Point-VOS: Pointing Up Video Object Segmentation

Idil Esen Zulfikar[1,*]   Sabarinath Mahadevan[1,*]   Paul Voigtlaender[2,*]   Bastian Leibe[1]

[1]RWTH Aachen University, Germany   [2]Google Research

[1]{zulfikar, mahadevan, leibe}@vision.rwth-aachen.de [2]voigtlaender@google.com

pointvos.github.io

Abstract

Current state-of-the-art Video Object Segmentation (VOS) methods rely on dense per-object mask annotations both during training and testing. This requires time-consuming and costly video annotation mechanisms. We propose a novel Point-VOS task with a spatio-temporally sparse point-wise annotation scheme that substantially reduces the annotation effort. We apply our annotation scheme to two large-scale video datasets with text descriptions and annotate over $19M$ points across $133K$ objects in $32K$ videos. Based on our annotations, we propose a new Point-VOS benchmark, and a corresponding point-based training mechanism, which we use to establish strong baseline results. We show that existing VOS methods can easily be adapted to leverage our point annotations during training, and can achieve results close to the fully-supervised performance when trained on pseudo-masks generated from these points. In addition, we show that our data can be used to improve models that connect vision and language, by evaluating it on the Video Narrative Grounding (VNG) task. We will make our code and annotations available at https://pointvos.github.io.

1. Introduction

Video Object Segmentation (VOS) has grown into a very popular research field [6, 39, 51, 53] that has shown considerable progress over the past few years [11, 13, 70], branching out into new downstream tasks with language referring expressions [25] or user interactions [8]. New datasets have been instrumental in advancing progress in VOS [3, 17, 21, 52, 59, 66]. However, the relatively costly annotation process necessary for creating VOS datasets has so far been a major limiting factor. The traditional VOS task requires temporally dense object segmentation masks for the frames of each training video. As a result, existing

Figure 1. **Comparison of the conventional VOS task with our new Point-VOS task.** (a) The conventional VOS task utilizes dense segmentation mask for each frame during training and initializes the first-frame reference with dense masks. (b) We propose to change this paradigm and use only spatially sparse point annotations on a sparse subset of frames during training, and only a few points for the first-frame reference initialization. Green and blue dots represent foreground points and red dots background points.

video segmentation datasets [21, 39, 40, 52] are usually relatively small in scale, and past community efforts to scale them up to at least several thousand videos required substantial annotation effort [3, 15, 17, 66]. Given the community's clear trend to connect vision to language [42, 60] and the consequent need for even larger datasets [44], there is thus an urgent need to reduce the annotation cost for videos.

Some approaches try to mitigate this problem by reducing the reliance of vision models on annotated training data, *e.g.*, by self-supervised learning [61, 64] or exploiting image-level mask annotations [1]. Another strategy has been to create semi-automatic annotation pipelines [59] that generate pseudo ground-truth masks from more readily available data, such as existing bounding box annotations. Nevertheless, neither of those presents a general solution.

In this work, we address the annotation cost problem by proposing an entirely point-based VOS framework, Point-

---

* Equal contribution. The ordering of the authors was determined by a last-minute coin flip.

VOS. Inspired by recent point-guided image segmentation methods [5, 10], Point-VOS moves away from using full mask supervision and instead relies on spatio-temporally sparse point annotations as weak supervision signals for VOS (see Fig. 1). Our point-based formulation enables us to design an efficient semi-automatic annotation pipeline (see Fig. 4) that requires substantially less annotation effort to create human-validated ground-truth video annotations.

We demonstrate the value of our proposed annotation pipeline by annotating two large-scale video datasets, Point-VOS Oops [19] (PV-Oops) and Point-VOS Kinetics [23] (PV-Kinetics), with altogether $19M$ points for $133K$ objects in $32K$ videos. Our annotations cover almost an order of magnitude more videos and objects than major previous VOS datasets [3, 17, 21, 39, 40, 52, 59, 66] (see Tab. 1). In particular, we show how our annotation pipeline can make use of existing information from the Video Localized Narratives corpus (VidLN [60]) in order to bootstrap the annotation process by automatically converting mouse traces from VidLN into Point-VOS initializations.

We launch a new Point-VOS benchmark based on these datasets, where VOS methods are expected to use only point annotations both as training supervision and as test-time initialization. We also develop two strong baselines by (i) adapting the state-of-the-art VOS method STCN [13] to work directly with points instead of masks, and (ii) training STCN on pseudo-masks generated from point annotations. Our experiments show that, despite the weaker level of supervision, this Point-VOS STCN baseline already reaches more than 90% of the performance of the original STCN when applied to the DAVIS benchmark [40] (see Sec. 4.2).

Finally, we also show a direct use case of our point annotations for language-guided VOS. As a consequence of our use of VidLN data to bootstrap the point annotation pipeline, the point annotations in PV-Oops and PV-Kinetics are connected to nouns in longer language captions (the Video Localized Narratives), describing the referred object's actions in the video. Thus, our annotations are multi-modal and bridge the gap between open-vocabulary language object descriptions and the corresponding video object segmentations. We showcase the usefulness of the multi-modal annotations by training a Video Narrative Grounding (VNG) [60] model using our datasets, resulting in significant improvements on two VNG benchmarks.

In summary, we make the following contributions: **(1)** We propose the new Point-VOS task for point-guided VOS, that includes weakly supervised training on spatio-temporally sparse point annotations. **(2)** We propose a novel and efficient semi-automatic annotation pipeline for Point-VOS that substantially reduces the annotation effort for creating human-validated ground-truth video annotations. **(3)** We demonstrate the value of our proposed annotation pipeline by annotating and releasing two large

Dataset	Videos	Objects	Annotations	Positive Points	Negative Points	Ambiguous Points
DAVIS'16 [39]	50	50	3.4K	-	-	-
DAVIS'17 [40]	90	205	13.5K	-	-	-
YT-VOS [66]	4.4K	7.7K	197K	-	-	-
BURST [3]	2.9K	16K	600K	-	-	-
VISOR [15]	7.8K	†	271K	-	-	-
VOST [52]	713	†	175K	-	-	-
MOSE [17]	2.1K	5.2K	431K	-	-	-
PV-Oops	8.4K	13.1K	93K	548K	1.2M	18K
PV-Kinetics	23.9K	120K	965K	5.2M	12.6M	253K

Table 1. **Comparison of VOS datasets with ours.** Our Point-VOS data is much larger compared to existing VOS datasets. "Annotations" counts for each object in how many frames it is annotated. †: The number of objects is not reported.

video datasets, PV-Oops and PV-Kinetics. By design, those datasets feature multi-modal vision-language annotations that connect open-vocabulary language object descriptions to the corresponding video object annotations. **(4)** We establish a new benchmark based on these datasets, where we train and test VOS methods either on point annotations or on pseudo masks, and present strong baselines. **(5)** We realize the potential of multi-modal vision-language annotations in our proposed datasets and showcase their use for language-guided VOS.

2. Related Work

Video Object Segmentation Datasets. DAVIS [39, 40] is one of the first densely annotated VOS datasets, with 90 videos. Later, the YouTube-VOS (YT-VOS) dataset [66] with $4.4K$ videos further advanced the state-of-the-art. Later datasets [3, 15, 17, 41, 52] focused on specific VOS sub-challenges; among them, VISOR [15] is the largest in terms of the number of videos with $7.8K$ kitchen videos. Despite the growing interest in the VOS task, VOS datasets are still small in scale mainly due to their expensive annotation process. In contrast, we introduce a much more efficient point-wise annotation scheme that enables us to annotate about $32K$ videos, 4 times more than VISOR.

Fully-Supervised VOS Methods. Early VOS methods [7, 31, 49, 55, 56] use online learning at test time which makes them very slow. The following methods [37, 57, 69] alleviate this by propagating predictions frame-by-frame, using limited context and accumulating errors on the way. Recent methods [11, 13, 38] address this by incorporating a larger temporal context using an external memory (*e.g.*, STM [38] and its extensions [11–13, 46]). More recent works [1, 2, 70] use Transformers with spatio-temporal attention. All of these methods rely on dense segmentation masks. In contrast, we use much cheaper point annotations in our work.

Weakly-Supervised VOS Methods. Weakly supervised

	VOS	PET	TAP-Vid	Point-VOS
Point annotations for training	✗	✗	✓	✓
Point annotations for test-time	✗	✓	✓	✓
Arbitrary point location on object	✗	✓	✗	✓
Temporally sparse annotations	✗	✗	✗	✓
Simple, fast & efficient annotations	✗	✗	✗	✓
Multi-modal annotations	(✓)	✗	✗	✓

Table 2. **Comparison of the design decisions of our new spatially-temporally sparse point annotation scheme in Point-VOS with the annotation schemes in other tasks**: VOS [39], Point Exemplar-guided Tracking (PET) [3] and Tracking Any Point in a Video (TAP-Vid) [18]. (✓): Multi-modal annotations are only available for some extensions of VOS datasets [16, 25, 45] that initialize by a referring expression.

VOS approaches can be mainly classified into two types: (i) The first type aims at reducing the first-frame supervision at test time by either using points [3] or bounding boxes [29, 50, 58, 62, 71] instead of dense masks while still relying on dense annotations for training. (ii) The second type [1, 24, 59] reduces the training supervision by exclusively using image-level datasets or by using bounding boxes instead of masks, while still relying on dense masks for the reference at test time. Unlike these methods, we use weak point supervision both at training and test times.

Point Supervision for Images. Point supervision during training has already been explored for various image tasks such as instance segmentation [5, 9, 10, 27, 32, 43, 47, 48], action recognition [35, 36] and object counting [28]. [10] annotate points on objects and show that instance segmentation methods can be effectively trained on them. [5] use an efficient mobile friendly point annotation scheme to collect a new image dataset. However, unlike our method, these methods work only with images and sometimes require object-level bounding boxes [10] for annotating points.

Point Supervision for Videos. Existing video-level point based annotations [18, 72] are mainly used for point tracking, which requires point correspondences that are again expensive to obtain and not relevant for tasks such as VOS. In contrast, we focus on annotating *random* points, *i.e.*, we want points that are on a certain object, but we do not require points in different frames to correspond to the exact same part of the object. To the best of our knowledge, [3] is the only previous attempt that utilizes point annotations for VOS-related tasks. However, different from our work, [3] makes use of point annotations only to initialize the first frame at test time while keeping the training supervision unchanged.

Video Annotations. We also compare the design decisions of the point annotation scheme in Point-VOS with existing video annotation schemes in Tab. 2. Conventional VOS

Figure 2. **Training *vs*. test-time point supervision results using simulated points on the DAVIS validation set.** ♦ represents our chosen setting, i.e. 10 points for training supervision and 10 points for test-time supervision. We run each experiment 5 times and report the mean score.

uses dense masks that are expensive to annotate. PET [3] uses point initialization at test-time but still dense masks for training. TAP-Vid [18] uses exact point correspondences for training that are extremely costly to annotate. In contrast, Point-VOS expects that only random points on objects are annotated and the annotation effort is even further reduced by making use of temporal sparsity.

3. Point Annotations for VOS

In the conventional VOS task, the training set contains N videos $V=\{v_1,v_2,...,v_N\}$, where each video $v \in V$ with T_v frames and O_v objects consist of a set of images $I_v=\{I_1,I_2,...,I_{T_v}\}$ and a set of dense segmentation masks $M_v=\{m_{t,o}|t\in\{1,...,T_v\},o\in\{1,...,O_v\}\}$. At test-time, the input is a video $v \notin V$ and the corresponding reference segmentation masks $M^R=\{m_1^R,m_2^R,...,m_{O^R}^R\}$ for O^R objects in a single frame (usually the first). The conventional VOS method then has to generate temporally consistent segmentation masks $\hat{M}_v=\{\hat{m}_{t,o}|t\in\{1,...,T_v\},o\in\{1,...,O^R\}\}$ for the O^R foreground objects for each frame of the video.

In our proposed Point-VOS task, we update this task by replacing the training masks M and reference masks M^R with point annotations, and by working with a sparse set of frames. Hence, in a Point-VOS training set, each video $v \in V$ has a set of images $I_v=\{I_1,I_2,...,I_{T_v}\}$ and a set of point annotations $P_v=\{P_{t,o}|t\in T_v^{sparse},o\in\{1,...,O_v\}\}$ with $T_v^{sparse}\subset\{1,...,T_v\}$, $|T_v^{sparse}| \ll T_v$, where each $P_{t,o}$ is a set of points for object o in frame t. At test-time, a Point-VOS method is initialized with reference points $P^R=\{P_1^R,P_2^R,...,P_{O^R}^R\}$ on O^R objects in the reference frame. The expected output $\hat{M}_v$ is the same as for the original VOS task, *i.e.*, a predicted segmentation mask for each frame of each object.

3

3.1. Simulating Point Annotations

To study the effect of training and initializing with points rather than masks, we perform a series of experiments where we train jointly on the DAVIS and YT-VOS training sets and evaluate on the DAVIS validation set.

First, we analyze the number of points required for training supervision and for test-time initialization. For this, we sample points randomly from each of the available ground-truth masks in every annotated video frame such that these points are at least $d = 20$ pixels apart from each other. When the required number of points under the distance constraint cannot be sampled, *e.g.*, when the ground-truth masks are very small, we retain the maximum possible number of points under the constraint.

We then use an STCN [13] version that is adapted to work with points (see Sec. 4), and train multiple models with varying numbers of points used for training supervision. We evaluate each model on the DAVIS validation set with different numbers of points for initialization on the reference frame. Fig. 2 shows that training with 10 points (—) achieves good results and adding more points during training only gives minor gains. When we have less than 10 points during training, the performance strongly degrades. For inference, we find that more points on the reference frame lead to better results. However, this increases the test-time annotation burden and is a trade-off which is highly application dependent. As a result, we propose to evaluate different numbers of points (up to 10) on the reference frame (♦). We also tried other sampling strategies such as farthest-point sampling, but random sampling gave better results (see supplement for details).

Next, we analyze the number of frames required for training on randomly sampled points 10 per frame per object. Starting from all frames, we sub-sample (evenly-spaced) up to 20 frames for each video. Fig. 3 shows the results of STCN trained on such a temporally sparse training set. We find that the performance of STCN saturates at 10 frames (—), where increasing it further does not yield any noticeable performance improvements, and having less than 10 frames deteriorates the performance. Here again, we try different frame sampling techniques, such as random sampling, but do not observe any performance difference (details in the supplement). In summary, we find 10 points per object on 10 frames to be a good setting (★).

3.2. Semi-Automatic Annotation Scheme

We design a very efficient semi-automatic point annotation pipeline to annotate videos with points (Fig. 4). Instead of annotating points in multiple frames manually from scratch, we aim to generate point candidates automatically that then only need to be quickly verified by human annotators.

Pseudo-mask Generation. To annotate an object, as a

Figure 3. **STCN results on DAVIS validation set for varying temporal sparsity, when trained on 10 randomly sampled points per frame per object.** ★ represents our chosen setting, *i.e.* 10 points for training supervision and 10 points for test-time supervision, on 10 frames. We run each experiment 3 times and report the mean score.

starting point we require only a rough localization of it (*e.g.*, by a few points) in a single frame of the video. We then convert this rough localization into a pseudo-mask using the interactive segmentation method DynaMITe [43]. called DynaMITe [43].

Pseudo-mask Propagation. We feed the pseudo-mask from the previous step into STCN [13] to propagate it both forward and backward in time to all other frames of the video and obtain pixel-wise binary probability maps $\mathcal{P} = \{p_1, p_2, ..., p_T\}$ for all frames $1, \ldots, T$ of the video.

Point Sampling. Because both DynaMITe and STCN can introduce errors in the previous process, we do not use the resulting pseudo-masks directly, but instead, we use the STCN output probability maps to sample points and let human annotators verify them. We sub-sample the probability maps temporally equally-spaced to 10 frames. Then, for each object and each retained frame, we threshold the probability map into (i) a high probability region that likely represents the foreground object, (ii) a low probability region that likely represents the background, and (iii) an uncertain region. Points from the uncertain region are hard for STCN and hence might provide a valuable learning signal after being manually annotated. For each of the 10 frames, we then randomly sample 10 potential background points from the low-probability region, 7 potential foreground points from the high-probability region, and 3 ambiguous points from the uncertain region. Here, we again ensure that each of these points are at least d distance apart from each other, and we obtain in total up to 200 points per object.

Annotator Verification. We show the annotators the rough

Figure 4. **Semi-automatic annotation pipeline used to annotate VidLN data.** We first extract a mouse trace segment for each noun in VidLN captions, and convert it into a pseudo mask using DynaMITe. We then use STCN to propagate the pseudo-mask across the video. We then use the STCN output probability maps to sample sparse point annotations and let annotators verify them. Green circles represent foreground points and red circles background points.

localization information used to generate the points, overlaid on the image, so that they understand which object should be considered. Next, we show them the foreground point candidates one by one overlaid on the image (see Fig. 4, right). They use a hotkey to either accept or reject this foreground point candidate or to indicate it is ambiguous. We repeat the same procedure with the background point candidates, and finally with the points with high uncertainty. By batching points of the same type (*e.g.*, foreground candidates) together, the annotators can very quickly verify them.

3.3. Point-VOS Datasets

For our annotations, we choose two large datasets from Video Localized Narratives (VidLN [60]). VidLN provides annotations, where annotators speak to provide a caption for the video, and while speaking, they move their mouse pointer over the object they refer to in multiple key-frames to provide a rough localization. Leveraging VidLN annotations has primarily two advantages for us: (i) the mouse traces can be used to automatically select foreground objects in a video, and correspondingly give us a free rough localization as starting point for our annotation scheme; and (ii) the associated text description can be used to develop multi-modal VOS algorithms. We build on the "location-output question" annotations from Oops [19] because they provide a set of mouse traces for nouns that are already verified to have good quality. Additionally, we choose Kinetics [23], because it is by far the largest VidLN dataset.

To convert the continuous mouse traces into segments, we first use the NLP toolkit spaCy to find nouns in the VidLN captions, and then for each noun retrieve a rough localization by mouse trace segments $\mathcal{T} = \{t_1, t_2, ..., t_n\}$ on key-frames $F = \{f_1, f_2, ..., f_n\}$ provided by VidLN [60]. We extract the mouse trace segment t_k on the key frame f_k on which it has the largest area. Each noun is thus localized with a corresponding mouse trace segment on a single key-frame.

Instance Segmentation from Mouse Traces. We adapt DynaMITe to work with mouse traces instead of user clicks and hence our version of DynaMITe takes a mouse trace segment t_k and the corresponding frame f_k as input, and

generates a binary segmentation mask m_k as output. More details can be found in the supplement.

Point Verification. On average, we get 147 points per object to be verified and, following this procedure, an annotator spends on average 140 seconds per object, *i.e.*, 0.95 seconds per point. In contrast, annotating a single dense mask can take $\sim 80s$ [30]. If we consider annotating an object with a mask in each frame of a single video in the DAVIS training set with an average of 70 frames, it takes about 5600s, which is 40 times slower than our annotation scheme. This demonstrates that our annotation scheme achieves an extreme speedup and lets us annotate much larger datasets than existing VOS benchmarks. Moreover, [18] report that annotating point correspondences over 250 frames for 10 objects with 3 points per object takes 3.3 hours, *i.e.*, 1,188s per object. This is 8 times slower than our annotation scheme, which shows that our point annotations are also much faster than annotating point correspondences.

The statistics of our point annotations in Tab. 1 show that in total we annotated more than $19M$ points for $133K$ objects in $32K$ videos. Thus, we annotated significantly more videos and objects than the largest existing VOS datasets VISOR [15] and BURST [3]. Our annotations cover 4 times more videos than VISOR, consisting of $7.8K$ videos, and 8 times more objects than BURST with $16K$ objects.

PV-Oops. Oops [19] consists of fail videos of unintentional action, often filmed by amateurs in diverse environments. They contain a lot of camera jitter and motion blur, making tracking and segmentation challenging. We create the Point-VOS Oops dataset (PV-Oops), and annotate more than $13K$ objects in $8.4K$ videos that are split into a training and a validation set. We also create a PV-Oops benchmark to measure Point-VOS performance on this challenging domain. For 991 objects in the validation set, we annotate points in the first frame for initialization and dense pixel masks for up to 3 frames per object for evaluation. By conducting simulation experiments on DAVIS, we found that the results evaluated against either masks in only 3 frames, or in all frames, correlate extremely well, meaning that 3 frames are sufficient for evaluation purposes (see supplement for details).

5

Figure 5. **Example point annotations for PV-Oops (top) and PV-Kinetics (bottom).** The objects are connected to nouns from a large vocabulary. Green dots represent foreground points and red dots background points.

PV-Kinetics. Kinetics [23] is a very large-scale action recognition dataset with $650K$ videos that cover 700 action classes. The Kinetics videos are approximately 10s long and are annotated with action labels. Similar to Oops, for the Point-VOS Kinetics (PV-Kinetics) dataset, we use the subset of videos with VidLN annotations and annotate $120K$ objects with points across $23.9K$ videos.

With our new annotations, we obtain the largest VOS-related dataset in terms of the number of videos that cover a wide range of human actions. Fig. 5 shows some example point annotations from PV-Oops and PV-Kinetics. More detailed statistics and more annotation visualizations are available in the supplement.

4. Experiments

4.1. Point-VOS Benchmark

We propose a new benchmark for the Point-VOS task in order to evaluate what a method can achieve by using point annotations for training and testing. At test time, for each foreground object, we provide multiple sets of point initializations with varying degrees of sparsity (1, 2, 5, or 10 points) on the corresponding reference frame, and also report their mean scores. This reflects different trade-offs at test-time between annotation effort and result quality.

For training, we use point annotations from our annotated PV-Oops and PV-Kinetics datasets, in addition to Point-VOS versions PV-DAVIS and PV-YT of the popular VOS training sets, that we create by sub-sampling the object masks both spatially and temporally, as explained in Sec. 3.1. The methods are then evaluated on the validation sets of PV-DAVIS, PV-YT, and PV-Oops. For PV-DAVIS and PV-Oops, we use the popular $\mathcal{J}\&\mathcal{F}$ metric, and report the mean score for the different point initilizations. On the PV-YT validation set, consistent with the original task, we report the $\mathcal{J}$ and $\mathcal{F}$ scores for both seen and unseen classes, along with their overall mean $\mathcal{G}$. Owing to the limited evaluations permitted by the YT-VOS evaluation server, we only consider initialization with 10 points.

We compute all scores on dense ground truth masks.

Point-STCN Baseline. As a first baseline, we adapt STCN to work with points both for training supervision and test-time initialization, and we call this adaptation Point-STCN. Point-STCN makes minimal changes to the original STCN model, showing that existing VOS methods can be easily adapted to work with our Point-VOS datasets.

Pre-training	FT	Mean	1-point	2-point	5-point	10-point
PV-Oops	✗	**59.8**	**48.6**	**57.8**	**65.5**	67.7
PV-Kinetics	✗	50.4	29.5	41.5	60.7	69.7
PV-Oops + PV-Kinetics	✗	54.2	35.2	45.9	63.6	**71.9**
—	✓	61.3	49.4	60.8	67.2	67.7
PV-Oops	✓	62.8	**50.6**	**62.4**	67.7	70.4
PV-Kinetics	✓	62.8	48.0	61.7	**69.6**	72.0
PV-Oops + PV-Kinetics	✓	**63.1**	48.4	61.4	69.5	**72.9**

Table 3. **PV-DAVIS benchmark results ($\mathcal{J}\&\mathcal{F}$) of Point-STCN.** FT: fine-tuning on PV-DAVIS and PV-YT.

Pre-training	FT	$\mathcal{G}$	$\mathcal{J}_{\mathcal{S}}$	$\mathcal{F}_{\mathcal{S}}$	$\mathcal{J}_{\mathcal{U}}$	$\mathcal{F}_{\mathcal{U}}$
PV-Oops	✗	51.6	**60.8**	**62.0**	40.1	43.5
PV-Kinetics	✗	49.6	48.2	50.6	46.3	53.4
PV-Oops + PV-Kinetics	✗	**52.2**	52.4	54.2	**47.7**	**54.5**
—	✓	51.9	59.2	60.5	41.7	46.5
PV-Oops	✓	54.4	61.1	62.6	44.6	49.5
PV-Kinetics	✓	56.6	61.5	64.4	46.9	53.6
PV-Oops+ PV-Kinetics	✓	**57.2**	**62.5**	**64.7**	**47.7**	53.7

Table 4. **PV-YT benchmark results of Point-STCN** when initialized with 10-points. FT: fine-tuning on PV-DAVIS and PV-YT.

The original STCN method is first pre-trained on synthetic video sequences created by augmenting static images, and then fine-tuned on the densely labelled DAVIS and YT-VOS video datasets. Additionally, STCN also uses a syn-

Training	Mean	1-point	2-point	5-point	10-point
PV-DAVIS + PV-YT	48.6	40.5	47.4	52.7	53.8
PV-Kinetics	42.5	27.3	35.7	50.3	56.7
PV-Oops	**61.2**	**54.6**	**60.2**	**64.4**	**65.5**

Table 5. **PV-Oops benchmark results ($\mathcal{J}\&\mathcal{F}$) of Point-STCN.**

thetic dataset called BL30K [12], which we do not use in our work. For Point-STCN, we also start from static image pre-training, and then directly train on our spatially-temporally sparse Point-VOS data. Here, we start by training on PV-DAVIS and PV-YT, and then further explore the benefits of adding our new PV-Oops and PV-Kinetics data as an additional pre-training step. More details about the implementation of Point-STCN are in the supplement.

Tabs. 3 and 4 demonstrate that the newly annotated PV-Oops and PV-Kinetics data bring large improvements as compared to starting from static images, especially for the settings where we fine-tune these models on PV-YT and PV-DAVIS. *E.g.*, the mean $\mathcal{J}\&\mathcal{F}$ on PV-DAVIS improves from 61.3% to 63.1%, and $\mathcal{G}$ on PV-YT improves from 51.9% to 57.2% when pre-training with both PV-Oops and PV-Kinetics. This demonstrates that our new PV-Oops and PV-Kinetics annotations serve as good initilizations for adapting models to other target domains. For results without fine-tuning, we observe that training on PV-Kinetics improves the scores only when sufficiently many points (> 5) are available as test-time initialization. This could be attributed to the domain mismatch between Kinetics and YT-VOS/DAVIS, hence requiring more test-time information.

Tab. 5 shows the results on PV-Oops. Using our PV-Oops annotations improve the performance from 48.6% to 61.2% $\mathcal{J}\&\mathcal{F}$ as compared to Point-STCN trained on just PV-DAVIS and PV-YT. This shows that using points, VOS models can be adapted to target domains with a strongly reduced annotation effort.

Pseudo-mask Baseline. As an alternative to training on points directly, we consider using the points to first generate pseudo-masks and then train STCN on those pseudo-masks. For this, we use the same training procedure that was used to train original STCN, but replace the bootstrapped cross-entropy loss with a more robust *Huberised cross-entropy loss* [34, 59] to reduce the influence of errors in the pseudo-masks. At test-time, we first generate pseudo-masks from the different point initializations and then use these masks as reference. The pseudo-masks provide much more useful information than points, but require an additional model at test-time and increase the run-time.

We generally use DynaMITe [43] to generate pseudo-masks from the point annotations. For the training set, we feed both positive and negative points for each object in every annotated frame to DynaMITe, while for the validation set, we only use the available foreground point ini-

Pre-training	FT	Mean	1-point	2-point	5-point	10-point
—	✗	65.6	55.2	67.4	69.5	70.4
PV-Oops	✗	67.2	59.0	69.8	69.1	70.9
PV-Kinetics	✗	68.9	59.9	71.1	71.6	73.0
PV-Oops + PV-Kinetics	✗	**70.4**	**61.1**	**72.5**	**73.2**	**74.8**
—	✓	70.3	61.6	72.0	72.7	75.0
PV-Oops	✓	70.6	61.0	72.4	73.1	75.8
PV-Kinetics	✓	71.0	62.4	72.9	73.6	75.0
PV-Oops + PV-Kinetics	✓	**71.6**	**63.0**	**73.4**	74.4	75.8
PV-Oops + PV-Kinetics *	✓	67.4	44.8	69.1	**77.0**	**78.8**

Table 6. **PV-DAVIS benchmark results ($\mathcal{J}\&\mathcal{F}$) of STCN [13] trained with pseudo-masks.** FT: fine-tuning on PV-DAVIS and PV-YT, *: using SAM pseudo-masks.

Pre-training	FT	$\mathcal{G}$	$\mathcal{J}_\mathcal{S}$	$\mathcal{F}_\mathcal{S}$	$\mathcal{J}_\mathcal{U}$	$\mathcal{F}_\mathcal{U}$
—	✗	63.0	63.8	66.0	58.2	63.7
PV-Oops	✗	63.6	67.9	69.9	55.5	61.0
PV-Kinetics	✗	67.3	67.8	70.5	62.0	**69.0**
PV-Oops + PV-Kinetics	✗	**68.3**	**70.0**	**72.7**	**62.1**	68.5
—	✓	67.7	69.4	72.7	60.9	67.8
PV-Oops	✓	67.7	69.5	73.0	60.3	68.0
PV-Kinetics	✓	68.1	70.6	73.7	60.4	67.6
PV-Oops + PV-Kinetics	✓	68.7	70.5	73.7	61.8	68.8
PV-Oops + PV-Kinetics *	✓	**73.7**	**75.5**	**77.6**	**68.1**	**73.9**

Table 7. **PV-YT benchmark results of STCN [13] trained with pseudo-masks, and evaluated on 10-points.** FT: fine-tuning on PV-DAVIS and PV-YT, *: using SAM pseudo-masks.

Training	Mean	1-point	2-point	5-point	10-point
—	57.1	50.7	56.7	60.4	60.8
PV-DAVIS+PV-YT	61.3	55.9	61.2	63.9	64.3
PV-Kinetics	61.3	55.4	60.9	64.2	64.7
PV-Oops	**64.9**	**59.7**	**64.9**	**67.4**	**67.7**
PV-Oops*	57.7	40.5	57.6	65.9	66.7

Table 8. **PV-Oops benchmark results ($\mathcal{J}\&\mathcal{F}$) of STCN [13] trained with pseudo-masks, starting from static-image pre-training.** *: using SAM pseudo-masks.

tialization. Recently, the very strong SAM [27] segmentation model became available, so for some additional setups we create masks with SAM with ViT-H backbone instead of DynaMITe and also report those results.

Similar to the previous baseline setup, we first train STCN on pseudo-masks generated from PV-DAVIS and PV-YT, and then later include the additional data from PV-Oops and PV-Kinetics as an additional pre-training step. Tabs. 6 and 7 show that the performance of STCN consistently improves with additional Point-VOS data on both PV-DAVIS and PV-YT. Without fine-tuning, the addi-

	Task	Training	Testing	$\mathcal{J}\&\mathcal{F}$
①	VOS	Masks	Masks	85.3
②	Point-VOS	Points†	Points	67.7
③	Point-VOS*	Pseudo-Masks†	Pseudo-Masks	76.9
④	Hybrid	Masks	Points	78.3
⑤	Hybrid*	Masks	Pseudo-Masks	80.4

Table 9. **Ablations on DAVIS using different training and test-time supervisions.** *: SAM pseudo-masks, †: temporally sparse. Hybrid: masks during training, points at test-time.

tional Point-VOS data improves the mean $\mathcal{J}\&\mathcal{F}$ for PV-DAVIS from 65.6% to 70.4% as compared to just using the static image pre-training, showing that the additional video annotations are very helpful. Likewise, the results on PV-YT improve the $\mathcal{G}$ from 63.0% to 68.3%. Also after fine-tuning, consistent improvements can be seen on both datasets. On DAVIS, using SAM pseudo-masks instead of DynaMITe is beneficial when more points are available at test-time, but leads to significantly worse results for initialization with only 1 or 2 points. This is likely because SAM tends to generate part-based segmentations for a low number of points.

Tab. 8 again shows that fine-tuning STCN on the PV-Oops training data significantly improves the results on the PV-Oops benchmark from 57.1% mean $\mathcal{J}\&\mathcal{F}$ to 64.9%, as compared to just using static images, further boosting the baseline performance.

4.2. Ablation Study

In Tab. 9, we compare the conventional VOS task with Point-VOS on the DAVIS validation set. The training and testing columns denote the annotations available at training and test-time, respectively for each task setup.

In the original VOS setup (①), STCN achieves 85.3% $\mathcal{J}\&\mathcal{F}$. The Point-VOS setup (②) just on points yields 67.7% $\mathcal{J}\&\mathcal{F}$, which is around 80% of the original VOS quality (①). However, when we use pseudo-masks (③), the gap closes more and we achieve 76.9% $\mathcal{J}\&\mathcal{F}$, which is more than 90% of the original VOS quality (①). This result is remarkable, considering that ③ has a three-fold disadvantage compared to ①: 1) weak point supervision instead of masks during training, 2) temporal sparsity during training, 3) sparse point initialization instead of masks at test-time.

We also consider a *Hybrid* task setting, which uses dense mask annotations for training but point-based initialization for testing (see supplement for implementation details). Results on the Hybrid setup show that switching from VOS to point-based training has a larger negative effect on quality than switching to point-based initialization (① → ④ *vs.* ④ → ②). Again, the use of pseudo-masks improves results (⑤) and shrinks the gap towards the original VOS setup.

Pre-Training	UVO-FT	OVIS-VNG		UVO-VNG
		No-FT	FT	
Static	✗	28.5	32.4	39.6
Static + PV-Oops	✗	24.5	31.4	32.9
Static + PV-Kinetics	✗	**33.9**	**35.1**	**51.8**
Static	✓	32.0	32.7	46.4
Static + PV-Oops	✓	31.8	32.6	48.0
Static + PV-Kinetics	✓	32.0	**35.0**	**52.8**
Static + PV-Kinetics *	✓	**32.9**	34.4	52.5

Table 10. **OVIS-VNG and UVO-VNG results ($\mathcal{J}\&\mathcal{F}$) of RF-VNG.** All models start with COCO-PNG pre-training. `UVO-FT`: fine-tuning on UVO-VNG data, `FT`: fine-tuning on OVIS-VNG data, `No-FT`: no fine-tuning. *: using SAM pseudo-masks.

4.3. Language-Guided Tasks

As described in Sec. 3.2, each object that we annotated is linked to a noun in a sentence. Hence, those multi-modal annotations can be used to improve models connecting vision and language, *e.g.*, for the Video Narrative Grounding (VNG) [60] task. In VNG, the input to a method is a text description (*e.g.*, "A green parrot with a red-black neckline is playing with the other parrot" [60]) in which the position of certain nouns (*e.g.*, both instances of "parrot") is marked. For each marked noun, the VNG method has to segment the corresponding noun in each frame of the video.

Our multi-modal point annotations link each point to a noun in a sentence, which matches the setup of VNG. We combine the pseudo-masks generated by DynaMITe based on the annotated points with the language annotations to create our new Oops-VNG and Kinetics-VNG training sets that cover 133K objects in 32K videos. This is more than 3 times larger than the existing VNG datasets OVIS-VNG [41, 60] and UVO-VNG [60, 63] that span 45K objects in 8K videos.

We conduct experiments using the state-of-the-art VNG method RF-VNG [60]. The original RF-VNG is trained in 3 steps: 1) pre-training on static images of the COCO-PNG [20] training set, 2) optional training on UVO-VNG, 3) optional fine-tuning on the OVIS-VNG training set for evaluation on the OVIS-VNG validation set. We use the same training recipe, but insert another pre-training step between steps 1) and 2), where we train RF-VNG [60] on our new Oops-VNG or our new Kinetics-VNG training sets.

Tab. 10 demonstrates that adding our new data significantly improves the baseline results for the VNG task. *E.g.*, the best result on OVIS-VNG improves from 32.7% $\mathcal{J}\&\mathcal{F}$ to 35.0%, and for UVO-VNG, from 46.4% to 52.8%, *i.e.*, we achieve an improvement of 6.4 percentage points.

5. Conclusion

In this work, we have proposed a point-based VOS task, Point-VOS, and a point-wise annotation scheme, which is much more efficient than the existing dense-mask annotation scheme. We use this to annotate two large-scale multimodal VOS datasets that are much larger than the largest available densely annotated VOS datasets. In addition, we also introduce a point-based training strategy for the VOS methods and correspondingly show that existing VOS methods can be easily adapted to leverage our point annotations. Finally, our experiments show the benefits of our newly annotated point data by advancing the state-of-the-art performance for various uni-modal and multi-modal (vision+language) benchmarks.

References

[1] Ali Athar, Jonathon Luiten, Alexander Hermans, Deva Ramanan, and Bastian Leibe. Hodor: High-level object descriptors for object re-segmentation in video learned from static images. In *CVPR*, 2022. 1, 2, 3

[2] Ali Athar, Alexander Hermans, Jonathon Luiten, Deva Ramanan, and Bastian Leibe. TarViS: A Unified Architecture for Target-based Video Segmentation. In *CVPR*, 2023. 2

[3] Ali Athar, Jonathon Luiten, Paul Voigtlaender, Tarasha Khurana, Achal Dave, Bastian Leibe, and Deva Ramanan. BURST: A Benchmark for Unifying Object Recognition, Segmentation and Tracking in Video. In *WACV*, 2023. 1, 2, 3, 5

[4] 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. 12

[5] Rodrigo Benenson and Vittorio Ferrari. From colouring-in to pointillism: revisiting semantic segmentation supervision. *arXiv preprint arXiv:2210.14142*, 2022. 2, 3

[6] Thomas Brox and Jitendra Malik. Object Segmentation by Long Term Analysis of Point Trajectories. In *ECCV*, 2010. 1

[7] 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. 2, 12

[8] Sergi Caelles, Alberto Montes, Kevis-Kokitsi Maninis, Yuhua Chen, Luc Van Gool, Federico Perazzi, and Jordi Pont-Tuset. The 2018 DAVIS Challenge on Video Object Segmentation. *arXiv:1803.00557*, 2018. 1

[9] Xi Chen, Zhiyan Zhao, Yilei Zhang, Manni Duan, Donglian Qi, and Hengshuang Zhao. Focalclick: Towards practical interactive image segmentation. In *CVPR*, 2022. 3

[10] Bowen Cheng, Omkar Parkhi, and Alexander Kirillov. Pointly-supervised instance segmentation. In *CVPR*, 2022. 2, 3

[11] Ho Kei Cheng and Alexander G. Schwing. XMem: Long-Term Video Object Segmentation with an Atkinson-Shiffrin Memory Model. In *ECCV*, 2022. 1, 2

[12] Ho Kei Cheng, Yu-Wing Tai, and Chi-Keung Tang. Modular interactive video object segmentation: Interaction-to-mask, propagation and difference-aware fusion. In *CVPR*, 2021. 7

[13] 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. 1, 2, 4, 7, 12, 16, 20, 21, 22

[14] 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. 12

[15] Ahmad Darkhalil, Dandan Shan, Bin Zhu, Jian Ma, Amlan Kar, Richard Higgins, Sanja Fidler, David Fouhey, and Dima Damen. EPIC-KITCHENS VISOR Benchmark: VIdeo Segmentations and Object Relations. In *NeurIPS Track on Datasets and Benchmarks*, 2022. 1, 2, 5

[16] Henghui Ding, Chang Liu, Shuting He, Xudong Jiang, and Chen Change Loy. MeViS: A Large-scale Benchmark for Video Segmentation with Motion Expressions. In *ICCV*, 2023. 3

[17] Henghui Ding, Chang Liu, Shuting He, Xudong Jiang, Philip HS Torr, and Song Bai. MOSE: A new dataset for video object segmentation in complex scenes. In *ICCV*, 2023. 1, 2

[18] Carl Doersch, Ankush Gupta, Larisa Markeeva, Adrià Recasens, Lucas Smaira, Yusuf Aytar, João Carreira, Andrew Zisserman, and Yi Yang. Tap-vid: A benchmark for tracking any point in a video. In *NeurIPS*, 2022. 3, 5

[19] Dave Epstein, Boyuan Chen, and Carl Vondrick. Oops! Predicting Unintentional Action in Video. In *CVPR*, 2020. 2, 5

[20] Cristina González, Nicolás Ayobi, Isabela Hernández, José Hernández, Jordi Pont-Tuset, and Pablo Arbeláez. Panoptic narrative grounding. In *CVPR*, 2021. 8

[21] Lingyi Hong, Wenchao Chen, Zhongying Liu, Wei Zhang, Pinxue Guo, Zhaoyu Chen, and Wenqiang Zhang. LVOS: A Benchmark for Long-term Video Object Segmentation. *arXiv preprint arXiv:2211.10181*, 2022. 1, 2

[22] 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. 12

[23] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 2, 5, 6

[24] A. Khoreva, R. Benenson, E. Ilg, T. Brox, and B. Schiele. Lucid data dreaming for object tracking. In *CVPRW*, 2017. 3

[25] Anna Khoreva, Anna Rohrbach, and Bernt Schiele. Video object segmentation with language referring expressions. In *ACCV*, 2019. 1, 3, 12

[26] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 15

[27] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer White-head, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *ICCV*, 2023. 3, 7, 21, 22

[28] Issam H. Laradji, Negar Rostamzadeh, Pedro O. Pinheiro, David Vázquez, and Mark Schmidt. Where are the Blobs: Counting by Localization with Point Supervision. In *ECCV*, 2018. 3

[29] Fanchao Lin, Hongtao Xie, Yan Li, and Yongdong Zhang. Query-Memory Re-Aggregation for Weakly-supervised Video Object Segmentation. In *AAAI*, 2021. 3

[30] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 5

[31] Jonathon Luiten, Paul Voigtlaender, and B. Leibe. PReMVOS: Proposal-generation, Refinement and Merging for Video Object Segmentation. In *ACCV*, 2018. 2, 12

[32] Sabarinath Mahadevan, Paul Voigtlaender, and Bastian Leibe. Iteratively Trained Interactive Segmentation. In *BMVC*, 2018. 3

[33] 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. 12

[34] Aditya Krishna Menon, Ankit Singh Rawat, Sashank J. Reddi, and Sanjiv Kumar. Can gradient clipping mitigate label noise? In *ICLR*, 2020. 7

[35] Pascal Mettes and Cees G. M. Snoek. Pointly-Supervised Action Localization. *IJCV*, 2018. 3

[36] Pascal Mettes, Jan C. van Gemert, and Cees G. M. Snoek. Spot On: Action Localization from Pointly-Supervised Proposals. In *ECCV*, 2016. 3

[37] Seoung Wug Oh, Joon-Young Lee, Kalyan Sunkavalli, and Seon Joo Kim. Fast Video Object Segmentation by Reference-Guided Mask Propagation. In *CVPR*, 2018. 2, 12

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

[39] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation. In *CVPR*, 2016. 1, 2, 3

[40] 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, 2, 12

[41] Jiyang Qi, Yan Gao, Yao Hu, Xinggang Wang, Xiaoyu Liu, Xiang Bai, Serge Belongie, Alan Yuille, Philip Torr, and Song Bai. Occluded Video Instance Segmentation: A Benchmark. *IJCV*, 2022. 2, 8

[42] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1

[43] Amit Rana, Sabarinath Mahadevan, Alexander Hermans, and Bastian Leibe. DynaMITe: Dynamic Query Bootstrapping for Multi-object Interactive Segmentation Transformer. In *ICCV*, 2023. 3, 4, 7, 14, 20

[44] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022. 1

[45] Seonguk Seo, Joon-Young Lee, and Bohyung Han. Urvos: Unified referring video object segmentation network with a large-scale benchmark. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16*, pages 208–223. Springer, 2020. 3

[46] Hongje Seong, Junhyuk Hyun, and Euntai Kim. Kernelized memory network for video object segmentation. In *ECCV*, 2020. 2

[47] Konstantin Sofiiuk, Ilia Petrov, Olga Barinova, and Anton Konushin. f-brs: Rethinking backpropagating refinement for interactive segmentation. In *CVPR*, 2020. 3

[48] Konstantin Sofiiuk, Ilia Petrov, and Anton Konushin. Reviving Iterative Training with Mask Guidance for Interactive Segmentation. *arXiv preprint arXiv:2102.06583*, 2021. 3

[49] Mingjie Sun, Jimin Xiao, Eng Gee Lim, Yanchun Xie, and Jiashi Feng. Adaptive ROI generation for video object segmentation using reinforcement learning. *PR*, 2020. 2

[50] Mingjie Sun, Jimin Xiao, Eng Gee Lim, Bingfeng Zhang, and Yao Zhao. Fast template matching and update for video object tracking and segmentation. In *CVPR*, 2020. 3

[51] P. Sundberg, T. Brox, M. Maire, P. Arbelaez, and J. Malik. Occlusion boundary detection and figure/ground assignment from optical flow. In *CVPR*, 2011. 1

[52] Pavel Tokmakov, Jie Li, and Adrien Gaidon. Breaking the "Object" in Video Object Segmentation. In *CVPR*, 2023. 1, 2

[53] David Tsai, Matthew Flagg, and James M. Rehg. Motion Coherent Tracking with Multi-label MRF optimization. In *BMVC*, 2010. 1

[54] 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. 12

[55] Paul Voigtlaender and Bastian Leibe. Online Adaptation of Convolutional Neural Networks for the 2017 DAVIS Challenge on Video Object Segmentation. In *CVPRW*, 2017. 2, 12

[56] Paul Voigtlaender and Bastian Leibe. Online adaptation of convolutional neural networks for video object segmentation. In *BMVC*, 2017. 2

10

[57] 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. 2, 12

[58] Paul Voigtlaender, Jonathon Luiten, Philip HS Torr, and Bastian Leibe. Siam r-cnn: Visual tracking by re-detection. In *CVPR*, 2020. 3, 12

[59] Paul Voigtlaender, Lishu Luo, Chun Yuan, Yong Jiang, and Bastian Leibe. Reducing the Annotation Effort for Video Object Segmentation Datasets. In *WACV*, 2021. 1, 2, 3, 7

[60] Paul Voigtlaender, Soravit Changpinyo, Jordi Pont-Tuset, Radu Soricut, and Vittorio Ferrari. Connecting Vision and Language with Video Localized Narratives. In *CVPR*, 2023. 1, 2, 5, 8, 13, 16

[61] Carl Vondrick, Abhinav Shrivastava, Alireza Fathi, Sergio Guadarrama, and Kevin Murphy. Tracking emerges by colorizing videos. In *ECCV*, 2018. 1

[62] Qiang Wang, Li Zhang, Luca Bertinetto, Weiming Hu, and Philip HS Torr. Fast online object tracking and segmentation: A unifying approach. In *CVPR*, 2019. 3, 12

[63] Weiyao Wang, Matt Feiszli, Heng Wang, and Du Tran. Unidentified video objects: A benchmark for dense, open-world segmentation. In *ICCV*, 2021. 8

[64] Xiaolong Wang, Allan Jabri, and Alexei A. Efros. Learning Correspondence from the Cycle-Consistency of Time. In *CVPR*, 2019. 1

[65] Ziqin Wang, Jun Xu, Li Liu, Fan Zhu, and Ling Shao. RANet: Ranking Attention Network for Fast Video Object Segmentation. In *ICCV*, 2019. 12

[66] 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, 2, 12

[67] Shuangjie Xu, Daizong Liu, Linchao Bao, Wei Liu, and Pan Zhou. MHP-VOS: Multiple Hypotheses Propagation for Video Object Segmentation. In *CVPR*, 2019. 12

[68] L. Yang, Yanran Wang, Xuehan Xiong, Jianchao Yang, and Aggelos K. Katsaggelos. Efficient Video Object Segmentation via Network Modulation. In *CVPR*, 2018. 12

[69] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *ICCV*, 2019. 2

[70] Zongxin Yang, Yunchao Wei, and Yi Yang. Associating Objects with Transformers for Video Object Segmentation. In *NeurIPS*, 2021. 1, 2

[71] Bin Zhao, Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Generating masks from boxes by mining spatio-temporal consistencies in videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13556–13566, 2021. 3

[72] Yang Zheng, Adam W. Harley, Bokui Shen, Gordon Wetzstein, and Leonidas J. Guibas. Pointodyssey: A large-scale synthetic dataset for long-term point tracking. In *ICCV*, 2023. 3

Point-VOS: Supplementary Material

Abstract

In this supplementary, we provide the experimental results of the additional simulations in Appendix A, the details for annotating the Point-VOS Oops validation set in Appendix B, the statistics for Point-VOS datasets in Appendix C, the implementation details in Appendix D and the additional qualitative results in Appendix E.

A. Additional Simulations

Farthest Point Sampling Strategy. In Sec. 3.1 of the main paper, we ran a number of point simulation experiments on DAVIS [40] and YT-VOS [66] 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. 6, we show the results for the FPS point sampling strategy on the DAVIS validation set. It can be seen that the performance 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 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. 7. 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.

Figure 6. **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 7. **Results on the DAVIS validation set for randomly sub-sampling frames.** We vary the number of randomly sampled frames.

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 [4, 7, 13, 14, 22, 25, 31, 33, 37, 38, 54, 55, 57, 58, 62, 65, 67, 68] from the DAVIS benchmark leaderboard, evaluate them on a sparse set of frames. Finally, we compare these results

Figure 8. **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 the results on the temporally dense validation set, *i.e.*, the original DAVIS validation set with all frames. As seen in Fig. 8, 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).

B. 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 exper-

iment for evaluating on a sparse subset of frames (see Appendix A), 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 [60] 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 11. **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.

C. Point-VOS Datasets Statistics

Overview. In Tab. 11, 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.

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. 9 shows the frame distribution for PV-Oops (see Fig. 9a) and PV-Kinetics (see Fig. 9b). 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. 10 for PV-Oops (see Fig. 10a) and PV-Kinetics (see Fig. 10b). As seen, we observe similar point distributions in both PV-Oops and PV-Kinetics. As the size of the objects varies,

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

Figure 9. **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.

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.

D. 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 [43] 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

14

(a) PV-Oops

(b) PV-Kinetics

Figure 10. **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.

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.

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 [26] 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 [60].

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.

E. Additional Qualitative Results

In Fig. 11 and Fig. 12, 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. 13 and Fig. 14, 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. 15, Fig. 17, and Fig. 19, 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. 16, Fig. 18 and Fig. 20, we demonstrate the results of STCN [13] (trained with pseudo-masks) on PV-Oops, PV-DAVIS and PV-YT, respectively.

Figure 11. **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 12. **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 13. **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 14. **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 15. **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 16. **Tracking results of STCN [13] on PV-Oops.** The model is trained on PV-Oops with DynaMITe [43] pseudo-masks, then evaluated on 10 points setup.

Figure 17. **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 18. **Tracking results of STCN [13] 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 [27].

Figure 19. **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 20. **Tracking results of STCN [13] 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 [27].