Tracking Anything with Decoupled Video Segmentation

Ho Kei Cheng[1†] Seoung Wug Oh[2] Brian Price[2] Alexander Schwing[1] Joon-Young Lee[2]
[1]University of Illinois Urbana-Champaign [2]Adobe Research
{hokeikc2,aschwing}@illinois.edu, {seoh,bprice,jolee}@adobe.com

Figure 1. Visualization of our semi-online video segmentation results. Top: our algorithm (**DEVA**) extends Segment Anything (SAM) [30] to video for open-world video segmentation with no user input required. Bottom: DEVA performs text-prompted video segmentation for novel objects (with prompt *"beyblade"*, a type of spinning-top toy) by integrating Grounding-DINO [38] and SAM [30].

Abstract

*Training data for video segmentation are expensive to annotate. This impedes extensions of end-to-end algorithms to new video segmentation tasks, especially in large-vocabulary settings. To 'track anything' without training on video data for every individual task, we develop a **de**coupled **v**ideo segmen**ta**tion approach (**DEVA**), composed of task-specific image-level segmentation and class/task-agnostic bi-directional temporal propagation. Due to this design, we only need an image-level model for the target task (which is cheaper to train) and a universal temporal propagation model which is trained once and generalizes across tasks. To effectively combine these two modules, we use bi-directional propagation for (semi-)online fusion of segmentation hypotheses from different frames to generate a coherent segmentation. We show that this decoupled formulation compares favorably to end-to-end approaches in several data-scarce tasks including large-vocabulary video panoptic segmentation, open-world video segmentation, referring video segmentation, and unsupervised video object segmentation. Code is available at:* hkchengrex.github.io/Tracking-Anything-with-DEVA.

1. Introduction

Video segmentation aims to segment and associate objects in a video. It is a fundamental task in computer vision and is crucial for many video understanding applications.

Most existing video segmentation approaches train end-to-end video-level networks on annotated video datasets. They have made significant strides on common benchmarks like YouTube-VIS [69] and Cityscape-VPS [27]. However,

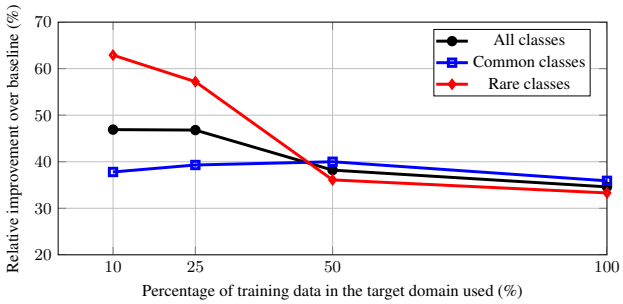

Figure 2. We plot relative $\overline{\text{VPQ}}$ increase of our decoupled approach over the end-to-end baseline when we vary the training data in the target domain (VIPSeg [45]). Common/rare classes are the top/bottom 50% most annotated object category in the training set. Our improvement is most significant (>60%) in rare classes when there is a small amount of training data. This is because our decoupling allows the use of external class-agnostic temporal propagation data – data that cannot be used by existing end-to-end baselines. Details in Section 4.5.1.

these datasets have small vocabularies: YouTube-VIS contains 40 object categories, and Cityscape-VPS only has 19. It is questionable whether recent end-to-end paradigms are scalable to large-vocabulary, or even open-world video data. A recent larger vocabulary (124 classes) video segmentation dataset, VIPSeg [45], has been shown to be more difficult – using the same backbone, a recent method [34] achieves only 26.1 VPQ compared with 57.8 VPQ on Cityscape-VPS. To the best of our knowledge, recent video segmentation methods [2, 39] developed for the open-world setting (e.g., BURST [2]) are not end-to-end and are based on tracking of per-frame segmentation – further highlighting the difficulty of end-to-end training on large-vocabulary datasets. As the number of classes and scenarios in the dataset increases, it becomes more challenging to train and develop end-to-end video models to jointly solve segmentation and association, especially if annotations are scarce.

In this work, we aim to reduce reliance on the amount of target training data by leveraging external data *outside of the target domain*. For this, we propose to study *decoupled video segmentation*, which combines task-specific image-level segmentation and task-agnostic temporal propagation. Due to this design, we only need an image-level model for the target task (which is cheaper) and a universal temporal propagation model which is trained once and generalizes across tasks. Universal promptable image segmentation models like 'segment anything' (SAM) [30] and others [76, 32, 24, 73, 74] have recently become available and serve as excellent candidates for the image-level model in a 'track anything' pipeline – Figure 1 shows some promising results of our integration with these methods.

Researchers have studied decoupled formulations before, as 'tracking-by-detection' [26, 58, 3]. However, these approaches often consider image-level detections immutable, while the temporal model only associates detected objects. This formulation depends heavily on the quality of per-image detections and is sensitive to image-level errors.

In contrast, we develop a (semi-)online bi-directional propagation algorithm to 1) denoise image-level segmentation with in-clip consensus (Section 3.2.1), and 2) combine results from temporal propagation and in-clip consensus gracefully (Section 3.2.2). This bi-directional propagation allows temporally more coherent and potentially better results than those of an image-level model (see Figure 2).

We do not aim to replace end-to-end video approaches. Indeed, we emphasize that specialized frameworks on video tasks with sufficient video-level training data (e.g., YouTubeVIS [69]) outperform the developed method. Instead, we show that our decoupled approach acts as a strong baseline when an image model is available but video data is scarce. This is in spirit similar to pretraining of large language models [52]: a *task-agnostic* understanding of natural language is available before being finetuned on specific tasks – in our case, we learn propagation of segmentations of *class-agnostic* objects in videos via a temporal propagation module and make technical strides in applying this knowledge to specific tasks. The proposed decoupled approach transfers well to large-scale or open-world datasets, and achieves state-of-the-art results in large-scale video panoptic segmentation (VIPSeg [45]) and open-world video segmentation (BURST [2]). It also performs competitively on referring video segmentation (Ref-YouTubeVOS [55], Ref-DAVIS [25]) and unsupervised video object segmentation (DAVIS-16/17[5]) without end-to-end training.

To summarize:

- We propose using decoupled video segmentation that leverages external data, which allows it to generalize better to target tasks with limited annotations than end-to-end video approaches and allows us to seamlessly incorporate existing universal image segmentation models like SAM [30].
- We develop bi-directional propagation that denoises image segmentations and merges image segmentations with temporally propagated segmentations gracefully.
- We empirically show that our approach achieves favorable results in several important tasks including large-scale video panoptic segmentation, open-world video segmentation, referring video segmentation, and unsupervised video object segmentation.

2. Related Works

End-to-End Video Segmentation. Recent end-to-end video segmentation approaches [50, 23, 62, 4, 6, 14, 13] have made significant progress in tasks like Video Instance Segmentation (VIS) and Video Panoptic Segmentation (VPS), especially in closed and small vocabulary datasets like YouTube-VIS [69] and Cityscape-VPS [27].

However, these methods require end-to-end training and their scalability to larger vocabularies, where video data and annotations are expensive, is questionable. MaskProp [4] uses mask propagation to provide temporal information, but still needs to be trained end-to-end on the target task. This is because their mask propagation is not class-agnostic. We circumvent this training requirement and instead decouple the task into image segmentation and temporal propagation, each of which is easier to train with image-only data and readily available class-agnostic mask propagation data respectively.

Open-World Video Segmentation. Recently, an open-world video segmentation dataset BURST [2] has been proposed. It contains 482 object classes in diverse scenarios and evaluates open-world performance by computing metrics for the common classes (78, overlap with COCO [37]) and uncommon classes (404) separately. The baseline in BURST [2] predicts a set of object proposals using an image instance segmentation model trained on COCO [37] and associates the proposals frame-by-frame using either box IoU or STCN [11]. OWTB [39] additionally associates proposals using optical flow and pre-trained Re-ID features. Differently, we use bi-directional propagation that generates segmentations instead of simply associating existing segmentations – this reduces sensitivity to image segmentation errors. UVO [18] is another open-world video segmentation dataset and focuses on human actions. We mainly evaluate on BURST [2] as it is much more diverse and allows separate evaluation for common/uncommon classes.

Decoupled Video Segmentation. 'Tracking-by-detection' approaches [26, 58, 3] often consider image-level detections immutable and use a short-term temporal tracking model to associate detected objects. This formulation depends heavily on the quality of per-image detections and is sensitive to image-level errors. Related long-term temporal propagation works exist [20, 19], but they consider a single task and do not filter the image-level segmentation. We instead propose a general framework, with a bi-directional propagation mechanism that denoises the image segmentations and allows our result to potentially perform better than the image-level model.

Video Object Segmentation. Semi-supervised Video Object Segmentation (VOS) aims to propagate an initial ground-truth segmentation through a video [47, 46, 70, 9]. However, it does not account for any errors in the initial segmentation, and cannot incorporate new segmentation given by the image model at later frames. SAM-PT [53] combines point tracking with SAM [12] to create a video object segmentation pipeline, while our method tracks masks directly. We find a recent VOS algorithm [9] works well for our temporal propagation model. Our proposed bi-directional propagation is essential for bringing image segmentation models and propagation models together as a unified video segmentation framework.

Unified Video Segmentation. Recent Video-K-Net [34] uses a unified framework for multiple video tasks but requires separate end-to-end training for each task. Unicorn [66], TarViS [1], and UNINEXT [67] share model parameters for different tasks, and train on all the target tasks end-to-end. They report lower tracking accuracy for objects that are not in the target tasks during training compared with class-agnostic VOS approaches, which might be caused by joint learning with class-specific features. In contrast, we only train an image segmentation model for the target task, while the temporal propagation model is always fully class-agnostic for generalization across tasks.

Segmenting/Tracking Anything. Concurrent to our work, Segment Anything (SAM) [30] demonstrates the effectiveness and generalizability of large-scale training for universal image segmentation, serving as an important foundation for open-world segmentation. Follow-up works [68, 12] extend SAM to video data by propagating the masks generated by SAM with video object segmentation algorithms. However, they rely on single-frame segmentation and lack the denoising capability of our proposed in-clip consensus approach.

3. Decoupled Video Segmentation

3.1. Formulation

Decoupled Video Segmentation. Our decoupled video segmentation approach is driven by an image segmentation model and a universal temporal propagation model. The image model, trained specifically on the target task, provides task-specific image-level segmentation hypotheses. The temporal propagation model, trained on class-agnostic mask propagation datasets, associates and propagates these hypotheses to segment the whole video. This design separates the learning of task-specific segmentation and the learning of general video object segmentation, leading to a robust framework even when data in the target domain is scarce and insufficient for end-to-end learning.

Notation. Using t as the time index, we refer to the corresponding frame and its final segmentation as I_t and $\mathbf{M}_t$ respectively. In this paper, we represent a segmentation as a set of non-overlapping per-object binary segments, *i.e.*, $\mathbf{M}_t = \{m_i, 0 < i \leq |\mathbf{M}_t|\}$, where $m_i \cap m_j = \emptyset$ if $i \neq j$.

The image segmentation model $\text{Seg}(I)$ takes an image I as input and outputs a segmentation. We denote its output segmentation at time t as $\text{Seg}(I_t) = \text{Seg}_t = \{s_i, 0 < i \leq |\text{Seg}_t|\}$, which is also a set of non-overlapping binary segments. This segmentation model can be swapped for different target tasks, and users can be in the loop to correct the segmentation as we do not limit its internal architecture.

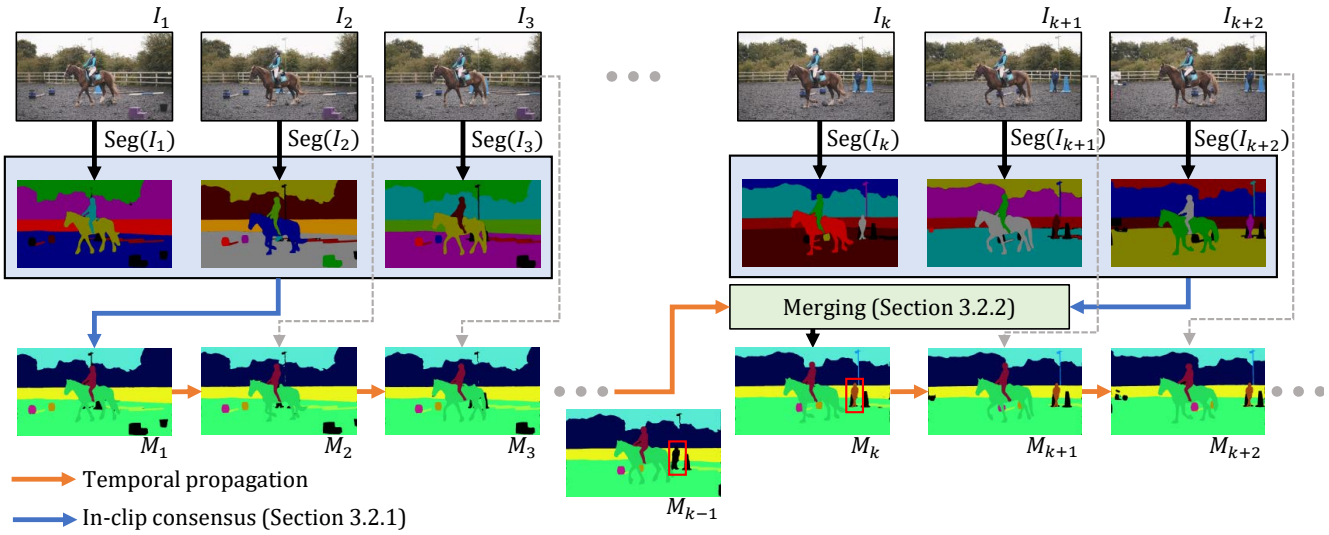

Figure 3. Overview of our framework. We first filter image-level segmentations with in-clip consensus (Section 3.2.1) and temporally propagate this result forward. To incorporate a new image segmentation at a later time step (for previously unseen objects, e.g., red box), we merge the propagated results with in-clip consensus as described in Section 3.2.2. Specifics of temporal propagation are in the appendix.

The temporal propagation model $\text{Prop}(\mathbf{H}, I)$ takes a collection of segmented frames (memory) $\mathbf{H}$ and a query image I as input and segments the query frame with the objects in the memory. For instance, $\text{Prop}(\{I_1, \mathbf{M}_1\}, I_2)$ propagates the segmentation $\mathbf{M}_1$ from the first frame I_1 to the second frame I_2. Unless mentioned explicitly, the memory $\mathbf{H}$ contains all past segmented frames.

Overview. Figure 3 illustrates the overall pipeline. At a high level, we aim to propagate segmentations discovered by the image segmentation model to the full video with temporal propagation. We mainly focus on the (semi-)online setting. Starting from the first frame, we use the image segmentation model for initialization. To denoise errors from single-frame segmentation, we look at a small clip of a few frames in the near future (in the online setting, we only look at the current frame) and reach an in-clip consensus (Section 3.2.1) as the output segmentation. Afterward, we use the temporal propagation model to propagate the segmentation to subsequent frames. We modify an off-the-shelf state-of-the-art video object segmentation XMem [9] as our temporal propagation model, with details given in the appendix. The propagation model itself cannot segment new objects that appear in the scene. Therefore, we periodically incorporate new image segmentation results using the same in-clip consensus as before and merge the consensus with the propagated result (Section 3.2.2). This pipeline combines the strong temporal consistency from the propagation model (past) and the new semantics from the image segmentation model (future), hence the name *bi-directional propagation*. Next, we will discuss the bi-directional propagation pipeline in detail.

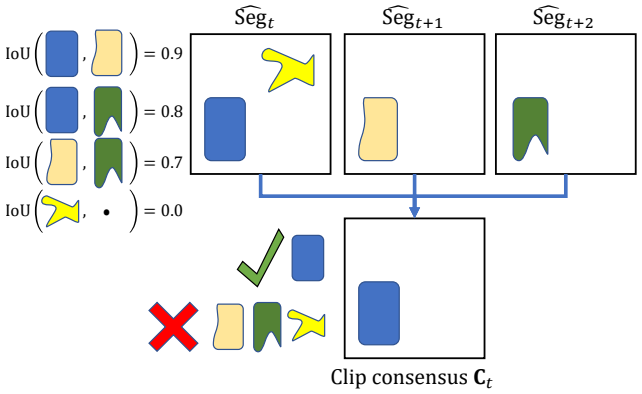

Figure 4. A simple illustration of in-clip consensus. The top three squares represent object proposals from three different frames aligned to time t. The blue shape is the most supported by other object proposals and is selected as output. The yellow shape is not supported by any and is ruled out as noise. The remaining are not used due to significant overlap with the selected (blue) shape.

3.2. Bi-Directional Propagation

3.2.1 In-clip Consensus

Formulation. In-clip consensus operates on the image segmentations of a small future clip of n frames (Seg_t, Seg_{t+1}, ..., Seg_{t+n-1}) and outputs a denoised consensus $\mathbf{C}_t$ for the current frame. In the online setting, $n = 1$ and $\mathbf{C}_t = \text{Seg}_t$. In the subsequent discussion, we focus on the semi-online setting, as consensus computation in the online setting is straightforward. As an overview, we first obtain a set of *object proposals* on the target frame t via spatial alignment, merge the object proposals into a combined rep-

resentation in a second step, and optimize for an indicator variable to choose a subset of proposals as the output in an integer program. Figure 4 illustrates this in-clip consensus computation in a stylized way and we provide details regarding each of the three aforementioned steps (spatial alignment, representation, and integer programming) next.

Spatial Alignment. As the segmentations (Seg_t, Seg_{t+1}, ..., Seg_{t+n-1}) correspond to different time steps, they might be spatially misaligned. This misalignment complicates the computation of correspondences between segments. To align segmentations Seg_{t+i} with frame t, techniques like optical flow warping are applicable. In this paper, we simply re-use the temporal propagation model to find the aligned segmentation $\widehat{\text{Seg}}_{t+i}$ (note $\widehat{\text{Seg}}_t = \text{Seg}_t$) via

$$\widehat{\text{Seg}}_{t+i} = \text{Prop}\left(\{I_{t+i}, \text{Seg}_{t+i}\}, I_t\right), 0 < i < n. \quad (1)$$

Note, the propagation model here only uses one frame as memory at a time and this temporary memory $\{I_{t+i}, \text{Seg}_{t+i}\}$ is discarded immediately after alignment. It does not interact with the global memory $\mathbf{H}$.

Representation. Recall that we represent a segmentation as a set of non-overlapping per-object binary segments. After aligning all the segmentations to frame t, each segment is an *object proposal* for frame I_t. We refer to the union of all these proposals via $\mathbf{P}$ (time index omitted for clarity):

$$\mathbf{P} = \bigcup_{i=0}^{n-1} \widehat{\text{Seg}}_{t+i} = \{p_i, 0 < i \leq |\mathbf{P}|\}. \quad (2)$$

The output of consensus voting is represented by an indicator variable $v^* \in \{0,1\}^{|\mathbf{P}|}$ that combines segments into the consensus output $\mathbf{C}_t$:

$$\mathbf{C}_t = \{p_i | v_i^* = 1\} = \{c_i, 0 < i \leq |\mathbf{C}|\}. \quad (3)$$

We resolve overlapping segments c_i in $\mathbf{C}_t$ by prioritizing smaller segments as they are more vulnerable to being majorly displaced by overlaps. This priority is implemented by sequentially rendering the segments c_i on an image in descending order of area. We optimize for v based on two simple criteria:

1. Lone proposals p_i are likely to be noise and should not be selected. Selected proposals should be supported by other (unselected) proposals.
2. Selected proposals should not overlap significantly with each other.

We combine these criteria in an integer programming problem which we describe next.

Integer Programming. We aim to optimize the indicator variable v to achieve the above two objectives, by addressing the following integer programming problem:

$$v^* = \text{argmax}_v \sum_i \left(\text{Supp}_i + \text{Penal}_i\right) \text{ s.t.} \sum_{i,j} \text{Overlap}_{ij} = 0. \quad (4)$$

Next, we discuss each of the terms in the program in detail.

First, we define the pairwise Intersection-over-Union (IoU) between the i-th proposal and the j-th proposal as:

$$\text{IoU}_{ij} = \text{IoU}_{ji} = \frac{|p_i \cap p_j|}{|p_i \cup p_j|}, 0 \leq \text{IoU}_{ij} \leq 1. \quad (5)$$

The i-th proposal *supports* the j-th proposal if $\text{IoU}_{ij} > 0.5$ – the higher the IoU, the stronger the support. The more support a segment has, the more favorable it is to be selected. To maximize the total support of selected segments, we maximize the below objective for all i:

$$\text{Supp}_i = v_i \sum_j \begin{cases} \text{IoU}_{ij}, & \text{if IoU}_{ij} > 0.5 \text{ and } i \neq j \\ 0, & \text{otherwise} \end{cases}. \quad (6)$$

Additionally, proposals that support each other should not be selected together as they significantly overlap. This is achieved by constraining the following term to zero:

$$\text{Overlap}_{ij} = \begin{cases} v_i v_j, & \text{if IoU}_{ij} > 0.5 \text{ and } i \neq j \\ 0, & \text{otherwise} \end{cases}. \quad (7)$$

Lastly, we introduce a penalty for selecting any segment for 1) tie-breaking when a segment has no support, and 2) excluding noisy segments, with weight α:

$$\text{Penal}_i = -\alpha v_i. \quad (8)$$

We set the tie-breaking weight $\alpha = 0.5$. For all but the first frame, we merge $\mathbf{C}_t$ with the propagated segmentation $\text{Prop}(\mathbf{H}, I_t)$ into the final output $\mathbf{M}_t$ as described next.

3.2.2 Merging Propagation and Consensus

Formulation. Here, we seek to merge the propagated segmentation $\text{Prop}(\mathbf{H}, I_t) = \mathbf{R}_t = \{r_i, 0 < i \leq |\mathbf{R}|\}$ (from the past) with the consensus $\mathbf{C}_t = \{c_j, 0 < j \leq |\mathbf{C}|\}$ (from the near future) into a single segmentation $\mathbf{M}_t$. We associate segments from these two segmentations and denote the association with an indicator a_{ij} which is 1 if r_i associates with c_j, and 0 otherwise. Different from the in-clip consensus, these two segmentations contain fundamentally different information. Thus, we do not eliminate any segments and instead fuse all pairs of associated segments while letting the unassociated segments pass through to the output. Formally, we obtain the final segmentation via

$$\mathbf{M}_t = \{r_i \cup c_j | a_{ij} = 1\} \cup \{r_i | \forall_j a_{ij} = 0\} \cup \{c_j | \forall_i a_{ij} = 0\}, \quad (9)$$

where overlapping segments are resolved by prioritizing the smaller segments as discussed in Section 3.2.1.

Maximizing Association IoU. We find a_{ij} by maximizing the pairwise IoU of all associated pairs, with a minimum association IoU of 0.5. This is equivalent to a maximum bipartite matching problem, with r_i and c_j as vertices and edge weight e_{ij} given by

$$e_{ij} = \begin{cases} \text{IoU}(r_i, c_j), & \text{if IoU}(r_i, c_j) > 0.5 \\ -1, & \text{otherwise} \end{cases}. \quad (10)$$

Requiring any matched pairs from two non-overlapping segmentations to have IoU > 0.5 leads to a unique matching, as shown in [29]. Therefore, a greedy solution of setting $a_{ij} = 1$ if $e_{ij} > 0$ and 0 otherwise suffices to obtain an optimal result.

Segment Deletion. As an implementation detail, we delete inactive segments from the memory to reduce computational costs. We consider a segment r_i inactive when it fails to associate with any segments c_j from the consensus for consecutive L times. Such objects might have gone out of view or were a misdetection. Concretely, we associate a counter cnt_i with each propagated segment r_i, initialized as 0. When r_i is not associated with any segments c_j from the consensus, i.e., $\forall_j a_{ij} = 0$, we increment cnt_i by 1 and reset cnt_i to 0 otherwise. When cnt_i reaches the pre-defined threshold L, the segment r_i is deleted from the memory. We set $L = 5$ in all our experiments.

4. Experiments

We first present our main results using a large-scale video panoptic segmentation dataset (VIPSeg [45]) and an open-world video segmentation dataset (BRUST [2]). Next, we show that our method also works well for referring video object segmentation and unsupervised video object segmentation. We present additional results on the smaller-scale YouTubeVIS dataset in the appendix, but unsurprisingly recent end-to-end specialized approaches perform better because a sufficient amount of data is available in this case. Figure 1 visualizes some results of the integration of our approach with universal image segmentation models like SAM [30] or Grounding-Segment-Anything [38, 30]. By default, we merge in-clip consensus with temporal propagation every 5 frames with a clip size of $n = 3$ in the semi-online setting, and $n = 1$ in the online setting. We evaluate all our results using either official evaluation codebases or official servers. We use image models trained with standard training data for each task (using open-sourced models whenever available) and a universal temporal propagation module for all tasks unless otherwise specified.

The temporal propagation model is based on XMem [9], and is trained in a class-agnostic fashion with image segmentation datasets [56, 60, 72, 33, 8] and video object segmentation datasets [65, 47, 48]. With the long-term memory of XMem [9], our model can handle long videos with ease.

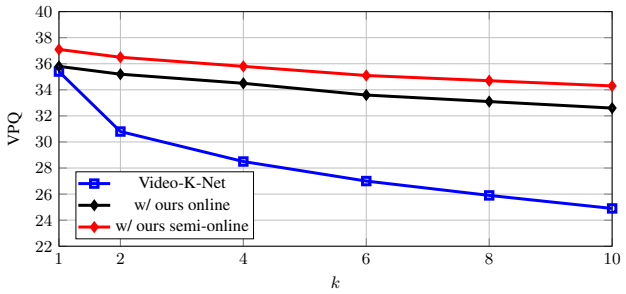

Figure 5. Performance trend comparison of Video-K-Net [34] and our decoupled approach with the same base model. Ours decreases slower with larger k, indicating that the proposed decoupled method has a better long-term propagation.

We use top-k filtering [10] with $k = 30$ following [9]. The performance of our modified propagation model on common video object segmentation benchmarks (DAVIS [47], YouTubeVOS [65], and MOSE [16]) are listed in the appendix.

4.1. Large-Scale Video Panoptic Segmentation

We are interested in addressing the large vocabulary setting. To our best knowledge, VIPSeg [45] is currently the largest scale in-the-wild panoptic segmentation dataset, with 58 things classes and 66 stuff classes in 3,536 videos of 232 different scenes.

Metrics. To evaluate the quality of the result, we adopt the commonly used VPQ (Video Panoptic Quality) [27] and STQ (Segmentation and Tracking Quality) [63] metrics. VPQ extends image-based PQ (Panoptic Quality) [29] to video data by matching objects in sliding windows of k frames (denoted VPQ^k). When $k = 1$, VPQ = PQ and associations of segments between frames are ignored. Correct long-range associations, which are crucial for object tracking and video editing tasks, are only evaluated with a large value of k. For a more complete evaluation of VPS, we evaluate $k \in \{1, 2, 4, 6, 8, 10, \infty\}$. Note, VPQ^∞ considers the entire video as a tube and requires global association. We additionally report $\overline{\text{VPQ}}$, which is the average of VPQ^∞ and the arithmetic mean of $\text{VPQ}^{\{1,2,4,6,8,10\}}$. This weights VPQ^∞ higher as it represents video-level performance, while the other metrics only assess frame-level or clip-level results. STQ is proposed in STEP [63] and is the geometric mean of AQ (Association Quality) and SQ (Segmentation Quality). It evaluates pixel-level associations and semantic segmentation quality respectively. We refer readers to [27] and [63] for more details on VPQ and STQ.

Main Results. Table 1 summarizes our findings. To assess generality, we study three models as image segmentation input (PanoFCN [35], Mask2Former [7], and Video-K-Net [34]) to our decoupled approach. The weights of these image models are initialized by pre-training on the COCO panoptic dataset [37] and subsequently fine-tuned

Backbone				VPQ^1	VPQ^2	VPQ^4	VPQ^6	VPQ^8	VPQ^{10}	VPQ^∞	$\overline{VPQ}$	STQ
Clip-PanoFCN		end-to-end [45]	semi-online	27.3	26.0	24.2	22.9	22.1	21.5	18.1	21.1	28.3
Clip-PanoFCN		decoupled (ours)	online	29.5	28.9	28.1	27.2	26.7	26.1	25.0	26.4	35.7
Clip-PanoFCN		decoupled (ours)	semi-online	**31.3**	**30.8**	**30.1**	**29.4**	**28.8**	**28.3**	**27.1**	**28.4**	**35.8**
Video-K-Net	R50	end-to-end [34]	online	35.4	30.8	28.5	27.0	25.9	24.9	21.7	25.2	33.7
Video-K-Net	R50	decoupled (ours)	online	35.8	35.2	34.5	33.6	33.1	32.6	30.5	32.3	38.4
Video-K-Net	R50	decoupled (ours)	semi-online	37.1	36.5	35.8	35.1	34.7	34.3	32.3	33.9	38.6
Mask2Former	R50	decoupled (ours)	online	41.0	40.2	39.3	38.4	37.9	37.3	33.8	36.4	41.1
Mask2Former	R50	decoupled (ours)	semi-online	**42.1**	**41.5**	**40.8**	**40.1**	**39.7**	**39.3**	**36.1**	**38.3**	**41.5**
Video-K-Net	Swin-B	end-to-end [34]	online	49.8	45.2	42.4	40.5	39.1	37.9	32.6	37.5	45.2
Video-K-Net	Swin-B	decoupled (ours)	online	48.2	47.4	46.5	45.6	45.1	44.5	42.0	44.1	48.6
Video-K-Net	Swin-B	decoupled (ours)	semi-online	50.0	49.3	48.5	47.7	47.3	46.8	44.5	46.4	48.9
Mask2Former	Swin-B	decoupled (ours)	online	55.3	54.6	53.8	52.8	52.3	51.9	49.0	51.2	**52.4**
Mask2Former	Swin-B	decoupled (ours)	semi-online	**56.0**	**55.4**	**54.6**	**53.9**	**53.5**	**53.1**	**50.0**	**52.2**	52.2

Table 1. Comparisons of end-to-end approaches (e.g., state-of-the-art Video-K-Net [34]) with our decoupled approach on the large-scale video panoptic segmentation dataset VIPSeg [45]. Our method scales with better image models and performs especially well with large k where long-term associations are considered. All baselines are reproduced using official codebases.

		Validation			**Test**		
Method		$OWTA_{all}$	$OWTA_{com}$	$OWTA_{unc}$	$OWTA_{all}$	$OWTA_{com}$	$OWTA_{unc}$
Mask2Former	w/ Box tracker [2]	60.9	66.9	24.0	55.9	61.0	24.6
Mask2Former	w/ STCN tracker [2]	64.6	71.0	25.0	57.5	62.9	23.9
OWTB [39]		55.8	59.8	38.8	56.0	59.9	38.3
Mask2Former	w/ ours online	69.5	74.6	42.3	70.1	75.0	44.1
Mask2Former	w/ ours semi-online	**69.9**	**75.2**	41.5	**70.5**	**75.4**	44.1
EntitySeg	w/ ours online	68.8	72.7	49.6	69.5	72.9	53.0
EntitySeg	w/ ours semi-online	69.5	73.3	**50.5**	69.8	73.1	**53.3**

Table 2. Comparison to baselines in the open-world video segmentation dataset BURST [2]. 'com' stands for 'common classes' and 'unc' stands for 'uncommon classes'. Our method performs better in both – in the common classes with Mask2Former [7] image backbone, and in the uncommon classes with EntitySeg [49]. The agility to switch image backbones is one of the main advantages of our decoupled formulation. Baseline performances are transcribed from [2].

on VIPSeg [45]. Our method outperforms both baseline Clip-PanoFCN [45] and state-of-the-art Video-K-Net [34] with the same backbone, especially if k is large, *i.e.*, when long-term associations are more important. Figure 5 shows the performance trend with respect to k. The gains for large values of k highlight the use of a decoupled formulation over end-to-end training: the latter struggles with associations eventually, as training sequences aren't arbitrarily long. Without any changes to our generalized mask propagation module, using a better image backbone (*e.g.*, SwinB [40]) leads to noticeable improvements. Our method can likely be coupled with future advanced methods in image segmentation for even better performance.

4.2. Open-World Video Segmentation

Open-world video segmentation addresses the difficult problem of discovering, segmenting, and tracking objects in the wild. BURST [2] is a recently proposed dataset that evaluates open-world video segmentation. It contains diverse scenarios and 2,414 videos in its validation/test sets. There are a total of 482 object categories, 78 of which are 'common' classes while the rest are 'uncommon'.

Metrics. Following [2], we assess Open World Tracking Accuracy (OWTA), computed separately for 'all', 'common', and 'uncommon' classes. False positive tracks are not directly penalized in the metrics as the ground-truth annotations are not exhaustive for all objects in the scene, but indirectly penalized by requiring the output mask to be mutually exclusive. We refer readers to [2, 42] for details.

Main Results. Table 2 summarizes our findings. We study two image segmentation models: Mask2Former [7], and EntitySeg [49], both of which are pretrained on the COCO [37] dataset. The Mask2Former weight is trained for the instance segmentation task, while EntitySeg is trained

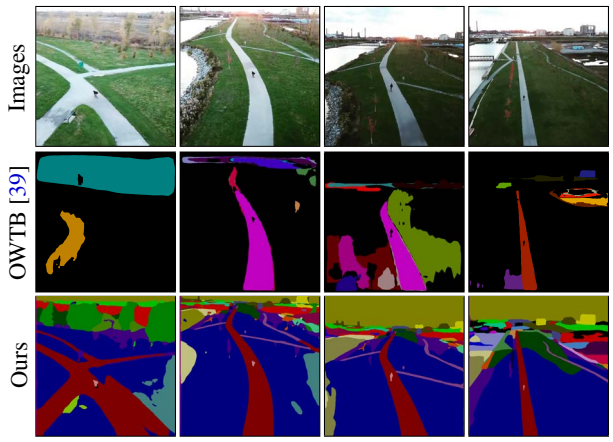

Figure 6. An in-the-wild result in the BURST [2] dataset. Note, we can even track the small skateboarder (pink mask on the road).

for 'entity segmentation', that is to segment all visual entities without predicting class labels. We find EntitySeg works better for novel objects, as it is specifically trained to do so. Being able to plug and play the latest development of open-world image segmentation models without any fine-tuning is one of the major advantages of our formulation.

Our approach outperforms the baselines, which all follow the 'tracking-by-detection' paradigm. In these baselines, segmentations are detected every frame, and a short-term temporal module is used to associate these segmentations between frames. This paradigm is sensitive to misdetections in the image segmentation model. 'Box tracker' uses per-frame object IoU; 'STCN tracker' uses a pretrained STCN [11] mask propagation network; and OWTB [39] uses a combination of IoU, optical flow, and Re-ID features. We also make use of mask propagation, but we go beyond the setting of simply associating existing segmentations – our bi-directional propagation allows us to improve upon the image segmentations and enable long-term tracking. Figure 6 compares our results on one of the videos in BURST to OWTB [39].

4.3. Referring Video Segmentation

Referring video segmentation takes a text description of an object as input and segments the target object. We experiment on Ref-DAVIS17 [25] and Ref-YouTubeVOS [55] which augments existing video object segmentation datasets [47, 65] with language expressions. Following [64], we assess $\mathcal{J}\&\mathcal{F}$ which is the average of Jaccard index ($\mathcal{J}$), and boundary F1-score ($\mathcal{F}$).

Table 3 tabulates our results. We use an image-level ReferFormer [64] as the image segmentation model. We find that the quality of referring segmentation has a high variance across the video (e.g., the target object might be too small at the beginning of the video). As in all competing approaches [55, 64, 17], we opt for an offline setting to reduce this variance. Concretely, we perform the initial in-clip consensus by selecting 10 uniformly spaced frames in the video and using the frame with the highest confidence given by the image model as a 'key frame' for aligning the other frames. We then forward- and backward-propagate from the key frame without incorporating additional image segmentations. We give more details in the appendix. Our method outperforms other approaches.

Method	Ref-DAVIS [25]	Ref-YTVOS [55]
URVOS [55]	51.6	47.2
ReferFormer [64]	60.5	62.4
VLT [17]	61.6	63.8
Ours	**66.3**	**66.0**

Table 3. $\mathcal{J}\&\mathcal{F}$ comparisons on two referring video segmentation datasets. Ref-YTVOS stands for Ref-YouTubeVOS [55].

4.4. Unsupervised Video Object Segmentation

Unsupervised video object segmentation aims to find and segment salient target object(s) in a video. We evaluate on DAVIS-16 [47] (single-object) and DAVIS-17 [5] (multi-object). In the single-object setting, we use the image saliency model DIS [51] as the image model and employ an offline setting as in Section 4.3. In the multi-object setting, since the image saliency model only segments one object, we instead use EntitySeg [49] and follow our semi-online protocol on open-world video segmentation in Section 4.2. Table 4 summarizes our findings. Please refer to the appendix for details.

Method	D16-val	D17-val	D17-td
RTNet [54]	85.2	-	-
PMN [31]	85.9	-	-
UnOVOST [43]	-	67.9	58.0
Propose-Reduce [36]	-	70.4	-
Ours	**88.9**	**73.4**	**62.1**

Table 4. $\mathcal{J}\&\mathcal{F}$ comparisons on three unsupervised video object segmentation datasets: DAVIS16 validation (D16-val), DAVIS17 validation (D17-val), and DAVIS17 test-dev (D17-td). Missing entries mean that the method did not report results on that dataset.

4.5. Ablation Studies

4.5.1 Varying Training Data

Here, we vary the amount of training data in the target domain (VIPSeg [45]) to measure the sensitivity of end-to-end approaches *vs*. our decoupled approach. We subsample different percentages of videos from the training set

Varying clip size	VPQ^1	VPQ^{10}	$\overline{\text{VPQ}}$	STQ	FPS
$n = 1$	41.0	37.3	36.4	41.1	**10.3**
$n = 2$	40.4	37.2	36.3	39.0	9.8
$n = 3$	**42.1**	**39.3**	38.3	41.5	7.8
$n = 4$	**42.1**	39.1	**38.5**	42.3	6.6
$n = 5$	41.7	38.9	38.3	**42.8**	5.6
Varying merge freq.	VPQ^1	VPQ^{10}	$\overline{\text{VPQ}}$	STQ	FPS
Every 3 frames	**42.2**	39.2	**38.4**	**42.6**	5.2
Every 5 frames	42.1	**39.3**	38.3	41.5	7.8
Every 7 frames	41.5	39.0	35.7	40.5	**8.4**
Spatial Align?	VPQ^1	VPQ^{10}	$\overline{\text{VPQ}}$	STQ	FPS
Yes	**42.1**	**39.3**	**38.3**	**41.5**	7.8
No	36.7	33.9	32.8	33.7	**9.2**

Table 5. Performances of our method on VIPSeg [45] with different hyperparameters and design choices. By default, we use a clip size of $n = 3$ and a merge frequency of every 5 frames with spatial alignment for a balance between performance and speed.

to train Video-K-Net-R50 [34] (all networks are still pretrained with COCO-panoptic [37]). We then compare end-to-end performances with our (semi-online) decoupled performances (the temporal propagation model is unchanged as it does not use any data from the target domain). Figure 1 plots our findings – our model has a much higher relative $\overline{\text{VPQ}}$ improvement over the baseline Video-K-Net for rare classes if little training data is available.

4.5.2 In-Clip Consensus

Here we explore hyperparameters and design choices in in-clip consensus. Table 5 tabulates our performances with different *clip sizes*, different *frequencies* of merging in-clip consensus with temporal propagation, and whether to use *spatial alignment* during in-clip consensus. Mask2Former-R50 is used as the backbone in all entries. For clip size $n = 2$, tie-breaking is ambiguous. A large clip is more computationally demanding and potentially leads to inaccurate spatial alignment as the appearance gap between frames in the clip increases. A high merging frequency reduces the delay between the appearance of a new object and its detection in our framework but requires more computation. By default, we use a clip size $n = 3$, merge consensus with temporal propagation every 5 frames, and enable spatial alignment for a balance between performance and speed.

4.5.3 Using Temporal Propagation

Here, we compare different approaches for using temporal propagation in a decoupled setting. Tracking-by-detection approaches [26, 58, 3] typically detect segmentation at every frame and use temporal propagation to associate these per-frame segmentations. We test these short-term association approaches using 1) mask IoU between adjacent frames, 2) mask IoU of adjacent frames warped by optical flow from RAFT [59], and 3) query association [22] of query-based segmentation [7] between adjacent frames. We additionally compare with variants of our temporal propagation method: 4) 'ShortTrack', where we consider only short-term tracking by re-initializing the memory $\mathbf{H}$ every frame, and 5) 'TrustImageSeg', where we explicitly trust the consensus given by the image segmentations over temporal propagation by discarding segments that are not associated with a segment in the consensus (i.e., dropping the middle term in Eq. (9)). Table 6 tabulates our findings. For all entries, we use Mask2Former-R50 [7] in the online setting on VIPSeg [45] for fair comparisons.

Temporal scheme	VPQ^1	VPQ^4	VPQ^{10}	$\overline{\text{VPQ}}$	STQ
Mask IoU	39.9	32.7	27.7	27.6	34.5
Mask IoU+flow	40.2	33.7	28.8	28.6	37.0
Query assoc.	40.4	33.1	28.1	28.0	35.8
'ShortTrack'	40.6	33.3	28.3	28.2	37.2
'TrustImageSeg'	40.3	37.5	33.7	33.2	37.9
Ours, bi-direction	**41.0**	**39.3**	**37.3**	**36.4**	**41.1**

Table 6. Performances of different temporal schema on VIPSeg [45]. Our bi-directional propagation scheme is necessary for the final high performance.

4.6. Limitations

As the temporal propagation model is task-agnostic, it cannot detect new objects by itself. As shown by the red boxes in Figure 3, the new object in the scene is missing from $\mathbf{M}_{k-1}$ and can only be detected in $\mathbf{M}_k$ – this results in delayed detections relating to the frequency of merging with in-clip consensus. Secondly, we note that end-to-end approaches still work better when training data is sufficient, i.e., in smaller vocabulary settings like YouTubeVIS [69] as shown in the appendix. But we think decoupled methods are more promising in large-vocabulary/open-world settings.

5. Conclusion

We present **DEVA**, a decoupled video segmentation approach for 'tracking anything'. It uses a bi-directional propagation technique that effectively scales image segmentation methods to video data. Our approach critically leverages external task-agnostic data to reduce reliance on the target task, thus generalizing better to tasks with scarce data than end-to-end approaches. Combined with universal image segmentation models, our decoupled paradigm demonstrates state-of-the-art performance as a first step towards open-world large-vocabulary video segmentation.

Acknowledgments. Work supported in part by NSF grants 2008387, 2045586, 2106825, MRI 1725729 (HAL [28]), and NIFA award 2020-67021-32799.

References

[1] Ali Athar, Alexander Hermans, Jonathon Luiten, Deva Ramanan, and Bastian Leibe. Tarvis: A unified approach for target-based video segmentation. *arXiv preprint arXiv:2301.02657*, 2023. 3

[2] 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. 2, 3, 6, 7, 8, 17, 18, 19

[3] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *ICCV*, 2019. 2, 3, 9

[4] Gedas Bertasius and Lorenzo Torresani. Classifying, segmenting, and tracking object instances in video with mask propagation. In *CVPR*, 2020. 2, 3, 19

[5] Sergi Caelles, Jordi Pont-Tuset, Federico Perazzi, Alberto Montes, Kevis-Kokitsi Maninis, and Luc Van Gool. The 2019 davis challenge on vos: Unsupervised multi-object segmentation. In *arXiv preprint arXiv:1905.00737*, 2019. 2, 8, 13, 18

[6] Bowen Cheng, Anwesa Choudhuri andd Ishan Misra, Alexander Kirillov, Rohit Girdhar, and Alexander G. Schwing. Mask2former for video instance segmentation. In *https://arxiv.org/abs/2112.10764*, 2021. 2

[7] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *CVPR*, 2022. 6, 7, 9, 16, 17, 18, 19

[8] Ho Kei Cheng, Jihoon Chung, Yu-Wing Tai, and Chi-Keung Tang. Cascadepsp: Toward class-agnostic and very high-resolution segmentation via global and local refinement. In *CVPR*, 2020. 6, 14

[9] Ho Kei Cheng and Alexander G Schwing. Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *ECCV*, 2022. 3, 4, 6, 13, 14, 15, 16, 17, 18

[10] 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. 6, 14

[11] 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. 3, 8, 13, 15

[12] Yangming Cheng, Liulei Li, Yuanyou Xu, Xiaodi Li, Zongxin Yang, Wenguan Wang, and Yi Yang. Segment and track anything. In *arXiv preprint arXiv:2305.06558*, 2023. 3

[13] A. Choudhuri, G. Chowdhary, and A. G. Schwing. Assignment-Space-Based Multi-Object Tracking and Segmentation. In *ICCV*, 2021. 2

[14] A. Choudhuri, G. Chowdhary, and A. G. Schwing. Context-Aware Relative Object Queries to Unify Video Instance and Panoptic Segmentation. In *CVPR*, 2023. 2

[15] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *NIPS Workshop*, 2014. 13

[16] 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. 6, 14

[17] Henghui Ding, Chang Liu, Suchen Wang, and Xudong Jiang. Vlt: Vision-language transformer and query generation for referring segmentation. In *TPAMI*, 2022. 8, 18

[18] Yuming Du, Wen Guo, Yang Xiao, and Vincent Lepetit. Uvo challenge on video-based open-world segmentation 2021: 1st place solution. *ICCV Workshop*, 2021. 3

[19] Shubhika Garg and Vidit Goel. Mask selection and propagation for unsupervised video object segmentation. In *WACV*, 2021. 3

[20] Vidit Goel, Jiachen Li, Shubhika Garg, Harsh Maheshwari, and Humphrey Shi. Msn: efficient online mask selection network for video instance segmentation. In *CVPR Workshop*, 2021. 3

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 13

[22] De-An Huang, Zhiding Yu, and Anima Anandkumar. Minvis: A minimal video instance segmentation framework without video-based training. In *NeurIPS*, 2022. 9, 19

[23] Sukjun Hwang, Miran Heo, Seoung Wug Oh, and Seon Joo Kim. Video instance segmentation using inter-frame communication transformers. *NeurIPS*, 2021. 2

[24] Lei Ke, Mingqiao Ye, Martin Danelljan, Yifan Liu, Yu-Wing Tai, Chi-Keung Tang, and Fisher Yu. Segment anything in high quality. In *arXiv*, 2023. 2

[25] Anna Khoreva, Anna Rohrbach, and Bernt Schiele. Video object segmentation with language referring expressions. In *ACCV*, 2019. 2, 8, 18, 19

[26] Chanho Kim, Fuxin Li, Arridhana Ciptadi, and James M Rehg. Multiple hypothesis tracking revisited. In *ICCV*, 2015. 2, 3, 9

[27] Dahun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Video panoptic segmentation. In *CVPR*, 2020. 1, 2, 6, 17

[28] Volodymyr Kindratenko, Dawei Mu, Yan Zhan, John Maloney, Sayed Hadi Hashemi, Benjamin Rabe, Ke Xu, Roy Campbell, Jian Peng, and William Gropp. Hal: Computer system for scalable deep learning. In *PEARC*, 2020. 9

[29] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *CVPR*, 2019. 6

[30] 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 *arXiv preprint arXiv:2304.02643*, 2023. 1, 2, 3, 6

[31] Minhyeok Lee, Suhwan Cho, Seunghoon Lee, Chaewon Park, and Sangyoun Lee. Unsupervised video object segmentation via prototype memory network. In *WACV*, 2023. 8

[32] Feng Li, Hao Zhang, Peize Sun, Xueyan Zou, Shilong Liu, Jianwei Yang, Chunyuan Li, Lei Zhang, and Jianfeng Gao. Semantic-sam: Segment and recognize anything at any granularity. In *arXiv*, 2023. 2

[33] Xiang Li, Tianhan Wei, Yau Pun Chen, Yu-Wing Tai, and Chi-Keung Tang. Fss-1000: A 1000-class dataset for few-shot segmentation. In *CVPR*, 2020. 6, 14

[34] Xiangtai Li, Wenwei Zhang, Jiangmiao Pang, Kai Chen, Guangliang Cheng, Yunhai Tong, and Chen Change Loy. Video k-net: A simple, strong, and unified baseline for video segmentation. In *CVPR*, 2022. 2, 3, 6, 7, 9, 17, 19

[35] Yanwei Li, Hengshuang Zhao, Xiaojuan Qi, Liwei Wang, Zeming Li, Jian Sun, and Jiaya Jia. Fully convolutional networks for panoptic segmentation. In *CVPR*, 2021. 6

[36] Huaijia Lin, Ruizheng Wu, Shu Liu, Jiangbo Lu, and Jiaya Jia. Video instance segmentation with a propose-reduce paradigm. In *ICCV*, 2021. 8

[37] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 3, 6, 7, 9, 18

[38] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *arXiv preprint arXiv:2303.05499*, 2023. 1, 6

[39] Yang Liu, Idil Esen Zulfikar, Jonathon Luiten, Achal Dave, Deva Ramanan, Bastian Leibe, Aljoša Ošep, and Laura Leal-Taixé. Opening up open world tracking. In *CVPR*, 2022. 2, 3, 7, 8, 19

[40] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 7

[41] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 14

[42] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixé, and Bastian Leibe. Hota: A higher order metric for evaluating multi-object tracking. *International journal of computer vision*, 129:548–578, 2021. 7

[43] Jonathon Luiten, Idil Esen Zulfikar, and Bastian Leibe. Unovost: Unsupervised offline video object segmentation and tracking. In *WACV*, 2020. 8

[44] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *CVPR*, 2016. 18

[45] Jiaxu Miao, Xiaohan Wang, Yu Wu, Wei Li, Xu Zhang, Yunchao Wei, and Yi Yang. Large-scale video panoptic segmentation in the wild: A benchmark. In *CVPR*, 2022. 2, 6, 7, 8, 9, 16, 17, 19

[46] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *ICCV*, 2019. 3, 13, 14

[47] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016. 3, 6, 8, 14, 15, 17, 18

[48] Jiyang Qi, Yan Gao, Yao Hu, Xinggang Wang, Xiaoyu Liu, Xiang Bai, Serge Belongie, Alan Yuille, Philip HS Torr, and Song Bai. Occluded video instance segmentation. *IJCV*, 2022. 6, 14, 15, 17

[49] Lu Qi, Jason Kuen, Yi Wang, Jiuxiang Gu, Hengshuang Zhao, Zhe Lin, Philip Torr, and Jiaya Jia. Open-world entity segmentation. In *arXiv preprint arXiv:2107.14228*, 2021. 7, 8, 18, 19

[50] Siyuan Qiao, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Vip-deeplab: Learning visual perception with depth-aware video panoptic segmentation. In *CVPR*, 2021. 2

[51] Xuebin Qin, Hang Dai, Xiaobin Hu, Deng-Ping Fan, Ling Shao, and Luc Van Gool. Highly accurate dichotomous image segmentation. In *ECCV*, 2022. 8, 18

[52] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018. 2

[53] Frano Rajič, Lei Ke, Yu-Wing Tai, Chi-Keung Tang, Martin Danelljan, and Fisher Yu. Segment anything meets point tracking. In *arXiv preprint arXiv:2307.01197*, 2023. 3

[54] Sucheng Ren, Wenxi Liu, Yongtuo Liu, Haoxin Chen, Guoqiang Han, and Shengfeng He. Reciprocal transformations for unsupervised video object segmentation. In *CVPR*, 2021. 8

[55] Seonguk Seo, Joon-Young Lee, and Bohyung Han. Urvos: Unified referring video object segmentation network with a large-scale benchmark. In *ECCV*, 2020. 2, 8, 18, 19

[56] Jianping Shi, Qiong Yan, Li Xu, and Jiaya Jia. Hierarchical image saliency detection on extended cssd. In *TPAMI*, 2015. 6, 14

[57] Konstantin Sofiiuk, Ilia Petrov, Olga Barinova, and Anton Konushin. f-brs: Rethinking backpropagating refinement for interactive segmentation. In *CVPR*, 2020. 16

[58] Siyu Tang, Mykhaylo Andriluka, Bjoern Andres, and Bernt Schiele. Multiple people tracking by lifted multicut and person re-identification. In *CVPR*, 2017. 2, 3, 9

[59] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020. 9

[60] Lijun Wang, Huchuan Lu, Yifan Wang, Mengyang Feng, Dong Wang, Baocai Yin, and Xiang Ruan. Learning to detect salient objects with image-level supervision. In *CVPR*, 2017. 6, 14

[61] Weiyao Wang, Matt Feiszli, Heng Wang, and Du Tran. Unidentified video objects: A benchmark for dense, open-world segmentation. In *CVPR*, 2021. 15

[62] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. In *CVPR*, 2021. 2

[63] Mark Weber, Jun Xie, Maxwell Collins, Yukun Zhu, Paul Voigtlaender, Hartwig Adam, Bradley Green, Andreas Geiger, Bastian Leibe, Daniel Cremers, et al. Step: Segmenting and tracking every pixel. In *NeurIPS*, 2021. 6

[64] Jiannan Wu, Yi Jiang, Peize Sun, Zehuan Yuan, and Ping Luo. Language as queries for referring video object segmentation. In *CVPR*, 2022. 8, 18, 19

[65] Ning Xu, Linjie Yang, Yuchen Fan, Dingcheng Yue, Yuchen Liang, Jianchao Yang, and Thomas Huang. Youtube-vos: A

large-scale video object segmentation benchmark. In *ECCV*, 2018. 6, 8, 14, 16, 17

[66] Bin Yan, Yi Jiang, Peize Sun, Dong Wang, Zehuan Yuan, Ping Luo, and Huchuan Lu. Towards grand unification of object tracking. In *ECCV*, 2022. 3

[67] Bin Yan, Yi Jiang, Jiannan Wu, Dong Wang, Ping Luo, Zehuan Yuan, and Huchuan Lu. Universal instance perception as object discovery and retrieval. In *CVPR*, 2023. 3

[68] Jinyu Yang, Mingqi Gao, Zhe Li, Shang Gao, Fangjing Wang, and Feng Zheng. Track anything: Segment anything meets videos. In *arXiv preprint arXiv:2304.11968*, 2023. 3

[69] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *ICCV*, 2019. 1, 2, 9, 13, 18

[70] Zongxin Yang, Yunchao Wei, and Yi Yang. Associating objects with transformers for video object segmentation. In *NeurIPS*, 2021. 3, 14, 15

[71] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In *ECCV*, 2016. 18

[72] Yi Zeng, Pingping Zhang, Jianming Zhang, Zhe Lin, and Huchuan Lu. Towards high-resolution salient object detection. In *ICCV*, 2019. 6, 14

[73] Chaoning Zhang, Dongshen Han, Yu Qiao, Jung Uk Kim, Sung-Ho Bae, Seungkyu Lee, and Choong Seon Hong. Faster segment anything: Towards lightweight sam for mobile applications. In *arXiv*, 2023. 2

[74] Xu Zhao, Wenchao Ding, Yongqi An, Yinglong Du, Tao Yu, Min Li, Ming Tang, and Jinqiao Wang. Fast segment anything. In *arXiv*, 2023. 2

[75] Xueyan Zou, Zi-Yi Dou, Jianwei Yang, Zhe Gan, Linjie Li, Chunyuan Li, Xiyang Dai, Harkirat Behl, Jianfeng Wang, Lu Yuan, et al. Generalized decoding for pixel, image, and language. In *CVPR*, 2023. 19

[76] Xueyan Zou, Jianwei Yang, Hao Zhang, Feng Li, Linjie Li, Jianfeng Gao, and Yong Jae Lee. Segment everything everywhere all at once. In *arXiv*, 2023. 2

This appendix is structured as follows:

- We first provide implementation details of our temporal propagation network (Section A).
- We then analyze the class-agnostic training data of the temporal propagation network (Section B).
- After that, we list additional details regarding our experimental settings and results (Section C).
- Next, we provide results on the small-vocabulary YouTube-VIS [69] dataset for reference (Section D).
- Lastly, we present qualitative results (Section E).

A. Implementation Details of Temporal Propagation

A.1. Overview

Recall that the temporal propagation model $\text{Prop}(\mathbf{H}, I)$ takes a set of segmented frames (memory) $\mathbf{H}$ and a query image I as input, and segments the query frame with the objects in the memory. For instance, $\text{Prop}\left(\{I_1, \mathbf{M}_1\}, I_2\right)$ propagates the segmentation $\mathbf{M}_1$ from the first frame I_1 to the second frame I_2. The memory $\mathbf{H}$ is a compact representation computed from all past segmented frames.

In our implementation, we adopt the design of the internal memory $\mathbf{H}$ from the recent Video Object Segmentation (VOS) approach XMem [9]. VOS algorithms are initialized by a first-frame segmentation (in our case, the first in-clip consensus output), and segment new incoming query frames. XMem is an online algorithm that maintains an internal feature memory representation $\mathbf{H}$. For each incoming frame I, it computes a query representation which is used to read from the feature memory. It then uses the memory readout $\mathbf{F}$ to segment the query frame. The segmentation result $(\text{Prop}(\mathbf{H}, I))$ is used to update the internal representation $\mathbf{H}$. With an internal memory management mechanism [9], this design has a bounded GPU memory cost with respect to the number of processed frames which is suitable for processing long video sequences.

We refer readers to [9] for details regarding XMem. We describe core details below for completeness. We make a few technical modifications to XMem to increase robustness in our generalized setting, which we also document below. We provide the full code at `hkchengrex.github.io/Tracking-Anything-with-DEVA`.

A.2. Network Architecture

The temporal propagation network consists of four network modules: a *key encoder*, a *value encoder*, a *mask decoder*, and a *Convolutional Gated Recurrent Unit (Conv-GRU)* [15].

The *key encoder*, implemented with a ResNet-50 [21], takes an image as input and produces multi-scale features at the first (stride 4), second (stride 8), and third (stride 16) stages. The fourth stage is discarded. The feature in the third stage is projected to a 'key', which is used for querying during memory reading. After segmentation, if we decide to add the segmented query frame into the memory $\mathbf{H}$, we will re-use this 'key' in the memory.

The *value encoder*, implemented with a ResNet-18 [21], takes an image and a corresponding object mask as inputs and produces a 'value' representation as part of the memory. We discard the fourth stage and only use the stride-16 output feature in the third stage. Objects are processed independently (done in mini-batches during inference).

The *mask decoder* takes the memory readout $\mathbf{F}$ and multi-scale skip connections from the key encoder as inputs and produces an object mask. It consists of three upsampling blocks. Each upsampling block uses the output from the previous layer as input, upsamples it bilinearly by a factor of two, and fuses the upsampled result with the skip connection at the corresponding scale with a residual block with two 3×3 convolutions. A 3×3 convolution is used as the last output layer to generate a single-channel (stride 4) logit and bilinearly upsamples it by four times to the original resolution. Similar to the value encoder, objects are processed independently, which can be done in mini-batches during inference. Soft-aggregation [46] is used to combine logits for different objects as in [9].

The *Convolutional Gated Recurrent Unit (Conv-GRU)* [15] takes the last hidden state and the output of every upsampling block in the mask decoder as input and produces an updated hidden state. 3×3 convolutions are used as projections in the Conv-GRU.

Our Modifications. Firstly, in XMem [9], the 1024-channel third-stage feature from the key encoder is directly concatenated with the memory readout for mask decoding. For efficiency, we instead first project the 1024-channel feature to 512 channels with a 1×1 convolutional layer before concatenating it with the memory readout. Secondly, in each upsampling block of the mask decoder, XMem uses a 3×3 convolution to pre-process the skip-connected feature. We replace it with a 1×1 convolution. Moreover, XMem [9] and prior works [46, 11] take the image, the target mask, and the sum of all non-target masks (excluding background) as input for the value encoder. We discard the 'sum of all non-target masks' as we note that it becomes uninformative when there are many objects in the scene – typical in open-world scenarios. We notice a moderate speed-up (22.6→25.8 FPS in DAVIS-2017 [5]) from these modifications.

A.3. Feature Memory

Representation. The feature memory consists of three parts: a sensory memory, a working memory, and a long-term memory. The sensory memory is represented by the hidden state of the Conv-GRU and contains positional information for temporal consistency that is updated every frame. Both the working memory and the long-term memory are attention-based and contain key-value pairs. The working memory is updated every r frames and has a maximum capacity of T_{max} frames. During each update, the 'key' feature from the key encoder and the 'value' feature from the value encoder will be appended to the working memory after segmentation of the current frame. When the working memory reaches its capacity, the oldest $T_{\text{max}} - T_{\text{min}}$ frames will be consolidated into the long-term memory. Please refer to [9] for details.

Memory Reading. The last hidden state of the Conv-GRU is used as the memory readout of the sensory memory. For the working and long-term memory, we compute a query from the query frame and perform space-time memory reading [46] to read from both types of memory. For spatial dimensions H, W, the memory readout for the sensory memory is $C_h \times H \times W$ and the memory readout for the working/long-term memory is $C_v \times H \times W$. In XMem, $C_h = 64$ and $C_v = 512$ and these two features are concatenated together as the final memory readout $\mathbf{F}$.

Our Modifications. For better temporal consistency, we expand the channel size C_h of the sensory memory to $C_h = C_v = 512$. For efficiency, we use 'addition' instead of the original 'concatenation' to fuse the memory readout from the sensory memory with the working/long-term memory. Besides, we supervise the sensory memory with an auxiliary loss – a 1×1 convolution is applied to the sensory memory to produce the weights and biases of a linear classifier on the stride 16 image feature (from the key encoder) for mask prediction. Cross-entropy loss with a weight of 0.1 is applied on this predicted mask and the network is trained end-to-end.

A.4. Inference Hyperparameters

Following [9], the sensory memory is updated every frame. A new memory frame is added to the working memory every r-th frame. We synchronize r with our in-clip consensus frequency, such that every in-clip consensus result is added to the working memory. Following the default hyperparameters in [9], we set $T_{\text{max}} = 10$, $T_{\text{min}} = 5$, the maximum number of long-term memory elements to be $10,000$, and use top-k filtering [10] with $k = 30$.

A.5. Training

XMem is first pretrained on static image segmentation datasets [56, 60, 72, 33, 8] by synthesizing small video clips of three frames with affine and thin-spline deformations. It is then trained on two video datasets: YouTubeVOS [65] and DAVIS [47] by sampling clips of length eight.

Our Modifications. We make three major modifications to the training process for better robustness:

1. We introduce the more challenging OVIS [48] data into training as we find models have already saturated and produced almost perfect segmentation results on DAVIS and YouTubeVOS during training.

2. We use a 'stable' data augmentation pipeline which leads to better temporal consistency. Current state-of-the-art data augmentation pipelines use aggressive augmentation, applying different rotations [9] or crops [70] to frames within the same sequence. This encourages an invariant appearance model but harms the learning of temporal information. We instead use the same rotation and crop augmentation for a video sequence. Figure S1 visualizes the difference.

3. We clip the norm of the gradient at 3.0 during training. We find that this leads to faster convergence and more stable training.

We use a batch size of 16, the same loss function (hard-mining cross-entropy loss with a warm-up and soft DICE loss) as XMem [9], and the AdamW [41] optimizer. During pre-training, we use a learning rate of $2e$-5 for 80,000 iterations. During main training, we use a learning rate of $1e$-5 for 150,000 iterations with a learning rate decay of 0.1 at the 120,000-th iteration and the 140,000-th iteration.

A.6. Video Object Segmentation Evaluation

We compare our temporal propagation model with state-of-the-art methods on three common video object segmentation benchmarks: DAVIS-2017 validation/test-dev [47], YouTubeVOS-2019 validation [65], and MOSE validation [16]. Table S1 tabulates our results. We resize all input such that the shorter side is 480px and bilinearly upsample the output back to the original resolution following XMem [9]. All frames in YouTubeVOS [65] are used by default. Our simple modifications bring noticeable improvements on all benchmarks. Though, we find that the overall framework is more important than these design choices (Section A.7.2).

A.7. Ablation Studies

A.7.1 On VOS Tasks

We assess the effects of our modifications on the training process on VOS tasks which purely evaluate temporal propagation performance. In addition to the standard DAVIS [47] dataset, we additionally convert the validation

Figure S1. AOT [70] and XMem [9] use different crops and rotations within a sequence respectively. We fix both within a sequence to encourage the learning of positional information.

	MOSE			DAVIS-17 val			DAVIS-17 test-dev			YouTubeVOS-2019 val					
Method	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$	$\mathcal{G}$	$\mathcal{J}_s$	$\mathcal{F}_s$	$\mathcal{J}_u$	$\mathcal{F}_u$	FPS
STCN [11]	52.5	48.5	56.6	85.4	82.2	88.6	76.1	72.7	79.6	82.7	81.1	85.4	78.2	85.9	13.2
AOT-R50 [70]	58.4	54.3	62.6	84.9	82.3	87.5	79.6	75.9	83.3	85.3	83.9	88.8	79.9	88.5	6.4
XMem [9]	56.3	52.1	60.6	86.2	82.9	89.5	81.0	77.4	84.5	85.5	84.3	88.6	80.3	88.6	22.6
DEVA (ours), w/o OVIS	60.0	55.8	64.3	86.8	83.6	90.0	82.3	78.7	85.9	85.5	85.0	89.4	79.7	88.0	**25.3**
DEVA (ours), w/ OVIS	**66.5**	**62.3**	**70.8**	**87.6**	**84.2**	**91.0**	**83.2**	**79.6**	**86.8**	**86.2**	**85.4**	**89.9**	**80.5**	**89.1**	**25.3**

Table S1. Comparison of DEVA's temporal propagation module with state-of-the-art video object segmentation methods. FPS is measured on YouTubeVOS-2019 validation with a V100 GPU. All available frames in YouTubeVOS are used by default.

sets of OVIS [48] and UVO [61] to the VOS format. Following DAVIS [47], we discard any segments that do not appear in the first frame and provide the first-frame ground-truth segmentation as input. These datasets are more diverse and allow for a more complete evaluation of temporal propagation performance. Table S2 tabulates our findings. For a fair comparison, we also re-train the original XMem [9] with additional OVIS [48] data. A qualitative comparison of aggressive *vs*. stable data augmentation is illustrated in Figure S2.

Variant	DAVIS	OVIS	UVO	FPS
XMem [9]	86.1	69.0	82.7	22.6
XMem [9] train w/ OVIS	86.1	72.0	83.0	22.6
With all our modifications	**87.6**	**75.7**	**83.5**	**25.8**
w/o stable data aug.	87.5	73.6	83.2	**25.8**
w/o gradient clipping	85.2	71.3	82.7	**25.8**

Table S2. $\mathcal{J}\&\mathcal{F}$ performance comparisons of XMem [9] and our different modifications on VOS tasks.

A.7.2 On Large-Scale Video Panoptic Segmentation

Next, we assess whether these improvements in VOS tasks transfer to target tasks like video panoptic segmentation.

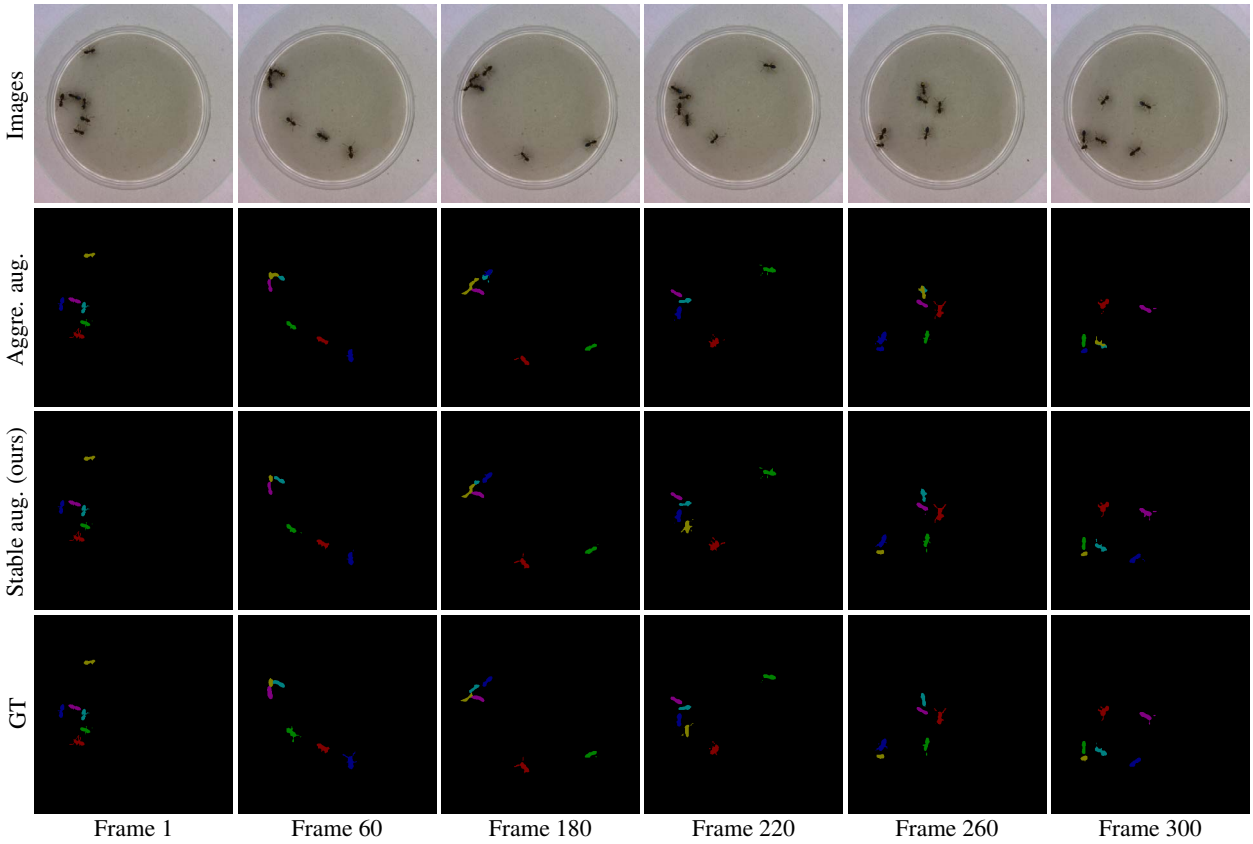

Figure S2. Comparison of methods tracking a group of ants with almost identical appearance. The variant with aggressive augmentation fails for the yellow, blue, and cyan ants toward the end while ours with stable data augmentation tracks all ants successfully. Ground-truth is annotated by us with an interactive image segmentation method, f-BRS [57]. Zoom in for details.

Varying Temporal Propagation Model	VPQ^1	VPQ^2	VPQ^4	VPQ^6	VPQ^8	VPQ^{10}	VPQ^∞	$\overline{VPQ}$	STQ
With standard XMem [9]	41.9	41.3	40.6	39.9	39.5	39.0	35.4	37.9	41.3
With our modified XMem [9]	**42.1**	**41.5**	**40.8**	**40.1**	**39.7**	**39.3**	**36.1**	**38.3**	**41.5**
Varying Training Data of Temporal Propagation	VPQ^1	VPQ^2	VPQ^4	VPQ^6	VPQ^8	VPQ^{10}	VPQ^∞	$\overline{VPQ}$	STQ
Image pretraining + 100% video training	**42.1**	**41.5**	**40.8**	**40.1**	**39.7**	**39.3**	**36.1**	**38.3**	**41.5**
Image pretraining + 50% video training	42.0	41.4	40.7	40.1	39.7	39.4	36.0	38.3	41.3
Image pretraining + 10% video training	40.7	40.1	39.3	38.5	38.1	37.7	34.6	36.8	40.1
Training on 100% YouTube-VOS [65] only	41.4	40.9	40.2	39.5	39.1	38.7	35.6	37.8	41.0
Training on 50% YouTube-VOS [65] only	40.5	39.4	38.0	36.6	35.6	34.4	31.3	34.4	37.8

Table S3. Performance comparisons of our method with different temporal propagation model settings on the VIPSeg [45] validation set. For a fair comparison, all are semi-online with a Mask2Former-R50 [7] image model input.

We compare our final model with/without these modifications on a large-scale video panoptic segmentation dataset VIPSeg [45] with the Mask2Former-R50 [7] backbone. Table S3 (top) tabulates our findings. Note, our method still works well even without our modifications to the temporal propagation network. We find the overall framework to be more important than particular design choices within the temporal propagation model.

B. Training Data of Temporal Propagation

B.1. Sensitivity to Training Data

We train the temporal propagation on class-agnostic image segmentation and mask propagation data as described in Section A.5. We note that these datasets are cheap to access and amass as they do not require class-specific annotations. Here, we evaluate the importance of large-scale training of the temporal propagation model. We vary the amount of class-agnostic video-level training data under two settings: 1) with full image pretraining, and all three mask propagation datasets (DAVIS [47], YouTubeVOS [65] and OVIS [48]), and 2) without image pre-taining and using YouTubeVOS as the only training data. Table S2 (bottom) tabulates our findings on the VIPSeg [45] validation set. The performance of our model decays gracefully with fewer training data.

B.2. Class Overlaps with VIPSeg

While we train the temporal propagation network in a class-agnostic setting, the segmented objects in the training set might have object categories that overlap with the target task (e.g., with the classes in VIPSeg [45]). Here, we investigate the effect of this overlap of temporal propagation training data with target task data on the final performance.

For this, we train the temporal propagation network with only YouTubeVOS [65] data which has 65 object categories (other datasets that we use for training have no class annotation). We manually match these 65 categories with the classes in VIPSeg [45] to partition the classes of VIPSeg into three sets: 'overlapping', 'non-overlapping', or 'ambiguous'.[1] We then evaluate the final task performance on the overlapping and the non-overlapping sets separately, while ignoring the 'ambiguous' set. We perform the same evaluation on an end-to-end method, Video-K-Net [34], as a measure of 'baseline difficulty' for each set. Table S4 tabulates our findings. We observe no significant difference between the overlapping and non-overlapping set when accounting for the difficulty delta (Δ) observed in the baseline. This indicates that our class-agnostic training does not overfit to the object categories in the training set.

C. Detailed Experimental Settings and Results

C.1. Large-Scale Video Panoptic Segmentation

Following the standard practice [45], we use the 720p version of the VIPSeg [45] dataset. We evaluate using its validation set (343 videos) and compute VPQ/STQ using the official codebase. During temporal propagation, we downsample the videos such that the shortest side is 480px and bilinearly upsample the result back to the original resolution following [9].

[1] The overlapping set includes flag, parasol_or_umbrella, car, bus, truck, bicycle, motorcycle, ship_or_boat, airplane, person, cat, dog, horse, cattle, skateboard, ball, box, bottle_or_cup, table_or_desk, mirror, and train (21 in total). The ambiguous set includes other_animal, bag_or_package, toy, and textiles (4 in total). The remaining (99) classes in VIPSeg are in the non-overlapping set.

Method	$\overline{\text{VPQ}}_{\text{overlap}}$	$\overline{\text{VPQ}}_{\text{no-overlap}}$	$\Delta_{\text{overlap}\rightarrow\text{non-overlap}}$
Video-K-Net	25.0	25.7	+0.7
Ours	38.1	38.6	+0.5

Table S4. Performance comparison on different classes of VIPSeg that overlap or do not overlap with the training data of temporal propagation. As a baseline, we use Video-K-Net-R50 [34]. For ours, we use Mask2Former-R50 with a temporal propagation model that is only trained on YouTubeVOS [65] and evaluated in a semi-online setting.

Video Panoptic Segmentation (VPS) requires the prediction of class labels. We obtain these labels from the image segmentation model and use online majority voting to determine the output label. Formally, we keep a list of class labels $\mathbf{Cl}_i$ for each object r_i. When an existing (propagated) segment r_i matches with a segment from the in-clip consensus c_j, i.e., $a_{ij} = 1$, we take the class label from the consensus c_j and append it to the list $\mathbf{Cl}_i$. At the output of every frame, we determine the class label associated with segment r_i by performing majority voting in $\mathbf{Cl}_i$. Note, in accordance with VPS evaluation [27], an object can only have one class label throughout the video. This means a change in class label necessitates a change in object id, which we also implemented. Thus, a change in class label might lead to lower association accuracy. An alternative algorithm would be to use the final major voting result to retroactively apply the class label in all frames, which would however not be strictly online/semi-online.

Running time Analysis Under our default semi-online setting, we use a clip size of 3 and perform merging every 5 frames (i.e., invoking the image model on 60% of all frames). We report time on VIPSeg [45], averaged across all frames, on an A6000 GPU. The mask propagation module takes 83ms per frame (VIPSeg has more objects per video than the standard VOS timing benchmark DAVIS-2017). For every merge, pre-processing (spatial alignment and finding pairwise IoU) takes 211ms, and solving the integer program takes 15ms. For the image model (R50 backbone), both Video-K-Net [34] and Mask2Former [7] take around 200ms per frame. Overall, our method runs at 4.0fps. Meanwhile, state-of-the-art Video-K-Net runs at 4.9fps. Ours is 18% slower but has a 52% higher $\overline{\text{VPQ}}$.

C.2. Open-World Video Segmentation

We evaluate on the validation (993 videos) and test (1421 videos) sets of BURST [2]. As in Section C.1, we downsample the videos during temporal propagation such that the shortest side is 480px and bilinearly upsample the result

back to the original resolution following [9]. For efficiency, we process only every three frames. Since the ground-truth is sparse (annotated every 24 or 30 frames), we can still perform a complete evaluation.

For the Mask2Former [7] image model, we follow BURST [2] and use the best-performing Swin-L checkpoint trained on COCO [37] provided by the authors. For the EntitySeg [49] image model, we also use the best available Swin-L model checkpoint trained on COCO [37]. For overlapping predictions, we use the post-processing for panoptic segmentation in Mask2Former [7] to resolve them.

We assess Open World Tracking Accuracy (OWTA) using official tools. OWTA is the geometric mean of Detection Recall (DetRe) and Association Accuracy (AssA). Please refer to [2] for details. For completeness, we additionally report DetRe and AssA of baselines and our method in Table S5.

C.3. Referring Video Segmentation

To evaluate on Ref-DAVIS [25] and Ref-YouTubeVOS [55], we use ReferFormer Swin-L [64] as the image model. The network is first pretrained on Ref-COCO [71], Ref-COCO+ [71], and G-Ref [44] datasets and finetuned on Ref-YouTubeVOS [55] following [64]. Unlike in video panoptic segmentation or open-world video segmentation, we do not need to use integer programming to associate segments from the image model in different frames. This is because each segment corresponds to a known language expression. Thus, we process each object independently and use argmax to fuse the final segmentations. As mentioned in the main paper, we employ an offline setting as in prior works [55, 64, 17].

In the offline setting, we first perform in-clip consensus by selecting 10 uniformly spaced frames in the video and using the frame with the highest confidence given by the image model as a 'key frame' for aligning the other frames. Soft probability maps are used in the consensus to preserve confidence levels in the prediction. We then forward- and backward-propagate from the key frame without incorporating additional image segmentations.

Formally, for a given object, we denote its soft probability map and confidence score given by the image model as $\mathbf{Pr}_t \in [0,1]^{H\times W}$ and $c_t \in [0,1]$ respectively. We denote the frame index of the ten chosen frames as $\mathbf{T}_c = \{t_1, t_2, ..., t_{10}\}$.

We aim to compute a soft probability consensus $\mathbf{Cs}_{t_k}$ at a keyframe index t_k by a weighted summation of the soft probability maps of the chosen frames $\{\mathbf{Pr}_{t_1}, \mathbf{Pr}_{t_2}, ..., \mathbf{Pr}_{t_{10}}\}$:

$$\mathbf{Cs}_{t_k} = \sum_{i\in\mathbf{T}_c} w_i \mathbf{Pr}_i, \tag{S1}$$

where w_i is a weighting coefficient, with $\sum_i w_i = 1$.

We use the frame with the highest confidence predicted by the image model as the keyframe:

$$t_k = \mathrm{argmax}_{i\in\mathbf{T}_c} c_i. \tag{S2}$$

We compute the weighting coefficients using a softmax of the confidences such that we weigh confident predictions more:

$$w_i = \frac{e^{c_i}}{\sum_{i\in\mathbf{T}_c} e^{c_i}}. \tag{S3}$$

After the consensus, $\mathbf{Cs}_{t_k}$ is used to initialize forward and backward propagation from frame t_k without incorporating additional image segmentations. The propagation is implemented as standard semi-supervised video object segmentation inference with the keyframe as initial guidance. During propagation, the internal memory $\mathbf{H}$ is updated every 5 frames using its own prediction as in [9].

C.4. Unsupervised Video Object Segmentation

For single-object unsupervised video object segmentation (DAVIS-2016 [47]), we use DIS [51] as the image segmentation model. Since it does not provide segmentation confidence, we approximate it with the normalized area of the predicted mask to ignore null detections, i.e., $c_i = \frac{1}{HW}\|\mathbf{Pr}_i\|_1$.

For multi-object unsupervised video object segmentation (DAVIS-2017 [5]), we follow our semi-online protocol in open-world video segmentation. The exception being DAVIS-2017 [5] allows a maximum of 20 objects in the prediction. We overcome this limitation online by only accepting the first 20 objects and discarding the rest. When there are more than 20 objects in the frame, we prioritize the ones with larger areas as they are less likely to be noisy.

D. Results on YouTube-VIS

Here we present additional results on the small-vocabulary YouTube-VIS [69] dataset, but unsurprisingly recent end-to-end specialized approaches perform better because a sufficient amount of data is available in this case. For this task, we use our online video panoptic segmentation setting. Besides the difference in the scale of vocabularies, our method assumes that no two objects occupy the same pixel, and produces a non-overlapping mask. Although this assumption is usually true, it harms the Average Precision (AP) evaluation of our method in VIS, with other methods typically outputting many ($\geq$100) potentially overlapping proposals for higher recall. We provide our result in Table S6.

E. Qualitative Results

E.1. Visualization

For all results (see our project page), we associate each object id with a unique color. When a segment changes

Method		Validation								
		All			Common			Uncommon		
		DetRe	AssA	OWTA	DetRe	AssA	OWTA	DetRe	AssA	OWTA
Mask2Former	w/ Box tracker [2]	66.9	55.8	60.9	78.7	57.1	60.9	20.1	30.5	24.0
Mask2Former	w/ STCN tracker [2]	67.0	62.6	64.6	78.8	64.1	71.0	20.0	33.3	25.0
OWTB [39]		70.9	45.2	56.2	76.8	47.0	59.8	46.5	34.3	38.5
Mask2Former	w/ ours online	72.1	67.5	69.5	80.2	69.9	74.6	39.8	46.4	42.3
Mask2Former	w/ ours semi-online	71.8	**68.5**	**69.9**	**80.3**	**70.7**	**75.2**	37.9	46.8	41.5
EntitySeg	w/ ours online	72.3	66.0	68.8	77.7	68.4	72.7	**50.3**	50.2	49.6
EntitySeg	w/ ours semi-online	**72.4**	67.1	69.5	78.1	69.3	73.3	50.0	**52.2**	**50.5**
		Test								
		All			Common			Uncommon		
Method		DetRe	AssA	OWTA	DetRe	AssA	OWTA	DetRe	AssA	OWTA
Mask2Former	w/ Box tracker [2]	61.5	51.1	55.9	71.4	52.5	61.0	21.1	30.0	24.6
Mask2Former	w/ STCN tracker [2]	61.6	54.1	57.5	71.5	55.7	62.9	21.0	28.6	23.9
OWTB [39]		70.9	45.2	56.2	76.8	47.0	59.8	46.5	34.3	38.5
Mask2Former	w/ ours online	72.2	68.6	70.1	**79.8**	70.8	75.0	40.7	49.2	44.1
Mask2Former	w/ ours semi-online	71.9	**69.6**	**70.5**	79.7	**71.7**	**75.4**	39.5	50.7	44.1
EntitySeg	w/ ours online	**72.5**	67.3	69.5	77.3	69.2	72.9	**52.3**	55.0	53.0
EntitySeg	w/ ours semi-online	72.4	67.7	69.8	77.4	69.5	73.1	51.9	**55.9**	**53.3**

Table S5. Extended results comparing baselines and our methods in the validation/test sets of BURST [2]. Baseline performances are transcribed from [2].

Method	mAP	AP@75
MaskProp [4]	40.0	42.9
Video-K-Net [34]	40.5	44.5
MinVIS [22]	**47.4**	**52.1**
Mask2Former [7] w/ Ours	40.8	44.3

Table S6. Performance comparisons on YouTube-VIS 2019 validation. All models use a ResNet-50 backbone. Note MinVIS is optimized for small-vocabulary YouTube-VIS and underperforms by 8.4 $\overline{\text{VPQ}}$ compared with our method in large-vocabulary VIPSeg (Tab. 6 of the main paper, Query assoc. *vs*. Ours).

color, its object id has changed. This change might happen often (e.g., flicker) if the method is not stable. We additionally show an 'overlay' which is a composite of the colored segmentation with the input image.

E.2. Large-Scale Video Panoptic Segmentation

We compare our method with state-of-the-art Video-K-Net [34]. We use the semi-online setting Mask2Former [7] as the image model. Videos are taken from VIPSeg [45] validation set.

E.3. Open-World Video Segmentation

We compare our method with the best open-world segmentation baseline (Mask2Former + STCN tracker). We use the semi-online setting EntitySeg [49] as the image model. Videos are collected from BURST [2] and the Internet.

E.4. Referring Video Segmentation in the Wild

We compare our method with state-of-the-art referring video segmentation ReferFormer [64]. We are interested in the open-world setting beyond standard Ref-DAVIS [25] and Ref-YouTubeVOS [55]. For this, we use a recent open-world referring image segmentation model X-Decoder [75] as our image model. The agility to switch image backbones and use the latest advancements in image segmentation is one of the main advantages of our decoupled formulation. We employ an offline setting following our referring video segmentation evaluation protocol (Section C.3). Note, ReferFormer [64] is also offline. Our model can segment rare objects like 'wheel of fortune' accurately.