that the human annotators indicate unsure if the given point is in challenging lighting condition (*first row)*) or at border (*second row*), or at motion blur (*third row*), or if the object is ambiguous (*fourth row*). Green dots represent positive annotations, red dots negative annotations, and gray dots ambiguous annotations.

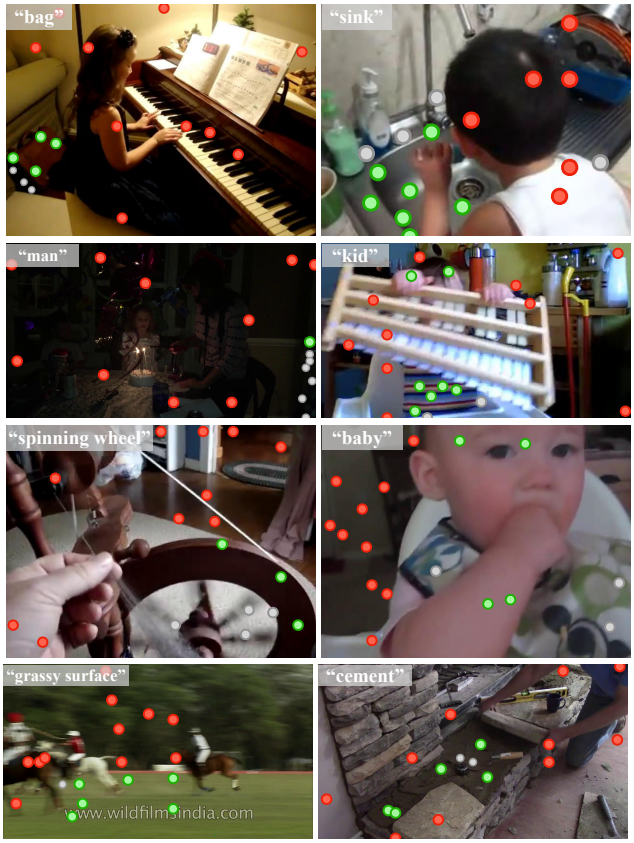


Figure 9. **Example ambiguous point annotations from Point-VOS Kinetics.** Similarly, the human annotators indicate unsure if the given point is in challenging lighting condition (*first column, first two rows*) or at border (*second column, first two rows*), or at motion blur (*first column, last two rows*), or if the object is ambiguous (*second column, last two rows*). Green dots represent positive annotations, red dots negative annotations, and gray dots ambiguous annotations.

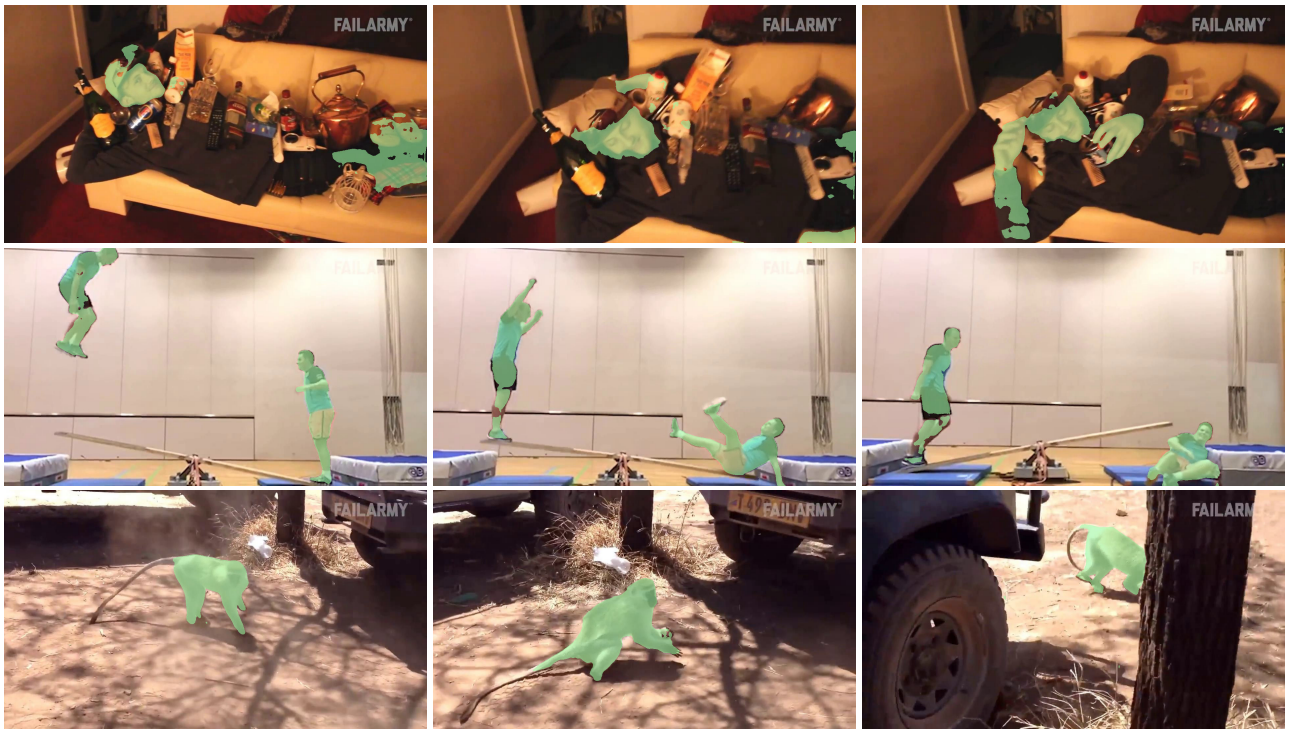


Figure 10. **Tracking results of Point-STCN on PV-Oops.** The model is trained on PV-Oops with points, then evaluated on the 10-point setup.

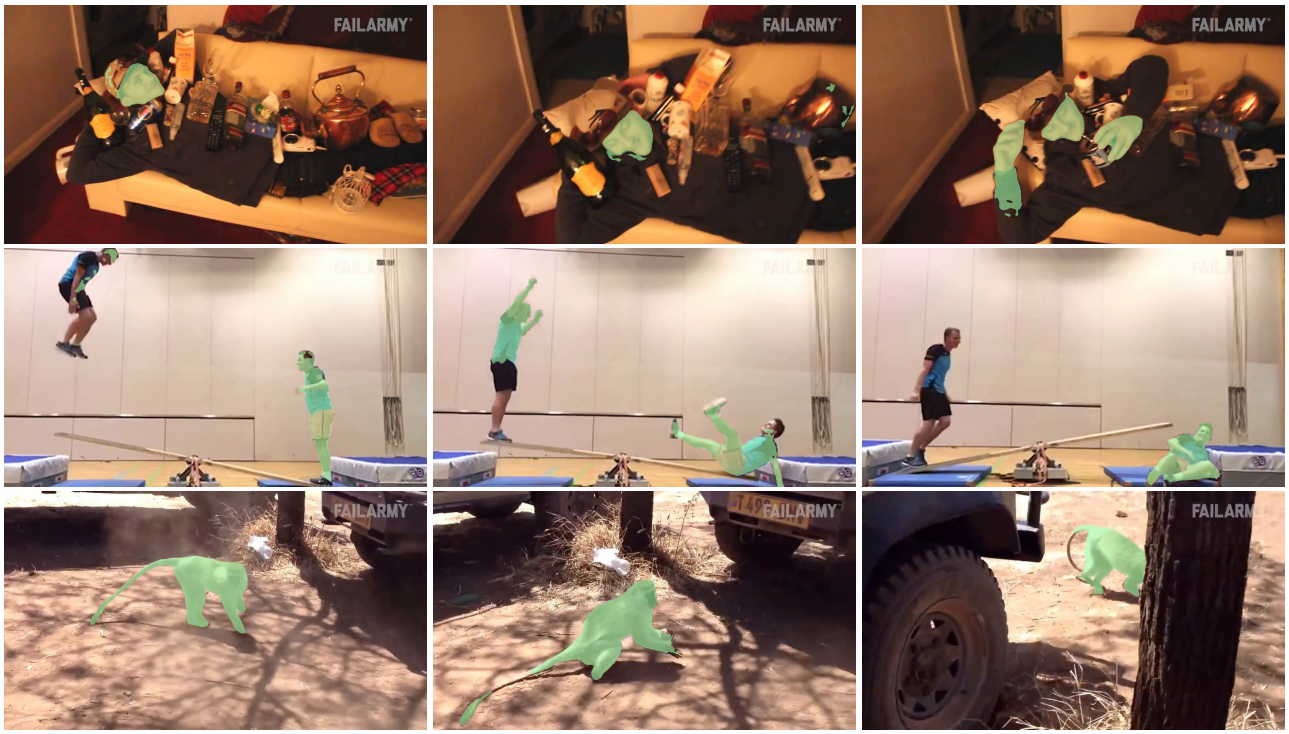


Figure 11. **Tracking results of STCN [3] on PV-Oops.** The model is trained on PV-Oops with DynaMITe [14] pseudo-masks, then evaluated on 10 points setup.

Figure 12. **Tracking results of Point-STCN on PV-DAVIS.** The model is first pre-trained on PV-Oops and PV-Kinetics with points, then fine-tuned on PV-DAVIS and PV-YT with points, and finally evaluated on the 10-point setup.

Figure 13. **Tracking results of STCN [3] on PV-DAVIS.** The model is pre-trained on PV-Oops and PV-Kinetics with pseudo-masks, then fine-tuned on PV-DAVIS and PV-YT with pseudo-masks, and finally evaluated on 10 points setup. The pseudo-masks are generated from SAM [8].

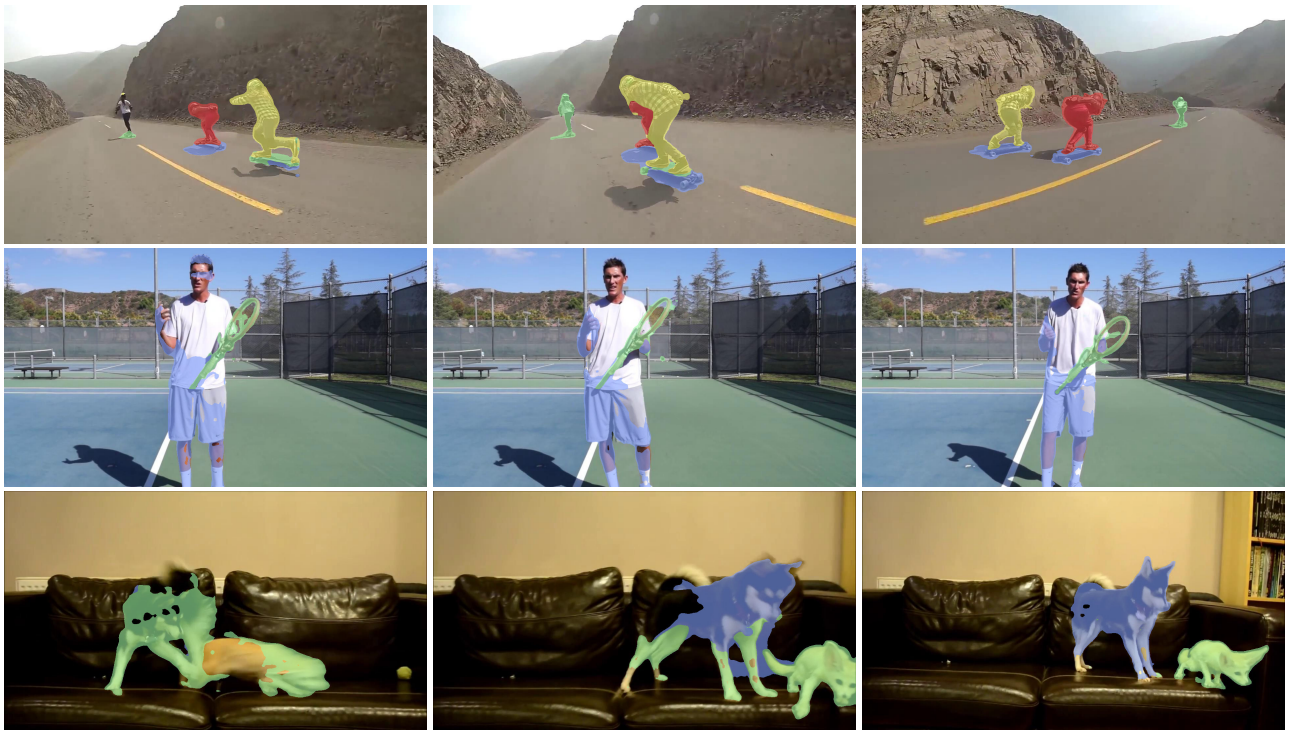


Figure 14. **Tracking results of Point-STCN on PV-YT.** The model is first pre-trained on PV-Oops and PV-Kinetics with points, then fine-tuned on PV-DAVIS and PV-YT with points, and finally evaluated on the 10-point setup.

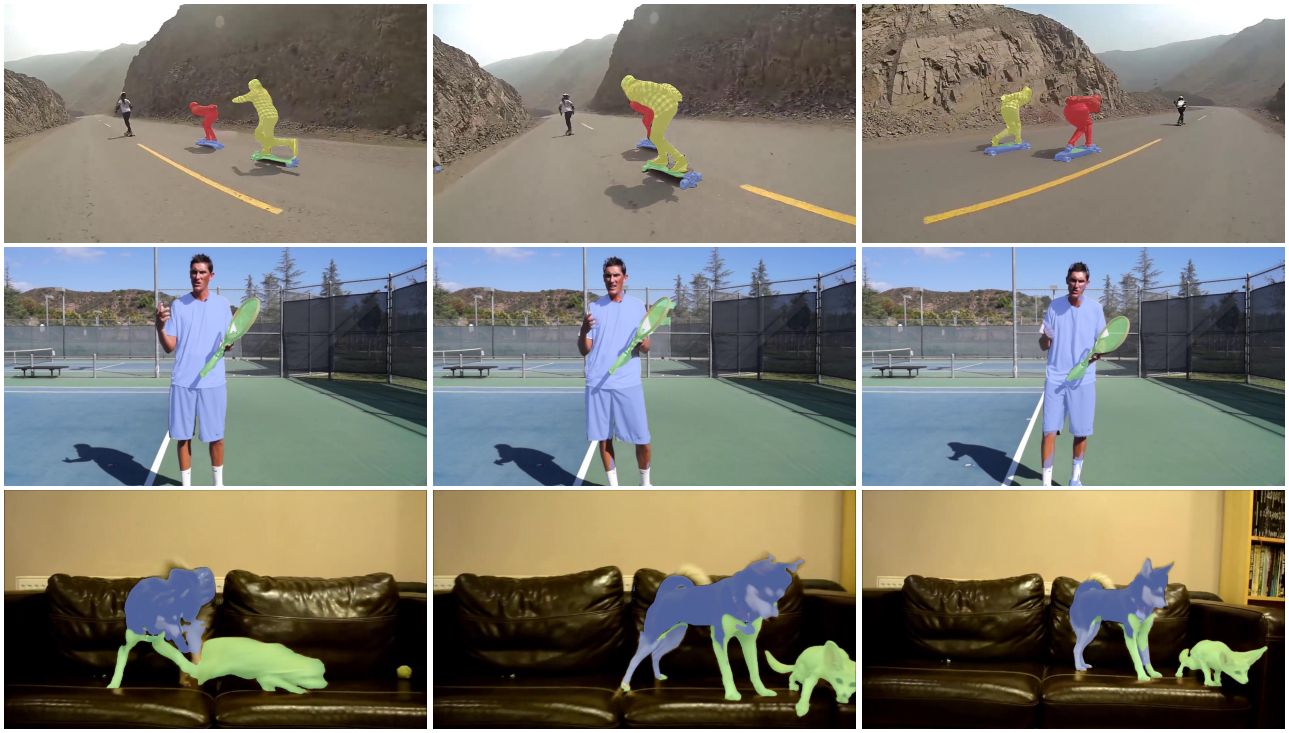


Figure 15. **Tracking results of STCN [3] on PV-YT.** The model is pre-trained on PV-Oops and PV-Kinetics with pseudo-masks, then fine-tuned on PV-DAVIS and PV-YT with pseudo-masks, and finally evaluated on 10 points setup. The pseudo-masks are generated from SAM [8].

References

[1] Linchao Bao, Baoyuan Wu, and Wei Liu. CNN in MRF: Video object segmentation via inference in a CNN-based higher-order spatio-temporal MRF. In *CVPR*, 2018. 2

[2] Sergi Caelles, Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. One-Shot Video Object Segmentation. In *CVPR*, 2017.

[3] Ho Kei Cheng, Yu-Wing Tai, and Chi-Keung Tang. Rethinking Space-Time Networks with Improved Memory Coverage for Efficient Video Object Segmentation. In *NeurIPS*, 2021. 6, 10, 11, 12

[4] J. Cheng, Y.-H. Tsai, W.-C. Hung, S. Wang, and M.-H. Yang. Fast and Accurate Online Video Object Segmentation via Tracking Parts. In *CVPR*, 2018.

[5] Joakim Johnander, Martin Danelljan, Emil Brissman, Fahad Shahbaz Khan, and Michael Felsberg. A generative appearance model for end-to-end video object segmentation. In *CVPR*, 2019.

[6] Anna Khoreva, Anna Rohrbach, and Bernt Schiele. Video object segmentation with language referring expressions. In *ACCV*, 2019. 2

[7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[8] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *ICCV*, 2023. 11, 12

[9] Jonathon Luiten, Paul Voigtlaender, and B. Leibe. PReMVOS: Proposal-generation, Refinement and Merging for Video Object Segmentation. In *ACCV*, 2018. 2

[10] Kevis-Kokitsi Maninis, Sergi Caelles, Yuhua Chen, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. Video Object Segmentation Without Temporal Information. In *IEEE TPAMI*, 2018.

[11] Seoung Wug Oh, Joon-Young Lee, Kalyan Sunkavalli, and Seon Joo Kim. Fast Video Object Segmentation by Reference-Guided Mask Propagation. In *CVPR*, 2018.

[12] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *ICCV*, 2019. 2

[13] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool. The 2017 DAVIS Challenge on Video Object Segmentation. *arXiv:1704.00675*, 2017. 1

[14] Amit Rana, Sabarinath Mahadevan, Alexander Hermans, and Bastian Leibe. DynaMITe: Dynamic Query Bootstrapping for Multi-object Interactive Segmentation Transformer. In *ICCV*, 2023. 4, 10

[15] Carles Ventura, Miriam Bellver, Andreu Girbau, Amaia Salvador, Ferran Marques, and Xavier Giro-i Nieto. Rvos: End-to-end recurrent network for video object segmentation. In *CVPR*, 2019. 2

[16] Paul Voigtlaender and Bastian Leibe. Online Adaptation of Convolutional Neural Networks for the 2017 DAVIS Challenge on Video Object Segmentation. In *CVPRW*, 2017.

[17] Paul Voigtlaender, Yuning Chai, Florian Schroff, Hartwig Adam, Bastian Leibe, and Liang-Chieh Chen. Feelvos: Fast end-to-end embedding learning for video object segmentation. In *CVPR*, 2019.

[18] Paul Voigtlaender, Jonathon Luiten, Philip HS Torr, and Bastian Leibe. Siam r-cnn: Visual tracking by re-detection. In *CVPR*, 2020. 2

[19] Paul Voigtlaender, Soravit Changpinyo, Jordi Pont-Tuset, Radu Soricut, and Vittorio Ferrari. Connecting Vision and Language with Video Localized Narratives. In *CVPR*, 2023. 3, 6

[20] Qiang Wang, Li Zhang, Luca Bertinetto, Weiming Hu, and Philip HS Torr. Fast online object tracking and segmentation: A unifying approach. In *CVPR*, 2019. 2

[21] Ziqin Wang, Jun Xu, Li Liu, Fan Zhu, and Ling Shao. RANet: Ranking Attention Network for Fast Video Object Segmentation. In *ICCV*, 2019. 2

[22] Ning Xu, Linjie Yang, Yuchen Fan, Dingcheng Yue, Yuchen Liang, Jianchao Yang, and Thomas Huang. Youtube-vos: A large-scale video object segmentation benchmark. *arXiv preprint arXiv:1809.03327*, 2018. 1

[23] Shuangjie Xu, Daizong Liu, Linchao Bao, Wei Liu, and Pan Zhou. MHP-VOS: Multiple Hypotheses Propagation for Video Object Segmentation. In *CVPR*, 2019. 2

[24] L. Yang, Yanran Wang, Xuehan Xiong, Jianchao Yang, and Aggelos K. Katsaggelos. Efficient Video Object Segmentation via Network Modulation. In *CVPR*, 2018. 2