TarViS: A Unified Approach for Target-based Video Segmentation

Ali Athar[1] Alexander Hermans[1] Jonathon Luiten[1,2] Deva Ramanan[2] Bastian Leibe[1]
[1]RWTH Aachen University, Germany [2]Carnegie Mellon University, USA
{athar,hermans,luiten,leibe}@vision.rwth-aachen.de deva@cs.cmu.edu

Abstract

The general domain of video segmentation is currently fragmented into different tasks spanning multiple benchmarks. Despite rapid progress in the state-of-the-art, current methods are overwhelmingly task-specific and cannot conceptually generalize to other tasks. Inspired by recent approaches with multi-task capability, we propose TarViS: a novel, unified network architecture that can be applied to any task that requires segmenting a set of arbitrarily defined 'targets' in video. Our approach is flexible with respect to how tasks define these targets, since it models the latter as abstract 'queries' which are then used to predict pixel-precise target masks. A single TarViS model can be trained jointly on a collection of datasets spanning different tasks, and can hot-swap between tasks during inference without any task-specific retraining. To demonstrate its effectiveness, we apply TarViS to four different tasks, namely Video Instance Segmentation (VIS), Video Panoptic Segmentation (VPS), Video Object Segmentation (VOS) and Point Exemplar-guided Tracking (PET). Our unified, jointly trained model achieves state-of-the-art performance on 5/7 benchmarks spanning these four tasks, and competitive performance on the remaining two. Code will be made public upon acceptance.

1. Introduction

The ability to understand video scenes has been a long-standing goal of computer vision research because of wide-ranging applications in intelligent vehicles and robots. Early approaches tackled simpler tasks involving contour-based [33, 40] and box-level tracking [21, 25, 41, 53], background subtraction [20, 62], and motion segmentation [8, 50]. The deep learning boom then revolutionized the landscape by enabling methods to perform pixel-precise segmentation on challenging, real-world videos. In the past few years, a number of benchmarks have emerged, which evaluate how well methods can perform video segmentation according to various task formulations. Over time,

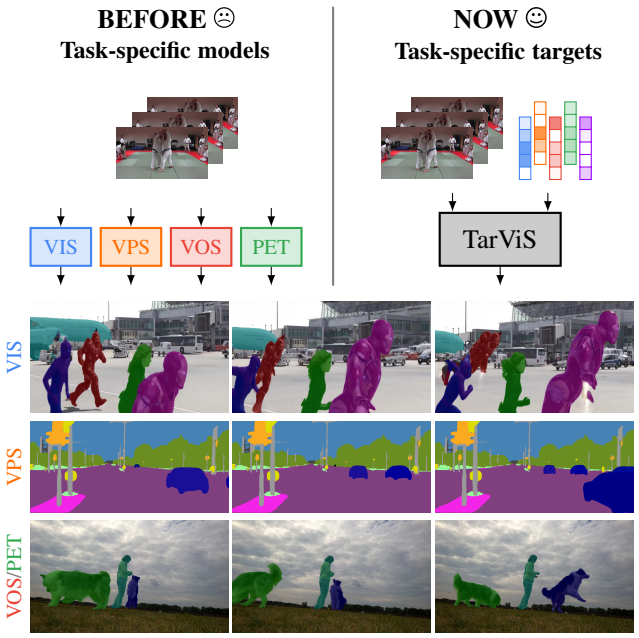


Figure 1. Predicted results from a jointly trained TarViS model for four different video segmentation tasks.

these tasks/benchmarks have ballooned into separate research sub-communities. Although existing methods are rapidly improving the state-of-the-art for these benchmarks, each of them typically tackles only one narrowly-defined task, and generalizing them is non-trivial since the task definition is baked into the core approach.

We argue that this fragmentation is unnecessary because video target segmentation tasks all require the same high-level capability, namely that of identifying, localizing and tracking rich semantic concepts. Meanwhile, recent progress on Transformer networks has enabled the wider AI research community to move towards unified, multi-task architectures [1, 30, 31, 39, 59], because the attention operation [63] is well-suited for processing feature sets with arbitrary structure and data modality. These developments give us the opportunity to unify the fractured landscape of target-based video segmentation. In this paper, we propose TarViS: a novel architecture which enables a single, unified

model to be *jointly trained for multiple video segmentation tasks*. During inference, the *same model can perform different tasks at runtime* by specifying the segmentation target.

The core idea is that TarViS tackles the generic task of segmenting a set of arbitrary *targets* in video (defined as semantic classes or as specific objects). These targets are encoded as *queries* which, together with the video features, are input to a Transformer-based model. The model iteratively refines these queries and produces a pixel-precise mask for each target entity. This formulation conceptually fuses all video segmentation tasks [3, 55, 67, 73] which fall under the umbrella of the above-mentioned generic task, because they differ only in how the *targets* are defined. During both training and inference, TarViS can hot-swap between tasks at run-time by providing the desired target query set.

To demonstrate our generalization capability, we tackle four different tasks: (1) Video Instance Segmentation (VIS) [55, 73], (2) Video Panoptic Segmentation (VPS) [35], (3) Video Object Segmentation [54], and (4) Point Exemplar-guided Tracking [3] (PET). For VIS, the segmentation targets are all objects in the video belonging to a pre-defined set of classes. The target set for VPS includes that for VIS, and additionally, a set of non-instantiable *stuff* semantic classes. For VOS, the targets are a specific set of objects for which the first-frame ground-truth mask is provided. PET is a more constrained version of VOS which only provides the location of a single point inside the object, rather than the full object mask.

Existing methods for these tasks lack generalization capability because task-specific assumptions are typically baked into the approach (see Sec. 2 and 3 for details). In contrast, TarViS can tackle all four tasks with a unified model because we encode the task-specific targets as a set of queries, thus decoupling the network architecture from the task definition. Moreover, our approach can theoretically generalize further, *e.g.*, one could potentially define the target set as all objects described by a given text prompt, though this is beyond the scope of this paper.

To summarize, our contributions are as follows: we propose TarViS, a novel architecture that can perform any task requiring segmentation of a set of *targets* from video. For the first time, we are able to jointly train and infer a single model on a collection of datasets spanning the four aforementioned tasks (VIS, VPS, VOS, PET). Our experimental results show that TarViS performs competitively for VOS, and achieves state-of-the-art results for VIS, VPS and PET.

2. Related Work

Multi-task Models. Multi-task learning has a long history [11] with several architectures and training strategies [24, 36, 38, 52, 60, 77]. Earlier approaches mostly consist of a shared backbone with fixed task-specific heads, whereas we design a more general architecture for video segmentation with task-specific targets to specify what to segment. Our approach is inspired by recent attention-based models, *e.g.*, PerceiverIO [30, 31], which can be trained on diverse data modalities and task-specific heads are replaced with output queries. UViM [39] follows a similar direction by creating a unified architecture for diverse dense prediction tasks. However, both of these models are trained separately for different tasks. Recent, powerful multi-task vision language models such as Flamingo [1] and GATO [59] tackle a multitude of tasks by requiring a sequence of task-specific input-output examples to prime the model. This is conceptually similar to our task-specific targets, however, our model does not require per-task priming. Moreover, our targets are not modeled as sequence prompts, and we aim for a video segmentation model which is several orders of magnitude smaller. In the realm of video tracking and segmentation, the recently proposed UNICORN [72] model tackles multiple object tracking-related tasks with a unified architecture. Unlike TarViS, however, UNICORN follows the task-specific output head approach and is generally oriented towards box-level tracking tasks [22, 46, 48, 76], thus requiring non-trivial modifications to tackle VPS or PET.

Query-based Transformer Architectures. Several works [2, 10, 13, 30, 31, 47, 66, 79] use query-based Transformer architectures for various tasks. The fundamental workhorse for task learning here is the iterative application of self- and cross-attention, where a set of query vectors (*e.g.*, representing objects) are refined by interacting with each other, and with the input data sample (*e.g.*, an image). Unlike existing methods which use queries in a task-specific context, TarViS adopts a query-based Transformer architecture in which the queries serve as a mechanism for decoupling the task definition from the architecture, *i.e.*, our model can learn to tackle different tasks while being agnostic to their definition because the latter is effectively abstracted behind a set of queries.

Task-specific Video Segmentation Methods. Current Video Instance Segmentation (VIS) approaches broadly work by predicting object tracks from the input video, followed by classification into a pre-defined set of categories. Several approaches [6, 9, 23, 28, 34, 55, 65, 70, 73] are based on the tracking-by-detection paradigm, some model video as a joint spatio-temporal volume [4, 5], whereas many recent works [12, 26, 29, 66, 69] adopt Transformer-based architectures (comparison to our approach in Sec. 3.1).

For Video Panoptic Segmentation (VPS), methods [35, 56, 67] generally extend image-level panoptic approaches [14] by employing multi-head network architectures for semantic segmentation and instance mask regression, classification, and temporal association. In the Video Object Segmentation (VOS) community, state-of-

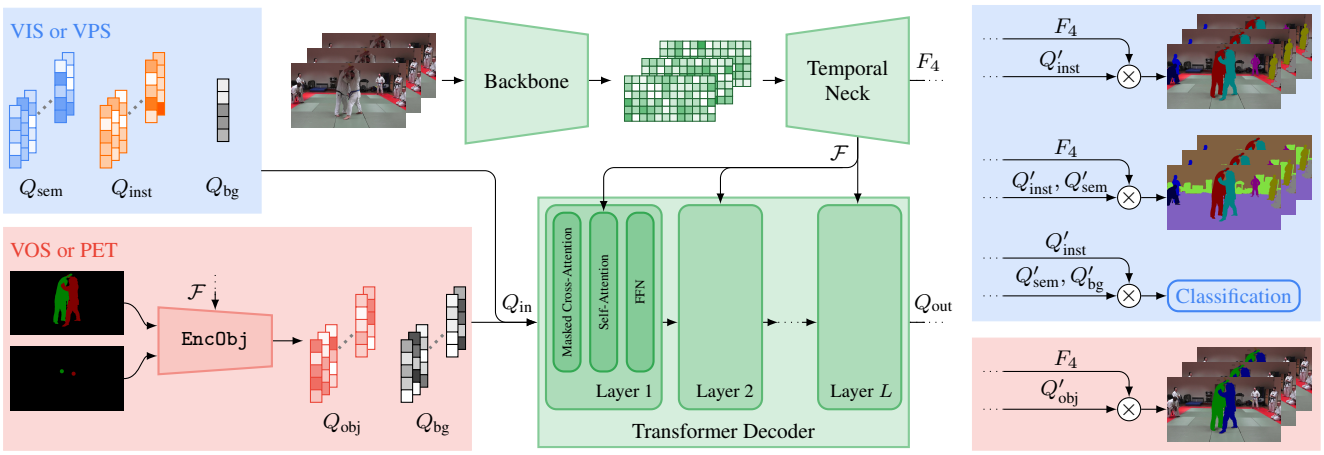


Figure 2. **TarViS Architecture.** Segmentation targets for different tasks are represented by a set of abstract target queries Q_{in}. The core network (in green) is agnostic to the task definitions. The inner product between the output queries Q_{out} and video feature F_4 yields segmentation masks as required by the task.

the-art methods are broadly based on the seminal work of Oh *et al*. [51], which learns *space-time correspondences* between pixels in different video frames, and then uses these to propagate the first-frame masks across the video. Subsequent methods [15–17, 61, 64, 71, 74, 74, 75] have significantly improved the performance and efficiency of this approach. Point Exemplar-guided Tracking (PET) [3, 27] is a relatively new task for which the current best approach [3] involves regressing a pseudo-ground-truth mask from the given point coordinates, and then applies a state-of-the-art VOS method [17] to this mask.

The above methods thus incorporate task-specific assumptions into their core approach. This can be beneficial for per-task performance, but makes it difficult for them to generalize across tasks. By contrast, TarViS can tackle all four aforementioned tasks, and generally any target-based video segmentation task, with a single, unified, jointly trained model.

3. Method

TarViS can segment arbitrary targets in video since the architecture is flexible with respect to how these targets are defined, thus enabling us to conceptually unify and jointly tackle the four aforementioned tasks (VIS, VPS, VOS, PET). The architecture is illustrated in Fig. 2.

For all tasks, the common input to the network is an RGB video clip of length T denoted by $V \in \mathbb{R}^{H \times W \times T \times 3}$. This is input to a 2D backbone network which produces image-level feature maps, followed by a *Temporal Neck*, which enables feature interaction across time and outputs a set of temporally consistent, multi-scale, D-dimensional feature maps $\mathcal{F} = \{F_{32}, F_{16}, F_8, F_4\}$ where $F_s \in \mathbb{R}^{\frac{H}{s} \times \frac{W}{s} \times T \times D}$. The feature maps $\mathcal{F}$ are then fed to our Transformer decoder, together with a set of queries Q_{in} which represent the segmentation targets. The decoder applies successive layers of self- and masked cross-attention wherein the queries are iteratively refined by attending to each other, and to the feature maps, respectively. The refined queries output by the decoder are denoted with Q_{out}. The following subsections explain how TarViS tackles each task in detail.

3.1. Video Instance Segmentation

VIS defines the segmentation target set as all objects belonging to a set of predefined classes. Accordingly, the input query set Q_{in} for VIS contains three types of queries: (1) *semantic queries* denoted by $Q_{\text{sem}} \in \mathbb{R}^{C \times D}$ where C is the number of classes defined by the dataset, *i.e*., each D-dimensional vector in Q_{sem} represents a particular semantic class. (2) *instance queries* denoted by $Q_{\text{inst}} \in \mathbb{R}^{I \times D}$ where I is assumed to be an upper bound on the number of instances in the video clip, and (3) a *background query* denoted by $Q_{\text{bg}} \in \mathbb{R}^{1 \times D}$ to capture inactive instance queries.

The three query sets are concatenated, *i.e*., $Q_{\text{in}} = \texttt{concat}(Q_{\text{sem}}, Q_{\text{inst}}, Q_{\text{bg}})$, and input to the Transformer decoder, which refines their feature representation through successive attention layers and outputs a set of queries $Q_{\text{out}} = \texttt{concat}(Q'_{\text{sem}}, Q'_{\text{inst}}, Q'_{\text{bg}})$. These are then used to produce temporally consistent instance mask logits by computing the inner product $\langle F_4, Q'_{\text{inst}} \rangle \in \mathbb{R}^{H \times W \times T \times I}$. To obtain classification logits, we compute the inner product $\langle Q'_{\text{inst}}, \texttt{concat}(Q'_{\text{sem}}, Q'_{\text{bg}}) \rangle \in \mathbb{R}^{I \times (C+1)}$.

The three types of queries are initialized randomly at the start of training and optimized thereafter. The instance queries Q_{inst} enable us to segment a varying number of objects from the input clip. During training, we apply Hungarian matching between the predicted and ground-truth instance masks to assign instance queries to video instances, and then supervise their predicted masks and classification

logits accordingly. When training on multiple datasets with heterogeneous classes, the semantic query sets are separately initialized per dataset, but Q_{inst} and Q_{bg} are shared.

Comparison to Instance Segmentation Methods. Several Transformer-based methods [10, 13, 66, 79] for image/video instance segmentation also use queries to segment a variable number of input instances. The key difference to our approach is the handling of object classes: existing works employ only instance queries which are input to a fully-connected layer with a fan-out of $C + 1$ to obtain classification (and background) scores. The notion of class-guided instance segmentation is thus baked into the approach. By contrast, TarViS is agnostic to the task-specific notion of object classes because it models them as arbitrary queries which are dynamic inputs to the network. The semantic representation for these queries is thus decoupled from the core architecture and is only learned via loss supervision. An important enabler for this approach is the background query Q_{bg}, which serves as a 'catch-all' class to represent everything that is not in Q_{sem}. It is used to classify non-active instance queries, and its mask logits are supervised to segment all non-object input pixels.

3.2. Video Panoptic Segmentation

VPS defines the segmentation targets as all objects belonging to a set of *thing* classes (*e.g.*, 'person', 'car'), and additionally, a set of non-instantiable *stuff* classes (*e.g.*, 'sky', 'grass') which cover all non-object pixels. TarViS can tackle VPS with virtually no modification to the workflow in Sec. 3.1. We can compute semantic segmentation masks for the input clip by simply taking the inner product between Q_{sem} and the video features: $\langle F_4, Q'_{\text{sem}} \rangle \in \mathbb{R}^{H \times W \times T \times C}$. Note that here, Q_{sem} contains queries representing both *thing* and *stuff* classes.

Comparison to VPS Methods. Current VPS datasets [35, 67] involve driving scene videos captured from moving vehicles. Methods tackling this task [35, 56] are based on earlier image panoptic segmentation approaches [14] which involve multi-head networks for semantic and instance segmentation prediction. In terms of image-level panoptic segmentation, Mask2Former [13] uses a Transformer-based architecture, but it models *stuff* classes as instances which are Hungarian-matched to the ground-truth target during training, whereas TarViS models semantic classes and instances using separate, designated queries.

3.3. Video Object Segmentation and Point Exemplar-guided Tracking

VOS and PET can be seen as instantiations of a general task where the segmentation targets are a set of O objects for which some ground-truth cue $\mathcal{G}$ is given. For VOS, $\mathcal{G}$ is provided in the form of first-frame object masks $M_{\text{obj}} \in \mathbb{R}^{O \times H \times W}$, whereas for PET, $\mathcal{G}$ is provided as the (x, y) coordinates $P_{\text{obj}} \in \mathbb{R}^{O \times 2}$ of a point inside each of the objects. TarViS jointly tackles these tasks by adopting a generalized approach in which the O target objects are encoded into a set of *object queries* Q_{obj}. Thus, both VOS and PET boil down to designing a function $\texttt{EncodeObjects}(\cdot)$ which regresses Q_{obj} from the given ground-truth cues $\mathcal{G}$ and feature maps $\mathcal{F}$:

$$Q_{\text{obj}} \leftarrow \texttt{EncodeObjects}(\mathcal{G}, \mathcal{F}). \tag{1}$$

Note that Q_{obj} is conceptually analogous to Q_{sem} and Q_{inst} used for VIS in that all three are abstract representations for their respective task-specific segmentation targets.

Video Object Segmentation. To implement `EncodeObjects` for VOS, we seek inspiration from HODOR [2], a recent method for weakly-supervised VOS, which encodes objects into concise *descriptors* as follows: the descriptors are initialized by average pooling the image features inside the object masks, followed by an iterative refinement where the descriptors attend to each other (self-attention) and to their respective soft-masked image features (cross-attention).

For TarViS, we employ a lightweight *Object Encoder* with a similar workflow to encode the objects as a set of queries Q_{obj}, but with two differences to HODOR [2]: instead of cross-attending to the entire image feature map ($H \cdot W$ points) with soft-masked attention, we apply hard-masked cross-attention to at most p_{max} feature points per object, where $p_{\text{max}} \ll H \cdot W$. Object masks containing more than p_{max} points are sub-sampled accordingly. This significantly improves the memory/run-time overhead of our Object Encoder. Secondly, we note that the process of distilling object features into a single descriptor involves a loss of object appearance information, which degrades performance. We therefore model each object with q_o queries (instead of one) by spatially dividing each object mask into q_o segments, *i.e.*, $Q_{\text{obj}} \in \mathbb{R}^{O \times q_o \times D}$ (we use $q_o = 4$).

In addition to Q_{obj}, we initialize a set of background queries $Q_{\text{bg}} \in \mathbb{R}^{B \times D}$ to model the non-target pixels in the reference frame. Following HODOR [2], we employ multiple background queries, which are initialized dynamically by dividing the video frame containing the ground-truth masks M_{obj} into a 4×4 grid and average pooling the non-object pixels in each grid cell. The Object Encoder jointly refines the background and object queries to yield $Q_{\text{in}} = \texttt{concat}(Q_{\text{obj}}, Q_{\text{bg}})$. During training, the mask logits for the multiple background queries are aggregated per-pixel by applying $\texttt{max}(\cdot)$ and supervised to segment all pixels not part of the target object set.

The remaining workflow follows that for VIS and VPS: Q_{in} is input to the Transformer decoder together with the video features $\mathcal{F}$. The refined output query set $Q_{\text{out}} =$

$\texttt{concat}(Q'_{\text{obj}}, Q'_{\text{bg}})$ is then used to compute the inner product $\langle F_4, Q'_{\text{obj}} \rangle \in \mathbb{R}^{H \times W \times T \times O \times q_o}$. Subsequently, $\texttt{max}(\cdot)$ is applied on the q_o-sized dimension to obtain the final mask logits for the O target objects.

Point Exemplar-guided Tracking. For PET we implement `EncodeObjects` in the exactly same way as VOS: the given point coordinates P_{obj} are converted into a mask with just one non-zero pixel, followed by iterative refinement by the Object Encoder (with shared weights for VOS and PET). The only difference is that here we represent each of the O objects with just one query, *i.e.*, $Q_{\text{obj}} \in \mathbb{R}^{O \times D}$ ($q_o = 1$). The subsequent workflow is also identical to that for VOS: the queries are refined by the Transformer decoder followed by an inner product with F_4 to obtain object mask logits.

Comparison to VOS and PET Methods. Current state-of-the-art VOS methods are largely based on STM [51]. It involves learning pixel-to-pixel correspondences across video frames, which are then used to propagate the given object mask across the video. This approach is effective since every pixel in the given mask can be individually mapped to future frames, thus preserving fine-grained object details. The core approach is, however, task-specific since it assumes the availability of first-frame object masks, and does not generalize to the PET (see Sec. 4.2). PET can be viewed as a more constrained version of VOS, where only a single object point is provided instead of the full mask. Consequently, PET [3] is currently tackled by casting it as a VOS problem by using an image instance segmentation network [13] to regress pseudo-ground-truth object masks from the given point coordinates P_{obj}.

On the other hand, our approach of encoding objects as concise queries causes loss of fine-grained object appearance information, but it has the advantage of being agnostic to how $\mathcal{G}$ is defined. As evident from the unified workflow for VOS and PET, any variation of these tasks with arbitrary ground-truth cues $\mathcal{G}$ can be seamlessly fused into our architecture as long as we can implement an effective `EncodeObjects` function to regress Q_{obj} from the given $\mathcal{G}$.

3.4. Network Architecture

Temporal Neck. As explained earlier, TarViS produces target masks by computing the inner product between Q_{out} and the video feature map F_4. For this to be effective, the per-pixel features $\mathcal{F}$ must be aligned for the same, and dissimilar for different targets. Some image instance segmentation methods [13, 79] apply *Deformable Attention* [79] to the backbone feature maps to efficiently learn consistent, multi-scale image features. For TarViS, however, the features must also be temporally consistent across the entire input video clip. To achieve this, we propose a novel *Temporal Neck* architecture which is based on the work of Bertasius *et al*. [7] for video action classification. We enable efficient spatio-temporal feature interaction by applying two types of self-attention in an alternating fashion: the first is spatially global and temporally localized, whereas the second is spatially localized and temporally global. The first operation is implemented with Deformable Attention, following existing work [12, 79]. The second operation, Temporal Attention, involves dividing the input space-time volume into a grid along the spatial axes, and then applying self-attention to the space-time feature volume inside each grid cell. Both operations allow feature interaction across multiple scales. Both attention operations are illustrated in Fig. 3. We exclude F_8 from temporal attention since we found this to be more memory-efficient without negatively impacting prediction quality.

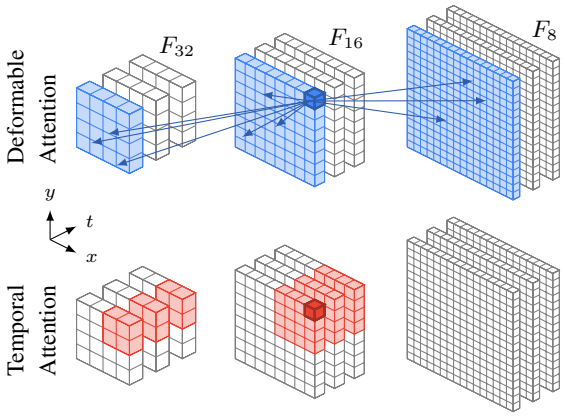


Figure 3. **Temporal Neck Layer.** Colored regions denote the attention field w.r.t the selected pixel (darkened). Deformable Attention is spatially unrestricted but temporally limited to a single frame, whereas Temporal Attention is spatially localized, but temporally unrestricted. F_8 is inactive for the temporal attention.

Transformer Decoder. The decoder architecture follows that of Mask2Former [13]: the input queries are iteratively refined over multiple layers. In each layer, the queries first cross-attend to their respective masked video features, then self-attend to each other, followed by feed-forward layers.

3.5. Inference

To infer on videos with arbitrary length, we split the input video into clips of length T_{clip} with an overlap of T_{ov} between successive clips. Object tracks are associated across clips based on their mask IoU in the overlapping frames. For the VOS tasks, the object queries for an intermediate clip are initialized by using the predicted masks in the overlapping frames from the previous clip as a pseudo-ground-truth. For VPS, we average the semantic segmentation logits in the overlapping frames. Our approach is thus *near-online* because the time delay in obtaining the output for a given frame is at most $T_{\text{clip}} - T_{\text{ov}} - 1$ (except for the very first clip in the video).

Table 1. Results for Video Instance Segmentation (VIS) on the YouTube-VIS 2021 [73] and OVIS [55] validation sets.

Method	Backbone	Shared Model	YouTube-VIS 2021					OVIS				
			AP	AP50	AP75	AR1	AR10	AP	AP50	AP75	AR1	AR10
Mask2Former-VIS [12]	R-50	✗	40.6	60.9	41.8	-	-	-	-	-	-	-
IDOL [70]	R-50	✗	43.9	68.0	49.6	38.0	50.9	30.2	51.3	30.0	15.0	37.5
MinVIS [28]	R-50	✗	44.2	66.0	48.1	39.2	51.7	25.0	45.5	24.0	13.9	29.7
VITA [26]	R-50	✗	45.7	67.4	49.5	**40.9**	53.6	19.6	41.2	17.4	11.7	26.0
Ours (TarViS)	R-50	✓	**48.3**	**69.6**	**53.2**	40.5	**55.9**	**31.1**	**52.5**	**30.4**	**15.9**	**39.9**
Mask2Former-VIS [12]	Swin-T	✗	45.9	68.7	50.7	-	-	-	-	-	-	-
Ours (TarViS)	Swin-T	✓	**50.9**	**71.6**	**56.6**	42.2	57.2	34.0	55.0	34.4	16.1	40.9
IDOL [70]	Swin-L	✗	56.1	80.8	63.5	45.0	60.1	42.6	65.7	**45.2**	17.9	49.6
VITA [26]	Swin-L	✗	57.5	80.6	61.0	47.7	62.6	27.7	51.9	24.9	14.9	33.0
Ours (TarViS)	Swin-L	✓	**60.2**	**81.4**	**67.6**	**47.6**	**64.8**	**43.2**	**67.8**	44.6	**18.0**	**50.4**

Table 2. Video Panoptic Segmentation (VPS) results for validation sets of KITTI-STEP [67], CityscapesVPS [35] and VIPSeg [45].

Method	Shared Model	KITTI-STEP			CityscapesVPS			VIPSeg			
		STQ	AQ	SQ	VPQ	VPQ^{Th}	VPQ^{St}	VPQ	VPQ^{Th}	VPQ^{St}	STQ
Mask Propagation [67]	✗	0.67	0.63	0.71	-	-	-	-	-	-	
Track [35]	✗	-	-	-	55.9	43.7	64.8	-	-	-	
VPSNet [35]	✗	0.56	0.52	0.61	57.0	44.7	66.0	14.0	14.0	14.2	20.8
VPSNet-SiamTrack [68]	✗	-	-	-	57.3	44.7	66.4	17.2	17.3	17.3	21.1
VIP-Deeplab [56]	✗	-	-	-	**63.1**	**49.5**	**73.0**	16.0	12.3	18.2	22.0
Clip-PanoFCN [45]	✗	-	-	-	-	-	-	22.9	25.0	20.8	31.5
Ours (TarViS - R-50)	✓	0.70	0.70	0.69	53.3	35.9	66.0	33.5	39.2	28.5	43.1
Ours (TarViS - Swin-T)	✓	0.71	0.71	0.70	58.0	42.9	69.0	35.8	42.7	29.7	45.3
Ours (TarViS - Swin-L)	✓	**0.72**	**0.72**	**0.73**	58.9	43.7	69.9	**48.0**	**58.2**	**39.0**	**52.9**

4. Experiments

4.1. Implementation Details

Our Temporal Neck contains 6 layers of Deformable and Temporal Attention. We pretrain our model for 500k iterations with batch size 32 on pseudo-video clips generated by applying on-the-fly augmentations to images from COCO [43], ADE20k [78], Mapillary [49] and Cityscapes [18]. These samples are either trained for VPS, VIS, VOS or PET. This is followed by fine-tuning for 90k iterations jointly on samples from YouTube-VIS [73], OVIS [55], KITTI-STEP [67], CityscapesVPS [35], VIPSeg [45], DAVIS [54] and BURST [3]. For each of the four query types (Q_{sem}, Q_{inst}, Q_{obj}, Q_{bg}) discussed in Sec. 3, we employ a learned query embedding, which is used when computing the $\texttt{Key}^T\texttt{Query}$ affinity matrix for multi-head attention inside the decoder. We refer to the supplementary for more details.

4.2. Benchmark Results

All results are computed with a single, jointly trained model which performs different tasks by simply providing the corresponding query set at run-time.

Video Instance Segmentation (VIS). We evaluate on two benchmarks: (1) YouTube-VIS 2021 [73] which covers 40 object classes and contains 2985/421 videos for training/validation, and (2) OVIS [55] which covers 25 object classes. It contains 607/140 videos for training/validation which are comparatively longer and more occluded. The AP scores for both are reported in Tab. 1. For all three backbones, TarViS achieves state-of-the-art results for both benchmarks even though other methods are trained separately per benchmark whereas we use a single model. On YouTube-VIS, TarViS achieves 48.3 AP with a ResNet-50 backbone compared to the 45.7 achieved by VITA [26]. With Swin-L, we achieve 60.2 AP which is also higher than the 57.5 by VITA. On OVIS with ResNet-50, our 31.1 AP is higher than the 30.2 for IDOL [70], and with Swin-L, TarViS (43.2 AP) out-performs the current state-of-the-art IDOL (42.6 AP).

Video Panoptic Segmentation (VPS). We evaluate VPS on three datasets: (1) KITTI-STEP [67], which contains 12/9 lengthy driving scene videos for training/validation with 19 semantic classes (2 *thing* and 17 *stuff* classes), (2) CityscapesVPS [35], which contains 50 short driving scene clips, each with 6 annotated frames, and (3) VIPSeg [45], which is a larger dataset with 2806/343 in-the-wild videos for training/validation and 124 semantic classes. The results

are reported in Tab. 2. For KITTI-STEP, TarViS achieves 70% STQ with a ResNet-50 backbone which is better than all existing approaches. The performance further improves to 72% with Swin-L. For CityscapesVPS, TarViS achieves 58.9 VPQ which is higher than all other methods except VIP-Deeplab [56] (63.1). However, VIP-Deeplab performs monocular depth estimation for additional guidance, and therefore requires ground-truth depth-maps for training.

For VIPSeg, TarViS out-performs existing approaches by a significant margin. With a ResNet-50 backbone, our 33.5 VPQ is 10.6% higher than the 22.9 by Clip-PanoFCN [45]. With a Swin-Large backbone, TarViS achieves 48.0 VPQ which is more than double that of Clip-PanoFCN (22.9). Note that VIP-Deeplab performs significantly worse for VIPSeg (16.0 VPQ), showing that TarViS generalizes better across benchmarks. Finally, we note that larger backbones results in significant performance gains for datasets with in-the-wild internet videos as in VIPSeg, but for specialized driving scene datasets (*e.g.* KITTI-STEP and Cityscapes-VPS), the improvements are much smaller.

Video Object Segmentation (VOS). We evaluate VOS on the DAVIS 2017 [54] dataset, which contains 60/30 YouTube videos for training/validation. The results in Tab. 3 show that TarViS achieves 85.3 $\mathcal{J}\&\mathcal{F}$ which is higher than all existing methods except STCN [17] (85.4) and XMem [15] (86.2). As mentioned in Sec. 3.3, encoding objects as queries incurs a loss of fine-grained information, which is detrimental to performance. On the other hand, space-time correspondence (STC) based approaches learn pixel-to-pixel affinities between frames, which enables them to propagate fine-grained object appearance information. We note, however, that TarViS is the first method not based on the STC paradigm which achieves this level is performance (85.3 $\mathcal{J}\&\mathcal{F}$), out-performing several STC-based methods as well as all non-STC based methods *e.g.* HODOR [2] (81.5) and UNICORN [72] (70.6).

Point Exemplar-guided Tracking (PET). PET is evaluated on the recently introduced BURST benchmark [3] which contains 500/1000 diverse videos for training/validation with indoor, outdoor, driving and scripted movie scenes. It is a constrained version of VOS which only provides the point coordinates of the object mask centroid instead of the full mask. Tab. 3 shows that existing methods can only tackle either VOS or PET. To verify this, we tried adapting STCN [17] for PET by training it with point masks, but the training did not converge. By contrast, TarViS encodes objects into queries, which enables it to tackle both tasks with a single model since the object guidance (point or mask) is abstracted behind the `EncodeObjects`$(\cdot)$ function.

TarViS achieves 37.5 HOTA_{all} which is significantly better than the 24.4 achieved by the best performing baseline method which casts PET as a VOS problem by regressing a pseudo-ground-truth mask from the given point, followed by applying a VOS approach (STCN [17]).

Table 3. Results for Mask-guided VOS on DAVIS [54] and Point-guided VOS on BURST [3] ('H' denotes 'HOTA' [44]).

Method	DAVIS (M-VOS)			BURST (P-VOS)		
	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$	H_{all}	H_{com}	H_{unc}
UNICORN∗ [72]	70.6	66.1	75.0	-	-	-
HODOR [2]	81.3	78.4	83.9	-	-	-
STM [51]	81.8	79.2	84.3	-	-	-
CFBI [74]	81.9	79.1	84.6	-	-	-
RMNet [71]	83.5	81.0	86.0	-	-	-
HMMN [61]	84.7	81.9	87.5	-	-	-
MiVOS [16]	84.5	81.7	87.4	-	-	-
AOT [75]	84.9	82.3	87.5	-	-	-
STCN [17]	85.4	82.2	88.6	-	-	-
XMem [15]	**86.2**	**82.9**	**89.5**	-	-	-
Box Tracker [32]	-	-	-	12.7	31.7	7.9
STCN+M2F [13, 17]	-	-	-	24.4	44.0	19.5
Ours (TarViS - R-50)	82.0	78.7	87.0	30.9	43.2	27.8
Ours (TarViS - Swin-T)	82.8	79.6	86.0	36.0	47.7	33.0
Ours (TarViS - Swin-L)	85.3	81.7	88.5	**37.5**	**51.7**	**34.0**

Figure 4. Qualitative results from a single TarViS model for all four tasks. Further results are shown in the supplementary.

4.3. Ablations

We ablate several aspects of our architecture/training under a reduced training setup with batch size 16. Pre-training is done only with COCO for 380k iterations, followed by fine-tuning on a smaller set of video datasets for 80k iterations. The results are presented in Table 4.

Task-specific Training (row 1-2). The first two rows show results for task-specific models. Since the KITTI-STEP [67] dataset is quite small, it was not possible to train a VPS-only variant. Moreover, we train a single model for VOS and PET since both tasks are closely related. We note that the VIS only model performs noticeably worse than the

Table 4. Ablation experiment results. A Swin-T backbone is used for all settings.

	Setting	Training Data					VIS		VPS	VOS	PET
		YTVIS	OVIS	KITTI	DAVIS	BURST	YTVIS (mAP)	OVIS (mAP)	KITTI-STEP (STQ)	DAVIS ($\mathcal{J}\&\mathcal{F}$)	BURST (HOTA_{all})
1.	VIS	✓	✓				49.8	30.0	-	-	-
2.	VOS + PET				✓	✓	-	-	-	81.4	32.8
3.	VIS (FC for CLS)	✓	✓				50.5	31.0	-	-	-
4.	No Temporal Neck	✓	✓	✓	✓	✓	46.8	25.4	0.67	78.0	29.7
5.	No Object Encoder	✓	✓	✓	✓	✓	50.1	32.6	0.66	75.6	25.9
	Final	✓	✓	✓	✓	✓	51.1	31.7	0.69	81.5	29.2

final setting (49.8 vs. 51.1 mAP on YouTube-VIS), which indicates that the combination of additional training data and multi-task supervision in the final model is beneficial for VIS. For VOS on DAVIS [54], the task-specific model achieves similar performance to the final setting (81.4 vs. 81.5), but for PET the task-specific variant performs noticeably better (32.8 vs. 29.2). We conclude that although the task-specific model performs better on PET, multi-task training generally improves performance across tasks.

Semantic Queries for VIS (row 3). As mentioned in Sec. 3.1, TarViS represents object classes as dynamic query inputs to the network (Q_{sem}). We ablate this by modifying our network to work for only VIS in the same way as existing instance segmentation methods [13, 66], *i.e.* using instance queries Q_{inst} in conjunction with a linear layer (separate for each dataset) for classification. The results show that this task-specific architecture is better suited for VIS since it achieves 50.5 mAP on YouTube-VIS vs. 49.8 for the setting in row 1 which is trained on similar data. However, our final multi-task model still performs slightly better than the current setting on both YouTube-VIS and OVIS.

Temporal Neck (row 4). We validate the effectiveness of our novel Temporal Neck (Sec. 3.4) by training a model with a simpler neck that contains only Deformable Attention layers [79], similar to Mask2Former [13], *i.e.* there is no feature interaction across frames. The results show significant performance degradation for YouTube-VIS (46.8 bs. 51.1), OVIS (25.4 vs. 31.7) and DAVIS (78.0 vs. 81.5). KITTI-STEP is impacted comparatively less (0.67 vs 0.69), whereas performance for PET actually shows a slight increase (29.7 vs. 29.2) Overall, however, we conclude that inter-frame feature interactions enabled by the Temporal Neck are beneficial for down-stream tasks.

Object Encoder for VOS/PET (row 5). As discussed in Sec. 3.3, we encode objects from their given first-frame mask/point as object queries Q_{obj} using an Object Encoder. We validate the efficacy of this module by training a simpler model which initializes object queries by average pooling the image features inside the mask for VOS, and indexing the feature map at the given point coordinates for PET. This model performs significantly worse than the final setting on both DAVIS (75.6 vs. 81.5) and BURST (25.9 vs. 29.2), indicating that the quality of the encoded object query has a profound impact on performance.

Figure 5. TarViS performing VIS and VOS in a single forward pass. We provide the mask for the dragon on the left, and the semantic query for the 'person' class.

5. Discussion

Limitations. As noted above, training on multiple datasets/tasks does not necessarily improve performance on all benchmarks. For VOS, the model exhibits class bias since it sometimes fails to track unusual objects which were not seen during training.

Future Outlook. We jointly trained TarViS for four different tasks to validate its generalization capability. The architecture can, however, tackle any video segmentation task for which the targets can be encoded as queries. The recent emergence of joint language-vision models [42, 57, 58] thus makes it possible to perform multi-object segmentation based on a text prompt if the latter can be encoded as a target query using a language encoder [19]. Another interesting possibility is that TarViS could be applied to multiple tasks *in the same forward pass* by simply concatenating the task-specific queries. Fig. 5 offers a promising outlook for this; it shows our model's output for a video clip from a

popular TV series where we perform VIS and VOS simultaneously by providing the semantic query for the 'person' class (from YouTube-VIS [73]), and the VOS-based object queries for the dragon by annotating its first frame mask, *i.e.* $Q_{\text{in}} = \texttt{concat}(Q_{\text{sem}}, Q_{\text{inst}}, Q_{\text{obj}}, Q_{\text{bg}})$. TarViS successfully segments all four persons in the scene (VIS) and the dragon (VOS), even though our model was never trained to simultaneously tackle both tasks in a single forward pass.

6. Conclusion

We presented TarViS: a novel, unified approach for tackling any task requiring pixel-precise segmentation of a set of *targets* in video. We adopt a generalized paradigm where the task-specific targets are encoded into a set of *queries* which are then input to our network together with the video features. The network is trained to produce segmentation masks for each target entity, but is inherently agnostic to the task-specific definition of these targets. To demonstrate the effectiveness of our approach, we applied it to four different video segmentation tasks (VIS, VPS, VOS, PET). We showed that a single TarViS model can be jointly trained for all tasks, and during inference can hot-swap between tasks without any task-specific fine-tuning. Our model achieved state-of-the-art performance on five benchmarks (YouTube-VIS, OVIS, KITTI-STEP, VIPSeg and BURST) and has multiple, promising directions for future work.

Acknowledgments. This project was partially funded by ERC Consolidator Grant DeeVise (ERC-2017-COG-773161). We thank Istvan Sarandi, Christian Schmidt and Alexey Nekarsov for constructive feedback. Compute resources were granted by RWTH Aachen under project ID `supp0003`, and by the Gauss Centre for Supercomputing e.V. through the John von Neumann Institute for Computing on the GCS Supercomputer JUWELS at Jülich Supercomputing Centre.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. In *Arxiv*, 2022. 1, 2

[2] 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. 2, 4, 7, 1

[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. 2, 3, 5, 6, 7, 1

[4] Ali Athar, Sabarinath Mahadevan, Aljosa Osep, Laura Leal-Taixé, and Bastian Leibe. Stem-seg: Spatio-temporal embeddings for instance segmentation in videos. In *ECCV*, 2020. 2

[5] Ali Athar, Sabarinath Mahadevan, Aljosa Osep, Laura Leal-Taixé, and Bastian Leibe. A single-stage, bottom-up approach for occluded vis using spatio-temporal embeddings. In *ICCV-W*, 2021. 2

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

[7] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *ICML*, 2021. 5

[8] Thomas Brox and Jitendra Malik. Object segmentation by long term analysis of point trajectories. In *ECCV*, 2010. 1

[9] Jiale Cao, Rao Muhammad Anwer, Hisham Cholakkal, Fahad Shahbaz Khan, Yanwei Pang, and Ling Shao. Sipmask: Spatial information preservation for fast image and video instance segmentation. In *ECCV*, 2020. 2

[10] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 2, 4

[11] Richard Caruana. Multitask Learning: A Knowledge-Based Source of Inductive Bias. In *ICML*, 1993. 2

[12] Bowen Cheng, Anwesa Choudhuri, Ishan Misra, Alexander Kirillov, Rohit Girdhar, and Alexander G Schwing. Mask2former for video instance segmentation. In *Arxiv*, 2021. 2, 5, 6

[13] Bowen Cheng, Anwesa Choudhuri, Ishan Misra, Alexander Kirillov, Rohit Girdhar, and Alexander G Schwing. Mask2former for video instance segmentation. In *CVPR*, 2022. 2, 4, 5, 7, 8

[14] Bowen Cheng, Maxwell D Collins, Yukun Zhu, Ting Liu, Thomas S Huang, Hartwig Adam, and Liang-Chieh Chen. Panoptic-deeplab: A simple, strong, and fast baseline for bottom-up panoptic segmentation. In *CVPR*, 2020. 2, 4

[15] Ho Kei Cheng and Alexander G Schwing. Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *ECCV*, 2022. 3, 7

[16] 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. 3, 7

[17] 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, 7

[18] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 6, 1

[19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *NAACL*, 2019. 8

[20] Ahmed Elgammal, David Harwood, and Larry Davis. Non-parametric model for background subtraction. In *ECCV*, 2000. 1

[21] Andreas Ess, Bastian Leibe, Konrad Schindler, and Luc Van Gool. Moving obstacle detection in highly dynamic scenes. In *ICRA*, 2009. 1

[22] Heng Fan, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Hexin Bai, Yong Xu, Chunyuan Liao, and Haibin Ling. Lasot: A high-quality benchmark for large-scale single object tracking. In *CVPR*, 2019. 2

[23] Yang Fu, Linjie Yang, Ding Liu, Thomas S Huang, and Humphrey Shi. Compfeat: Comprehensive feature aggregation for video instance segmentation. In *AAAI*, 2021. 2

[24] Golnaz Ghiasi, Barret Zoph, Ekin D Cubuk, Quoc V Le, and Tsung-Yi Lin. Multi-Task Self-Training for Learning General Representations. In *CVPR*, 2021. 2

[25] Helmut Grabner, Michael Grabner, and Horst Bischof. Real-time tracking via on-line boosting. In *BMVC*, 2006. 1

[26] Miran Heo, Sukjun Hwang, Seoung Wug Oh, Joon-Young Lee, and Seon Joo Kim. Vita: Video instance segmentation via object token association. In *NeurIPS*, 2022. 2, 6

[27] Namdar Homayounfar, Justin Liang, Wei-Chiu Ma, and Raquel Urtasun. Videoclick: Video object segmentation with a single click. *Arxiv*, 2021. 3

[28] De-An Huang, Zhiding Yu, and Anima Anandkumar. Minvis: A minimal video instance segmentation framework without video-based training. In *NeurIPS*, 2022. 2, 6

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

[30] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. Perceiver io: A general architecture for structured inputs & outputs. *ICLR*, 2022. 1, 2

[31] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *ICML*, 2021. 1, 2

[32] Arne Hoffhues Jonathon Luiten. Trackeval. https://github.com/JonathonLuiten/TrackEval, 2020. 7

[33] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *IJCV*, 1(4):321–331, 1988. 1

[34] Lei Ke, Xia Li, Martin Danelljan, Yu-Wing Tai, Chi-Keung Tang, and Fisher Yu. Prototypical cross-attention networks for multiple object tracking and segmentation. In *NeurIPS*, 2021. 2

[35] Dahun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Video panoptic segmentation. In *CVPR*, 2020. 2, 4, 6

[36] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *CVPR*, 2019. 2

[37] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. In *CVPR*, 2020. 1

[38] Iasonas Kokkinos. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *CVPR*, 2017. 2

[39] Alexander Kolesnikov, André Susano Pinto, Lucas Beyer, Xiaohua Zhai, Jeremiah Harmsen, and Neil Houlsby. Uvim: A unified modeling approach for vision with learned guiding codes. In *NeurIPS*, 2022. 1, 2

[40] Dieter Koller, Joseph Weber, and Jitendra Malik. Robust multiple car tracking with occlusion reasoning. In *ECCV*, 1994. 1

[41] Bastian Leibe, Konrad Schindler, Nico Cornelis, and Luc Van Gool. Coupled object detection and tracking from static cameras and moving vehicles. *PAMI*, 2008. 1

[42] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *CVPR*, 2022. 8

[43] 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. 6, 1

[44] 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. *IJCV*, 2020. 7

[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. 6, 7, 2

[46] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking. In *arXiv:1603.00831*, 2016. 2

[47] Ishan Misra, Rohit Girdhar, and Armand Joulin. An end-to-end transformer model for 3d object detection. In *ICCV*, 2021. 2

[48] Matthias Muller, Adel Bibi, Silvio Giancola, Salman Alsubaihi, and Bernard Ghanem. Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In *ECCV*, 2018. 2

[49] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *ICCV*, 2017. 6, 1

[50] Peter Ochs, Jitendra Malik, and Thomas Brox. Segmentation of moving objects by long term video analysis. *PAMI*, 2013. 1

[51] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *ICCV*, 2019. 3, 5, 7

[52] Kilian Pfeiffer, Alexander Hermans, István Sárándi, Mark Weber, and Bastian Leibe. Visual person understanding through multi-task and multi-dataset learning. In *GCPR*, 2019. 2

[53] Hamed Pirsiavash, Deva Ramanan, and Charless C Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR*, pages 1201–1208. IEEE, 2011. 1

[54] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. In *Arxiv*, 2017. 2, 6, 7, 8, 1

[55] 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: A benchmark. In *IJCV*, 2022. 2, 6

[56] 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, 4, 6, 7

[57] 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 *ICML*, 2021. 8

[58] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *ICML*, 2021. 8

[59] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al. A generalist agent. In *Arxiv*, 2022. 1, 2

[60] Sebastian Ruder. An Overview of Multi-Task Learning in Deep Neural Networks. *arXiv:1706.05098*, 2017. 2

[61] Hongje Seong, Seoung Wug Oh, Joon-Young Lee, Seongwon Lee, Suhyeon Lee, and Euntai Kim. Hierarchical memory matching network for video object segmentation. In *ICCV*, 2021. 3, 7

[62] C. Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. In *CVPR*, 1999. 1

[63] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017. 1

[64] 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. 3

[65] Paul Voigtlaender, Michael Krause, Aljosa Osep, Jonathon Luiten, Berin Balachandar Gnana Sekar, Andreas Geiger, and Bastian Leibe. Mots: Multi-object tracking and segmentation. In *CVPR*, 2019. 2

[66] 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, 4, 8

[67] Mark Weber, Jun Xie, Maxwell Collins, Yukun Zhu, Paul Voigtlaender, Hartwig Adam, Bradley Green, Andreas Geiger, Bastian Leibe, Daniel Cremers, Aljoša Ošep, Laura Leal-Taixé, and Liang-Chieh Chen. STEP: Segmenting and tracking every pixel. In *NeurIPS*, 2021. 2, 4, 6, 7

[68] Sanghyun Woo, Dahun Kim, Joon-Young Lee, and In So Kweon. Learning to associate every segment for video panoptic segmentation. In *CVPR*, 2021. 6

[69] Junfeng Wu, Yi Jiang, Wenqing Zhang, Xiang Bai, and Song Bai. Seqformer: a frustratingly simple model for video instance segmentation. In *ECCV*, 2022. 2

[70] Junfeng Wu, Qihao Liu, Yi Jiang, Song Bai, Alan Yuille, and Xiang Bai. In defense of online models for video instance segmentation. In *ECCV*, 2022. 2, 6

[71] Haozhe Xie, Hongxun Yao, Shangchen Zhou, Shengping Zhang, and Wenxiu Sun. Efficient regional memory network for video object segmentation. In *CVPR*, 2021. 3, 7

[72] Bin Yan, Yi Jiang, Peize Sun, Dong Wang, Zehuan Yuan, Ping Luo, and Huchuan Lu. Towards grand unification of object tracking. In *ECCV*, 2022. 2, 7

[73] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *ICCV*, 2019. 2, 6, 9

[74] Zongxin Yang, Yunchao Wei, and Yi Yang. Collaborative video object segmentation by foreground-background integration. In *ECCV*, 2020. 3, 7

[75] Zongxin Yang, Yunchao Wei, and Yi Yang. Associating objects with transformers for video object segmentation. In *NeurIPS*, 2021. 3, 7

[76] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *CVPR*, 2020. 2

[77] Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *CVPR*, 2018. 2

[78] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017. 6, 1

[79] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICML*, 2020. 2, 4, 5, 8

Supplementary Material

S1. Extended VOS/PET Ablations

Extended ablation results are given in Table S1 and discussed below. Note that similar to the ablations in Sec. 4 of the main text, we use a smaller training schedule with fewer datasets for these experiments.

Table S1. Extended ablation results for VOS and PET tasks on DAVIS [54] and BURST [3] benchmarks, respectively.

Setting	VOS ($\mathcal{J}\&\mathcal{F}$)	PET (HOTA_{all})
Without Q_{bg}	78.0	25.9
$\lvert Q_{\text{obj}}\rvert = q_0 = 1$	80.6	28.2
Final	81.5	29.2

Background Queries (row 1). We stated in the main text that we model the non-object pixels in the input video using background queries for the VOS and PET task. We ablate this design decision by training TarViS without this sort of background modeling, *i.e*. for both VOS and PET tasks, the input set of queries contains only the object queries Q_{obj}. This reduces the $\mathcal{J}\&\mathcal{F}$ score for VOS from 81.5 to 78.0, and the HOTA_{all} score for PET from 29.2 to 25.9. Thus, we conclude that background modelling has a noticeable, positive impact on prediction quality.

Number of Object Queries (row 2). We mentioned in the main text that we modify the approach adopted by HODOR [2] for VOS by using multiple (q_0) object queries to represent a single target object. We ablate this by training our model using $q_0 = 1$ (in the final setting we use $q_0 = 4$). We see that this causes the performance on DAVIS to reduce from 81.5 to 80.6, and that on BURST from 29.2 to 28.2. Note that $q_0 = 1$ for PET even for the final setting, but because PET inference over lengthy videos involves VOS-style mask-guidance, the choice of q_0 for VOS effects performance for PET as well.

S2. Implementation Details

Several details related to the training and inference setup which were omitted from the main paper are given below.

Setup. We train our models on 32 Nvidia A100 GPUs with a batch size of 32 with clips of 3 frames. The pretraining takes 2-3 days (depending on the backbone) and the finetuning takes 10-16 hours. An AdamW optimizer is used, and the learning rate is 10^4 at the start of training followed by two step decays with a factor of 0.1 each. Inference is performed on a single RTX 3090 and runs at approximately 10fps with a Swin-T backbone (there is some variation for different datasets due to the varying input image resolution).

Table S2. Loss functions used for mask prediction for different targets. BCE: Binary cross-entropy, MCE: Multi-class cross-entropy, DICE: soft IoU loss

Target Type	Task	Loss	Sparse
Instance	VIS	DICE + BCE	✓
Semantic Class		MCE	✓
Instance	VPS	DICE + BCE	✓
Semantic Class		MCE	✗
Object	VOS/PET	DICE + BCE	✓

The clip length during inference is usually 12 with a frame overlap of 6 frames.

Loss Supervision. Table S2 shows the type of loss function applied for mask regression for different tasks. Generally, the supervision signal is a combination of DICE and cross-entropy losses. For instances/objects we apply per-pixel binary cross-entropy whereas for semantic segmentation (where multiple classes compete for each pixel), we apply a multi-class cross-entropy loss. 'Sparse' means that the loss is not applied to every pixel in the mask, but rather only to a subset of sampled pixels which contain a certain fraction of hard negatives and other randomly sampled points. This type of supervision strategy was proposed by Kirillov *et al*. [37].

Pretraining. We pretrain on synthetic video samples generated by applying random, on-the-fly augmentations from the following image-level datasets: COCO [43], ADE20k [78], Mapillary [49], Cityscapes [18]. Since these datasets provide panoptic annotations, we can train the data samples as any of the four target tasks (VPS, VIS, VOS, PET) *e.g*. to train for VOS/PET, we assume that the first-frame mask/point is available for a random sub-set of ground-truth objects and ignore the class labels. The task weights for pretraining are given in Table S3.

Table S3. Task weights during pretraining stage.

Task	VPS	VIS	VOS	PET
Weight	0.3	0.3	0.28	0.12

Video Finetuning. The finetuning is done on actual video datasets for all four tasks. The sampling weights for each dataset/task are given in Table S4. Note that data samples from DAVIS [54] and BURST [3] can be trained for both VOS and PET.

Point Exemplar-guided Tracking Inference. As mentioned in Sec. 3 of the main text, the PET task is tackled using the same workflow as for VOS *i.e*. the target objects

Table S4. Dataset weightage during video finetuning.

Dataset	Task	Weight
KITTI-STEP [67]	VPS	0.075
CityscapesVPS [35]	VPS	0.075
VIPSeg [45]	VPS	0.15
YouTube-VIS [73]	VIS	0.225
OVIS [55]	VIS	0.225
DAVIS [54]	VOS/PET	0.05
BURST [3]	VOS/PET	0.2

are encoded as object queries using the Object Encoder. An additional detail about inference on arbitrary length video sequences which is not mentioned in the main text is as follows: the point $\rightarrow$ object query regression is only used for the first clip in which the object appears. For subsequent clips, we have access to the dense mask predictions for that object from our model. Hence, for subsequent clips, we regress the object query from the previous mask predictions (as we do for VOS).

S3. Query Visualization

To gain some insight into the feature representation learned by TarViS for different targets, we provide visualizations of the target queries for various tasks and input video clips in Fig. S1,S2,S3. The setup is as follows: for each video clip, we run inference twice: (1) as VIS where the targets are all instances belonging to the 40 object classes from YouTube-VIS [73], and (2) as VOS by providing the first-frame mask for the objects. We deliberately used videos where the set of set of ground-truth objects would be the same for both tasks. The plot on the right visualizes the union of the target query set for both runs by projecting them from 256 dimensions down to 2 using PCA. The image tile on the left shows our model's predicted masks for the target objects (the prediction quality for these video is very good for both VIS and VOS, so we choose one set of results arbitrarily).

For ease of understanding, we use fixed colors for semantic and background queries (as indicated in the plot legend). For the object queries (VOS) and instance queries (VIS), the color of the query point is consistent with the color of the object mask in the image tile. Note that for VOS we used $q_o = 4$ object queries per target, hence there are 4 hollow diamond shaped points per object.

Though not all aspects of these plots are intuitively explainable, we offer some speculative intuition as listed below:

- The internal representation for a given object is generally consistent across tasks. As an example, consider the horse and person targets in Fig. S1: we note that the green query points (person) are close to each other for both VIS and VOS. Likewise the blue query points (horse) follow the same behavior.
- The network devotes a large portion of the feature space for instances/objects, and relatively less for the various semantic classes. As seen in all three plots, the semantic queries are tightly clustered together, whereas the instance/object queries are spread out over a larger span of the feature space.

Iterative Evolution of Feature Representation. Fig. S4 shows a side-by-side visualization of how the query feature representation evolves inside the transformer decoder as it iteratively refined the queries using multiple attention layers. The plot on the left shows the queries at the 'zeroth' layer (*i.e.* prior to any interaction with the video features), and the plot on the right shows the final output queries from the last layer (these are identical to the plot in Fig. S1 except for the axes range). We note that the distance between the queries for the two objects increases after refinement, and that the semantic queries are also slightly more spaced out after refinement.

S4. Qualitative Results

The following figures show qualitative results for the different tasks. VIS on YouTube-VIS (Fig. S5,S6,S7) and OVIS (Fig. S8,S9,S10), VPS on KITTI-STEP (Fig. S11,S12,S13), VOS on DAVIS (Fig. S14,S15,S16), and PET on BURST (Fig. S17,S18,S19). One can see that TarViS is able to segment a broad range of objects depending on the target queries and overall is good at assigning consistent IDs. Fig. S20 shows an example of a failure case with several ID switches. Given that we run inference on short overlapping clips, once an ID switch has been made, we cannot recover the original ID. In the example, it seems that TarViS is not able to track the elephant while they are turning around, even though before and after the turn they are assigned consistent IDs. Given that we also train on similar short clips, it is not surprising that TarViS struggles here and we could potentially improve this by looking into other training schemes that span longer clips.

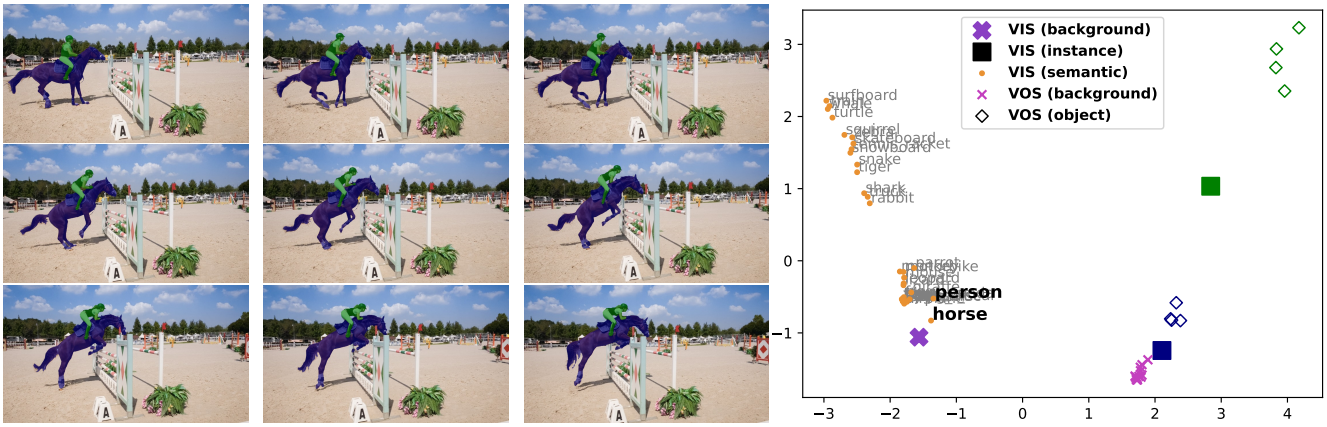


Figure S1. Target query visualization for the 'horsejump-high' sequence in DAVIS.

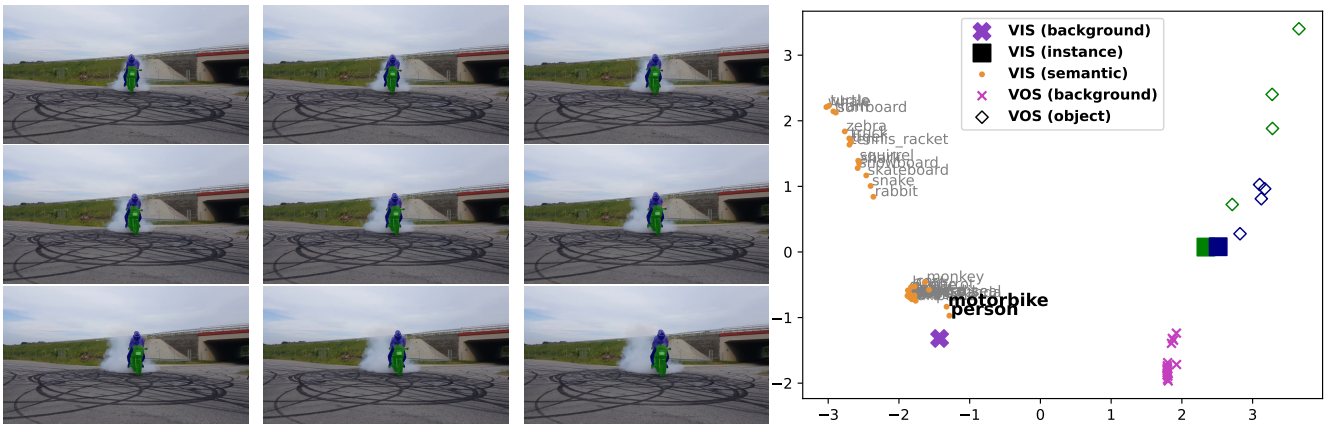


Figure S2. Target query visualization for the 'mbike-trick' sequence in DAVIS.

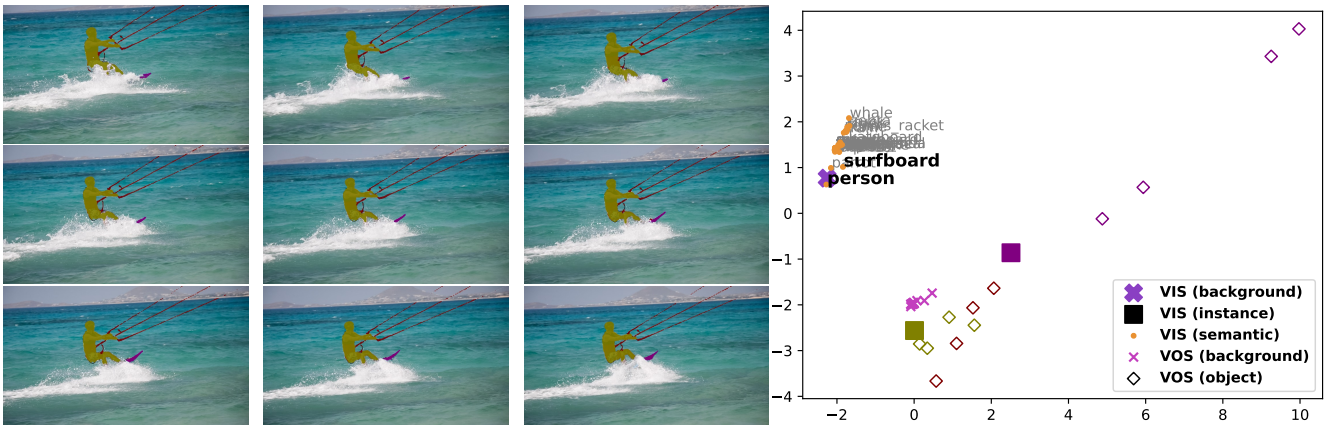
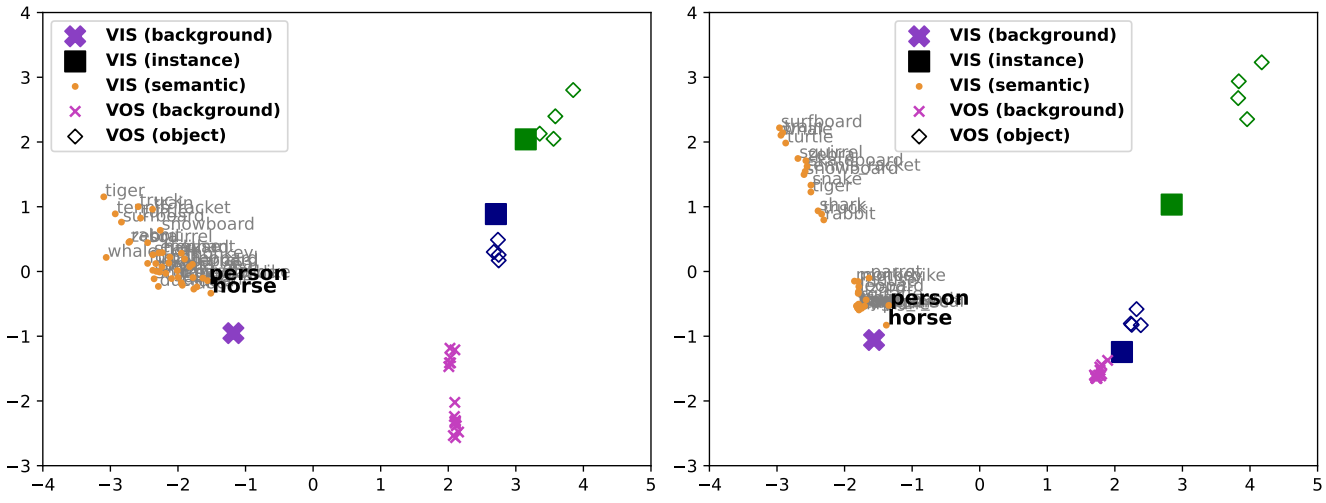


Figure S3. Target query visualization for the 'kitesurf' sequence in DAVIS.



(a) First layer queries.

(b) Last layer queries.

Figure S4. Evolution of the different queries from the first layer to the last layer of the transformer decoder. Queries correspond to the 'horsejump-high' video from DAVIS as shown in Figure S1

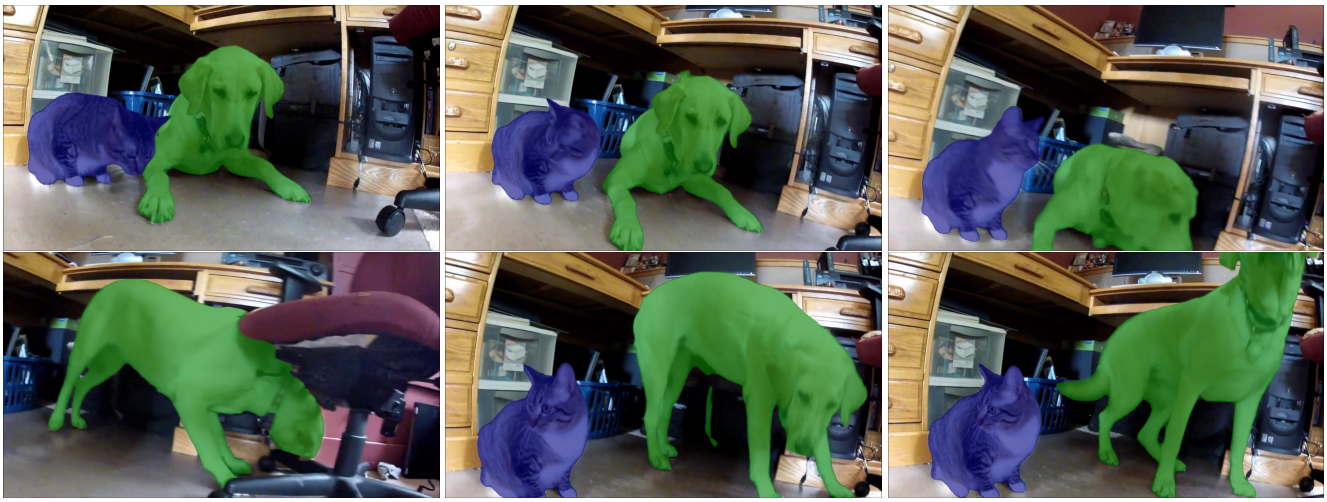


Figure S5. VIS on a YTVIS sequence showing a cat and a dog.

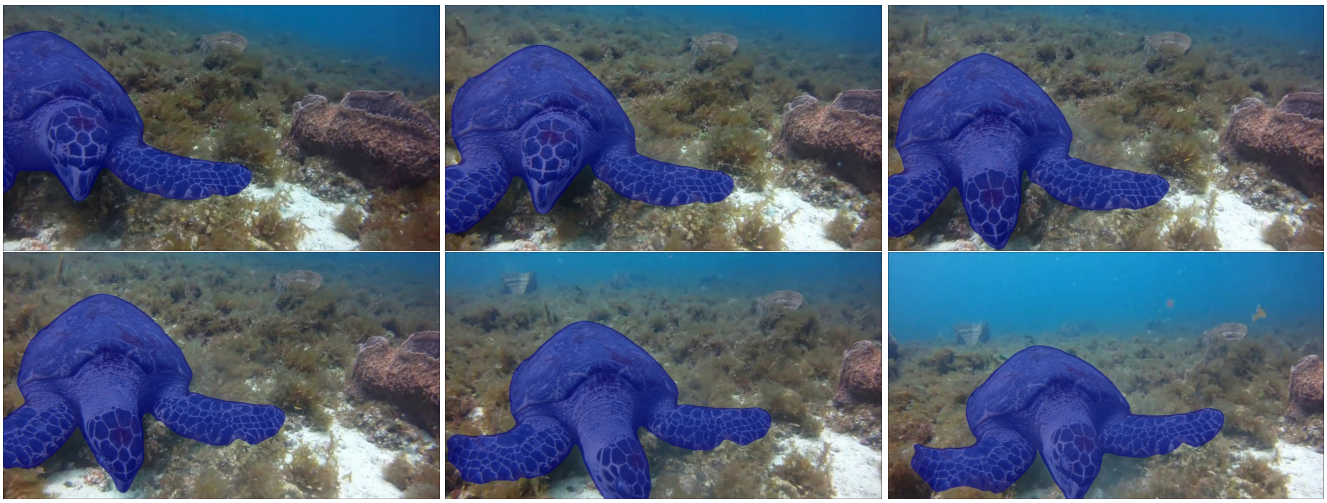


Figure S6. VIS on a YTVIS sequence showing a turtle.

Figure S7. VIS on a YTVIS sequence showing a man and a lizard.

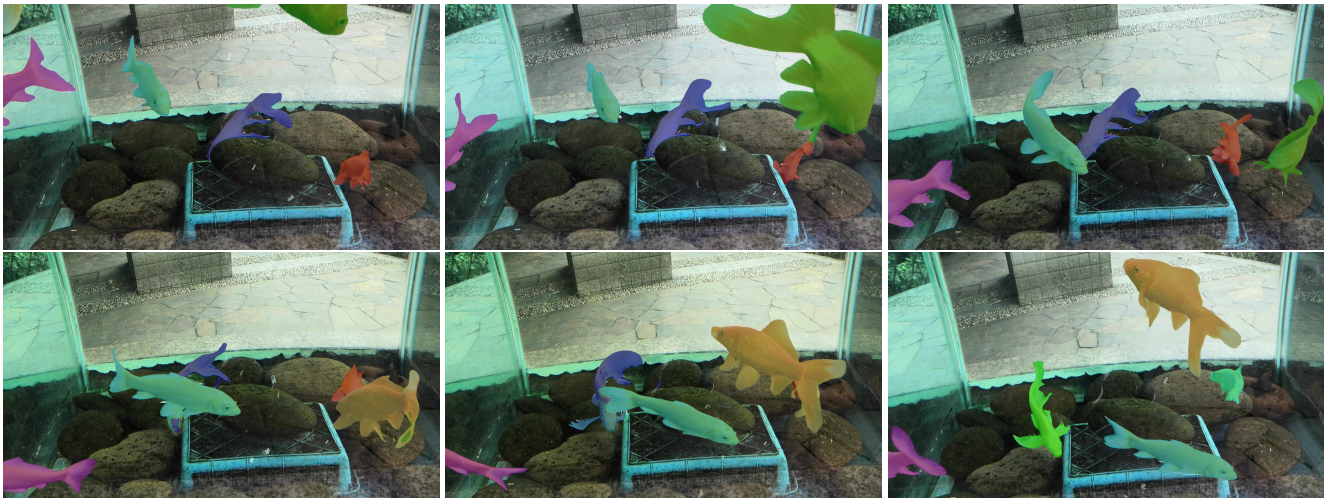


Figure S8. VIS on an OVIS sequence showing an aquarium with fish.

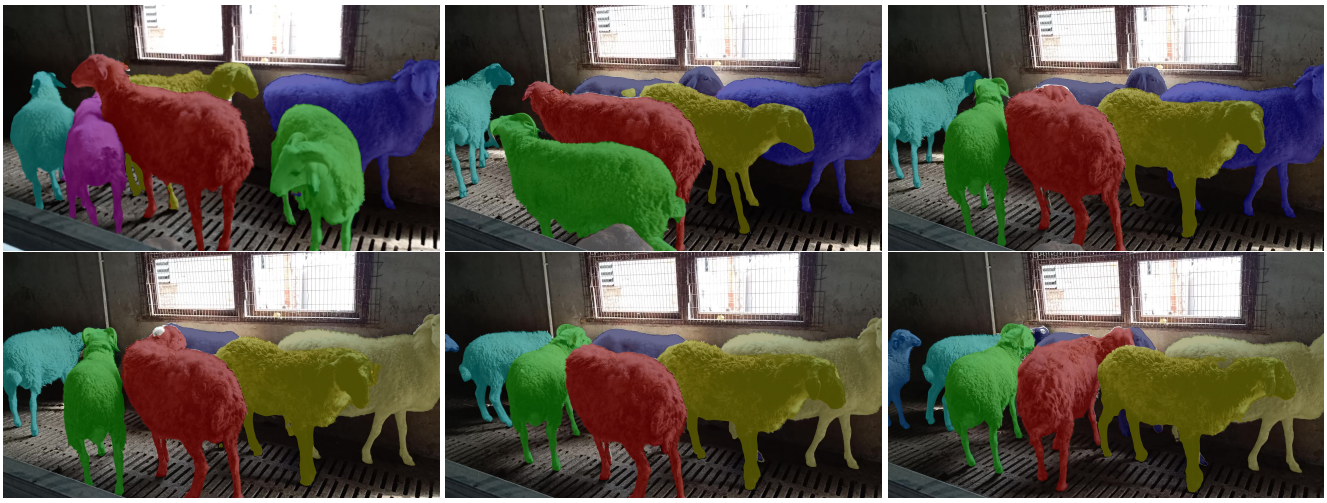


Figure S9. VIS on an OVIS sequence showing several sheep.

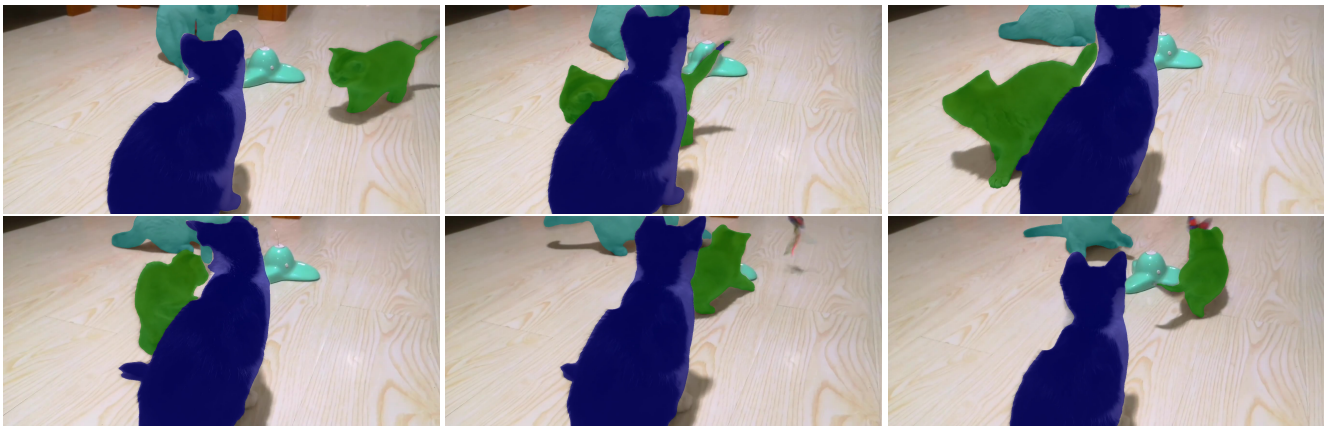


Figure S10. VIS on an OVIS sequence showing three cats.

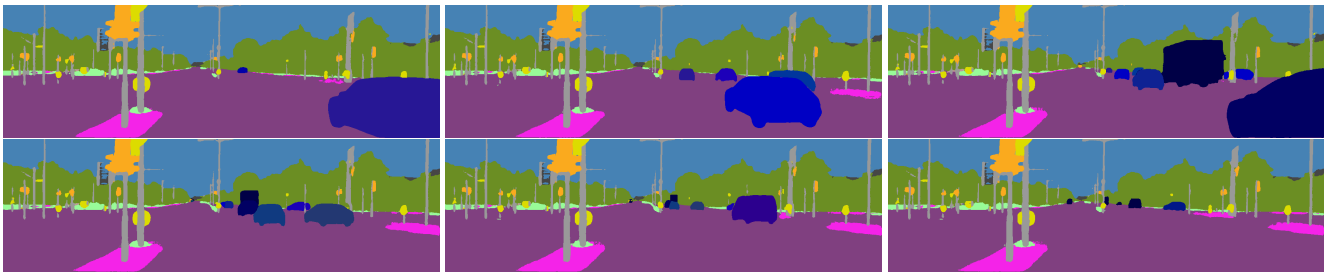


Figure S11. VPS on a KITTI STEP sequence showing a busy intersection.

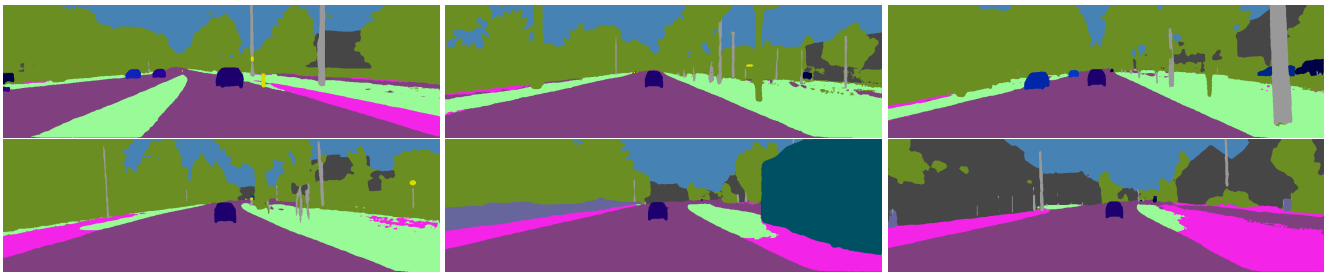


Figure S12. VPS on a KITTI STEP sequence showing how a car is followed for a while.

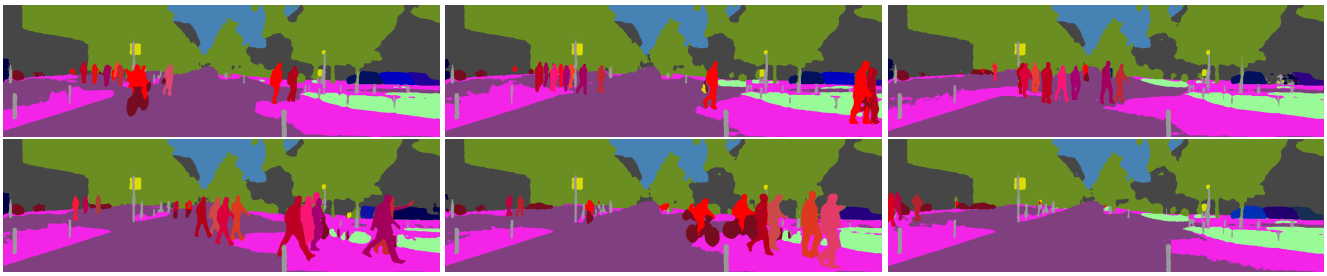


Figure S13. VPS on a KITTI STEP sequence showing a busy pedestrian crossing.

Figure S14. VOS on a DAVIS sequence of a dancer.

Figure S15. VOS on a DAVIS sequence showing several gold fish.

Figure S16. VOS on DAVIS sequence an action movie scene.

Figure S17. PET on a BURST sequence showing three men and a gun.

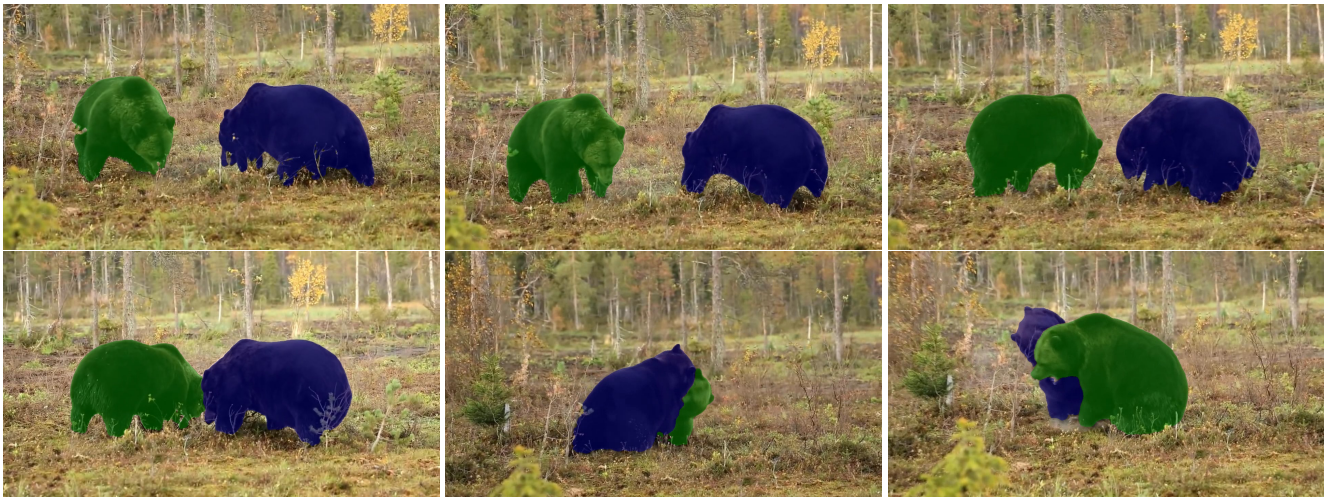


Figure S18. PET on a BURST sequence showing two bears fight, note there is no ID switch.

Figure S19. PET on a BURST sequence showing several cars on a street.

Figure S20. VIS on an OVIS sequence of several elephants and their trainers. This sequence shows that TarVis sometimes has issues with ID switches, especially when the appearance of objects changes, e.g. here the elephants are not tracked consistently while turning around..