Context-Aware Relative Object Queries to Unify Video Instance and Panoptic Segmentation

Anwesa Choudhuri, Girish Chowdhary, Alexander G. Schwing
University of Illinois at Urbana-Champaign
{anwesac2,girishc,aschwing}@illinois.edu

Abstract

Object queries have emerged as a powerful abstraction to generically represent object proposals. However, their use for temporal tasks like video segmentation poses two questions: 1) How to process frames sequentially and propagate object queries seamlessly across frames. Using independent object queries per frame doesn't permit tracking, and requires post-processing. 2) How to produce temporally consistent, yet expressive object queries that model both appearance and position changes. Using the entire video at once doesn't capture position changes and doesn't scale to long videos. As one answer to both questions we propose 'context-aware relative object queries', which are continuously propagated frame-by-frame. They seamlessly track objects and deal with occlusion and re-appearance of objects, without post-processing. Further, we find context-aware relative object queries better capture position changes of objects in motion. We evaluate the proposed approach across three challenging tasks: video instance segmentation, multi-object tracking and segmentation, and video panoptic segmentation. Using the same approach and architecture, we match or surpass state-of-the art results on the diverse and challenging OVIS, Youtube-VIS, Cityscapes-VPS, MOTS 2020 and KITTI-MOTS data.

1. Introduction

Video instance segmentation (VIS) [56] and Multi-Object Tracking and Segmentation (MOTS) combines segmentation and tracking of object instances across frames of a given video, whereas video panoptic segmentation (VPS) requires to also pixel-wise categorize the entire video semantically. These are challenging tasks because objects are occasionally partly or entirely occluded, because the appearance and position of objects change over time, and because objects may leave the camera's field of view only to re-appear at a later time. Addressing these challenges to obtain an accurate method for the aforementioned tasks that works online is important in fields like video editing, autonomous systems, and augmented as well as virtual reality, among others.

Classically, VIS or MOTS treat every frame or clip in a video independently and associate the predictions temporally via a post-processing step [1, 3, 4, 6, 12, 19, 35, 41, 50, 56, 57]. Many of these approaches are based on object proposal generation, that are used in classical detection methods [16, 43]. For image detection and segmentation, recently, query-vectors have been shown to encode accurate object proposals [7, 9, 10]. These query-vector-based object proposals are more flexible than classical object proposals because they are not axis-aligned but rather feature-vector based. Using these accurate query vectors for images, recent methods on VIS [22, 52] adopt the classical method of operating frame-by-frame independently, followed by a post-processing step for associating the query vectors temporally based on their similarity. It remains unclear how the query-vector-based object proposals can be seamlessly extended to the temporal domain.

Some recent transformer-based works [8, 25, 49, 51] use global object queries to process entire videos at once offline, but these methods fail to scale to long videos. However, intuitively, offline approaches should be more accurate than online methods since they operate with a much larger temporal context. Surprisingly, this is not the case. The best methods on VIS [18,22,52] produce query vectors frame-by-frame independently, raising the question why global query vectors fail to accurately represent objects spatio-temporally. We study this carefully and observe that the query vectors are often too reliant on the static spatial positions of objects in a few frames. They hence fail to encode the position changes well. This over-reliance of query vectors on spatial positions has not been observed before in the context of video segmentation. How to address this remains an open question. It also remains unclear how the query-vector-based object proposals can be extended to the temporal domain, while keeping the processing of frames sequential.

In a first attempt to sequentially propagate object queries, the problem of multi-object tracking was studied [38, 44, 58]. These works use separate, distinct queries to represent existing object tracks and new objects. New object queries

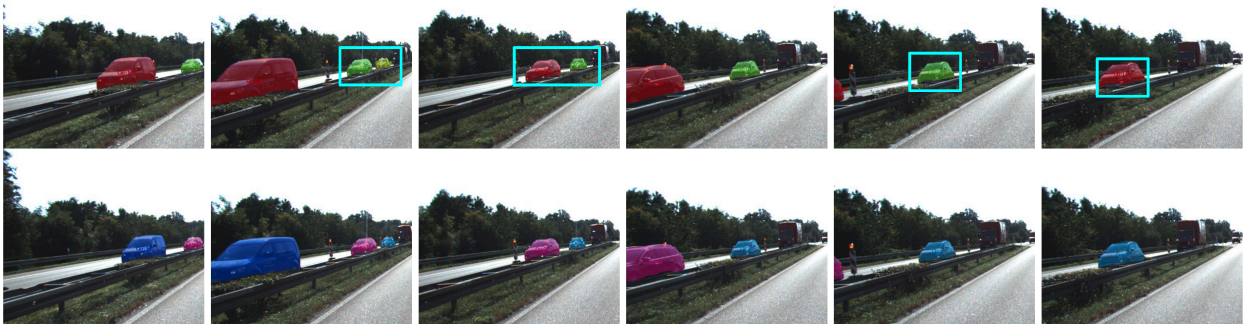

Figure 1. An example from the KITTI-MOTS dataset showing the need for context-aware relative object queries. Object queries from Mask2Former-VIS [8] (top row) heavily rely on the spatial positions of objects, hence can't reason about the position-changes of the cars in the scene. The green car in the first frame is mistaken as the red car when the original red car leaves the scene and the green car takes its spatial position. Similarly, the yellow car is first mistaken as the green car and later as the red car. Cyan boxes in the top row indicate the identity switches. Our method (bottom row) is able to retain identities of the cars despite their significant motion.

are initialized each frame. However, it remains unclear how to seamlessly unify 1) the new object queries, and 2) track queries, while avoiding heuristic post-processing.

Different from prior work, we develop a simple approach which propagates object queries frame-by-frame while simultaneously refining queries via a transformer decoder. Intuitively, the query-vectors in the proposed approach represent all objects of interest in a video without the need to introduce new object queries every frame. Instead, queries are activated if the objects they represent appear in a frame. A continuous refinement of the query-vectors permits to adjust to gradual appearance changes. Their propagation across frames helps them carry long-term temporal information, so that they can seamlessly handle long-term occlusions or absence from the camera field-of-view. While studying why global object queries are sub-optimal at encoding position changes of objects, we observed that the use of absolute position encodings during self- and cross-attention causes the object queries to heavily rely on the object positions in a few frames, as illustrated in the top row of Fig. 1. To address this, we use relative positional encodings (inspired from [13]) instead of absolute encodings. The 'relative object queries' (queries with relative positional encodings) better encode the position changes of objects (bottom row of Fig. 1). Moreover, we use spatio-temporal context (image features from previous frames and the current frame) to modulate the object queries in the transformer decoder, making them 'context-aware.' This permits to more holistically reason about the current frame without losing spatio-temporal details.

We evaluate the proposed approach on the challenging VIS, VPS and MOTS tasks. We outperform methods that reason about an entire video at once by 5% and 11% on the challenging OVIS data using the Resnet-50 and Swin-L backbones. We perform similar to image or clip-based online methods which rely heavily on post-processing. We also outperform or perform close to the state-of-the-art on the Youtube-VIS, Cityscapes VPS, MOTS 2020, and KITTI-MOTS data, demonstrating generalizability of the approach to video segmentation tasks.

2. Related Work

2.1. Video Instance Segmentation

Video instance segmentation (VIS) was proposed by Yang et al. [56] who also introduced the Youtube-VIS datasets. More recently, the occluded video instance segmentation (OVIS) dataset [41] increased the difficulty of this task. Existing VIS approaches can be broadly categorized into online and offline methods.

Online methods. They either operate frame-by-frame [6, 22, 41, 52, 56] or process short, sequential, possibly overlapping clips [1, 3, 5, 57]. In both cases, the local results for frames or clips are merged via post-processing, often involving heuristics which lead to error-prone results. For example, VIS is carried out by first segmenting objects in every frame using available instance segmentation methods (e.g., [9, 16]), sometimes with an added contrastive or temporal loss [52], and then associating the objects or queries to generate time-consistent identities [6,22,41,52,56]. In contrast, InsPro [15] propagates instance queries along with traditional region-based proposals frame-by-frame, the latter being arguably redundant because queries are more flexible object proposals.

Offline Methods. They process the entire video at once. Recently, transformer-based methods [8, 49, 51] have been proposed that perform the VIS task in a single step. IFC [25] includes communication between frames in a transformer encoder. SeqFormer [51] generates global instance queries and frame-level box queries, both of which are used to predict dynamic mask head weights to generate mask sequences. Mask2Former [9] (for image-level segmentation) trivially generalizes to videos [8] by attending to spatio-temporal volumes. However, offline methods can only be used on

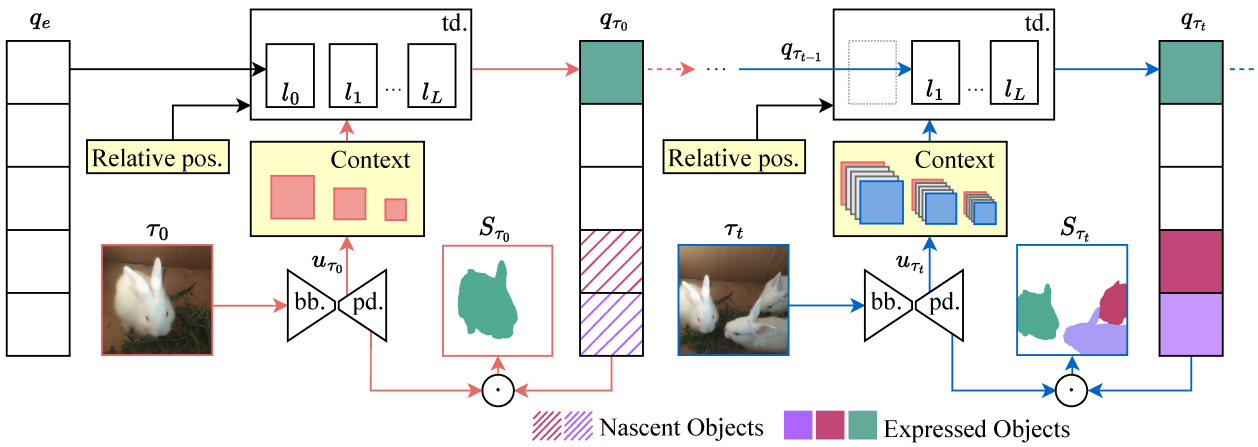

Figure 2. Propagation of context-aware relative object queries across video frames. A single set of query-vectors represents all objects in a video; objects are either expressed or nascent in a given frame. The query vectors q_{τ_0} for the first frame τ_0 are obtained using learnt query embeddings q_e and context-features (pixel decoder features u_{τ_0}) from the first frame. Query vectors q_{τ_0} act as strong object proposals for the next frame and so on. Hence, $q_{\tau_{t-1}}$ can be used to meaningfully generate the query vectors q_{τ_t} with fewer transformer decoder layers (l_0 is skipped). The context-features, obtained from the current and few previous frames, modulate the object queries. S_{τ_0} and S_{τ_t} represent the output segmentation masks in frames τ_0 and τ_t respectively. The abbreviations 'bb.', 'pd.' and 'td.' stand for backbone, pixel decoder and transformer decoder. 'Relative pos.' refers to the learnt relative position encoding in the transformer decoder that better captures an object's position changes across frames. Orange and blue represent input, output, features and queries corresponding to frames τ_0 (orange) and τ_t (blue). Object queries and segmentation masks corresponding to the three rabbits are shown using green, purple and maroon.

short videos. Besides, surprisingly, offline methods don't perform as well as some query-based online frame-by-frame methods [22, 52]. The best performing offline method [18] generates object queries per frame and then combines them temporally for predictions. Results indicate that query vectors don't extend trivially to temporal tasks.

Different from prior work, we propose a transformer-based online method that seamlessly propagates query vectors to subsequent frames for predictions, obviating the need for any post-processing. Context-based frame-processing produces high quality segmentations, and our query propagation seamlessly links predictions across frames. The use of relative positional encodings accurately captures position changes of objects, leading to better association.

2.2. Multi-Object Tracking and Segmentation

The multi-object tracking and segmentation (MOTS) and VIS tasks are identical, but MOTS datasets often consists of fast moving objects. MOTS mainly focuses on 2 categories (cars and pedestrians), whereas VIS includes many more (e.g., 40 categories in the Youtube-VIS dataset).

Several works on MOTS [4, 12, 19, 35, 50] first detect (and segment) objects for individual frames before associating the objects across frames. Others [27, 34, 37, 47, 54, 60] approach detection (and segmentation) and association jointly. Bergmann et al. [2] introduced the idea of tracking by regression, i.e., the bounding box of an object for a current frame acts as a region proposal for the same object in the next frame. In our work, we adopt a similar strategy. However, instead of bounding boxes, object queries in a current frame act as proposals for the object queries in the next frame.

Several transformer-based methods [38, 44, 58] have been proposed recently to approach multi-object tracking. Some operate frame-by-frame [38, 44], others clip-by-clip [58]. These approaches extend DeTR [7] temporally and use 2 types of queries: a) new object queries that are initialized in every frame or clip to generate new objects; b) track queries that are retained to represent old objects. Predictions are obtained via post-processing.

Our work also adopts a query propagation approach like [38, 44, 58]. However, it is simpler and seamless, without any need for heuristics and post-processing. Specifically, unlike prior work, we use a *single* set of query vectors to represent all objects in a given video and refine them again and again for new frames: a) to detect new objects, b) to remove objects that disappeared, and c) to retain existing objects without any bells and whistles.

2.3. Video Panoptic Segmentation

Video panoptic segmentation [28] requires pixel-wise semantic categorization of a video and simultaneous assignment of instance identities to the 'thing' objects, similar to image panoptic segmentation. The 'stuff' category isn't instantiated. VPS-Net [28] and ViP-DeepLab [42] approach this problem using a branched architecture separately for instance and semantic segmentation, before combining the predictions. In contrast, we use query vectors to generically represent objects both from the 'thing' as well as the 'stuff' categories without treating them separately.

2.4. Other Video Segmentation Tasks

Apart from the aforementioned tasks, other video segmentation tasks have been proposed. Video object segmentation (VOS) [53] uses given ground truth segmentations of objects (belonging to any category) in the first frame to track the objects throughout the video [11, 21, 23, 24, 40, 46, 48, 59]. Video semantic segmentation (VSS) [14, 20, 31] requires pixel-wise categorization of a video, an extension of the image segmentation task. In this work, we are interested in objects from known categories (unlike VOS) whose identities need to be associated over time (unlike VSS). Hence, we don't tackle these two tasks.

2.5. General Video Segmentation

Recently there has been progress in unifying video related segmentation tasks *or* tracking tasks. TubeFormer [29] unifies VSS, VPS and VIS by offline processing of entire videos using dual-path transformer blocks with local and global memory. Unicorn [55] unifies many tracking tasks via a separate frame-level and instance level embedding. However, despite these attempts, there remains a gap between tracking *and* video segmentation tasks, and little efforts have been made to unify both video segmentation *and* tracking.

3. Context-Aware Relative Object Query Propagation

3.1. Overview

Given a video, the goal of video instance segmentation is to predict the classes for a set of objects of interest in the video and the corresponding instance-level segmentation masks of the objects at every frame. To achieve this goal, we adopt a meta-architecture similar to recent works [9, 10]. However note, our approach is flexible enough to be used with other query-based meta-architectures like [7]. As illustrated in Fig. 2, similar to prior work [9, 10], our meta-architecture consists of a backbone (bb.) feature extractor for frames, a pixel decoder (pd.) to generate high-resolution image features, and a transformer decoder (td.) to generate meaningful object queries. An object's class is computed every frame from the object query using a linear layer following [9]. The segmentation mask of an object at every frame is computed by passing the object query through linear layers, followed by an inner product between the output of the linear layers and high-resolution image features coming from the pixel decoder.

Using this meta-architecture, every frame of a video can be processed independently. But it is unclear how to extend the architecture to video segmentation tasks, where segmentations need to be linked across time. This is because 1) online frame-by-frame processing has limited temporal context and frame-by-frame or clip-by-clip methods require a post processing association step; 2) offline processing of the entire video using global object queries doesn't accurately encode the position-changes of objects across frames and doesn't scale to long videos.

To address both concerns, as illustrated in Fig. 2, we sequentially process each frame τ of the given video with a transformer-based refinement of context-aware relative object queries across frames. We first discuss the propagation of object queries from one frame to the next (Sec. 3.2). We then describe the use of relative position encodings (Sec. 3.3) for computing the relative object queries. Finally we discuss the processing of a single frame with temporal context (Sec. 3.4). Please see the supplementary material for more training details.

3.2. Propagation of Object Queries

Given the t-th input frame τ_t, we define the query vectors $q_{\tau_t} \in \mathbb{R}^{N \times C}$ to be a collection of N object queries (each of which is C dimensional) which represent a maximum of N objects in the entire video. Importantly, the query vectors q_{τ_t} represents two kinds of objects: (a) *expressed objects* which appear in the current frame τ_t; and (b) *nascent objects* which are absent in τ_t but have appeared previously, or are yet to appear. Query vectors attend to itself via self-attention and to image features via cross-attention, both using a new relative positional encoding, discussed in Sec. 3.3.

We introduce the idea of representing all objects in a video by a single set of query vectors. This allows us to seamlessly use the same query vectors again and again for new frames: (a) to detect new objects, (b) to remove disappearing objects, and (c) to retain existing objects without any bells and whistles. This differs from and is simpler than prior work [38], where new object queries are initialized at every frame, are concatenated with the old object queries, and are passed through attention layers to remove redundancies.

Fig. 2 illustrates the proposed query vector propagation approach. Orange and blue borders and arrows represent input, output, features and queries corresponding to the first frame τ_0 (orange) in a video and the t-th frame τ_t (blue).

For the first input frame τ_0, the query vectors q_{τ_0} are obtained from learnt query embeddings $q_e \in \mathbb{R}^{N \times C}$ and refined via multi-level context-features described in Sec. 3.4. Intuitively, the learnt query embeddings q_e act as abstract proposals for the objects in the frame. Subsequent transformer decoder layers successively refine the initial proposals q_e into the final query vectors q_{τ_0}, which are obtained after q_e is modified by $L+1$ transformer decoder layers $(l_0, \ldots, l_L)$ as shown in Fig. 2.

As the appearance of objects changes gradually across frames, the final query vectors $q_{\tau_{t-1}}$ obtained in frame τ_{t-1} contain valuable information about the next frame τ_t. We leverage the redundancies across frames and only use $q_{\tau_{t-1}}$ as proposals for the current frame τ_t. Importantly, we don't use the abstract query embeddings q_e to generate the query vectors q_{τ_t} for the current frame, as $q_{\tau_{t-1}}$ already contains meaningful information about frame τ_t. Hence, we find

fewer transformer decoder layers, i.e., $l_1, \ldots, l_L$, are enough to modulate the existing objects' appearances, express new objects that were nascent in the previous frame, or suppress old objects that are nascent in the current frame. For this, l_0 is skipped, as shown in Fig. 2.

We now make this process more formal. For frame τ_t, we denote the query vectors modified by the l-th transformer decoder layer as $q^l_{\tau_t}$. We obtain $q^l_{\tau_t}$ by masked cross-attention, self-attention and feed forward operations following [9]. The cross-attention operation is formulated as follows:

$$q^l_{\tau_t} = \mathrm{softmax}(\mathcal{M}^{l-1}_{\tau_t} + \alpha^{\mathrm{rel},l}_{\tau_t})V^l U^l_{\tau_t} + q^{\mathrm{prev}}, \quad (1)$$

where $q^{\mathrm{prev}} = \mathbb{1}_{\{l\neq 1\}}[q^{l-1}_{\tau_t}] + \mathbb{1}_{\{l=1\}}[q_{\tau_{t-1}}]$. Moreover, $\mathcal{M}^{l-1}_{\tau_t} \in \{0, -\infty\}^{N\times H^l W^l T}$ is the attention mask from the previous layer following [9], $\alpha^{\mathrm{rel},l}_{\tau_t} \in \mathbb{R}^{N\times H^l W^l T}$ is the relative attention matrix described in Sec. 3.3, $U^l_\tau \in \mathbb{R}^{H^l W^l T\times C}$ are the context features from the pixel decoder as described in Sec. 3.4 and V^l is a linear transformation for the context features. $H^l W^l T$ represents the flattened context features of height H^l, width W^l and context length T. The final set of query vectors is the output from the last decoder layer, i.e., $q_{\tau_t} = q^L_{\tau_t}$.

3.3. Relative Object Queries

We now discuss the relative attention matrix $\alpha^{\mathrm{rel},l}_{\tau_t}$ employed in Eq. (1) which is used to compute the relative object queries. We observe that relative positional encodings are crucial for accurate results. This differs from the use of absolute encodings in prior work, which often cause query vectors to overly rely on spatial positions in the image space (as seen in Fig. 1).

Transformer-based image detection/segmentation models [7,9] use absolute spatial encodings to represent ordering of tokens while calculating self- or cross-attention. Video segmentation models [8, 49] extend this to use absolute spatio-temporal encodings. However, we observe an absolute spatio-temporal encoding to create an unwanted dependency of the query vectors on the spatial position of objects. To illustrate this, let the cross-attention matrix with absolute encodings for frame τ_t and decoder-layer l be called $\alpha^{\mathrm{abs},l}_{\tau_t}$. Following [8], $\alpha^{\mathrm{abs},l}_{\tau_t}$ can be written as follows:

$$\alpha^{\mathrm{abs},l}_{\tau_t} = Q^l(q^{\mathrm{prev}} + \xi_Q)[K^l(U^l_{\tau_t} + \xi_K)]^\top. \quad (2)$$

Here, K^l and Q^l are linear transformations for context features $U^l_{\tau_t}$ and query vectors q^{prev} respectively; $\xi_Q \in \mathbb{R}^{N\times C}$ and $\xi_K \in \mathbb{R}^{H^l W^l T\times C}$ refer to the absolute learnt position encodings for q^{prev} and absolute sinusoidal position encodings for $U^l_{\tau_t}$ following [9]. Absolute positional encodings cause object queries to depend on the temporary positions of objects in the given frame. This can be seen when Eq. (2) is expanded. The terms $Q^l\xi_Q {U^l_{\tau_t}}^\top {K^l}^\top$ and $Q^l q^{\mathrm{prev}} {\xi_K}^\top {K^l}^\top$ combine positions and content of q^{prev} and $U^l_{\tau_t}$. The term $Q^l\xi_Q {\xi_K}^\top {K^l}^\top$ operates purely on positional encodings. This mix of content and positions of object queries and context-features is not desirable in a temporal setting because it causes object queries to not capture motion properly. In addition, when an object replaces another object spatially, identity switches are common. This observation has been made previously in language modeling tasks [13], but has not been addressed for video segmentation.

To address this concern, we propose to introduce relative positional encodings to compute the self and cross-attention operations in the transformer decoder as shown in Fig. 2. The idea is to encode the relative positional information between q^{prev} and $U^l_{\tau_t}$, instead of injecting the absolute positions, which helps to incorporate temporal information.

In particular, the cross attention matrix $\alpha^{\mathrm{rel},l}_{\tau_t}$ for the τ_t-th frame and level l of the transformer decoder is computed as follows:

$$\alpha^{\mathrm{rel},l}_{\tau_t} = Q^l q^{\mathrm{prev}} {U^l_{\tau_t}}^\top {K^l}^\top + Q^l q^{\mathrm{prev}} {\xi^{\mathrm{rel},l}}^\top {K^l}^\top. \quad (3)$$

Here, $\xi^{\mathrm{rel},l} \in \mathbb{R}^{H^l W^l T\times C}$ refers to the relative positional encodings. Each element of $\xi^{\mathrm{rel},l}$ is a relative distance between two positions. See the supplementary material for more.

3.4. Processing of a Single Frame With Context

To obtain predictions for a single frame τ_t, as shown in Fig. 2, the frame is first passed through the backbone and the pixel decoder and flattened to obtain multi-level image-features $u_{\tau_t} = \{u^l_{\tau_t}\}$, where $u^l_{\tau_t} \in \mathbb{R}^{H^l W^l\times C}$. Here, l denotes the level. We define context-features as a temporary bank of multi-level image features for T consecutive frames ($T-1$ past frames and u_{τ_t}) as shown in Fig. 2. T refers to the context-length. The context-features $U^l_{\tau_t} \in \mathbb{R}^{H^l W^l T\times C}$ for level l can be represented as follows:

$$U^l_{\tau_t} = [u^l_{\tau_{t-T+1}}, \ldots, u^l_{\tau_{t-1}}, u^l_{\tau_t}]. \quad (4)$$

Features from frames older than τ_{t-T+1} are discarded from the bank.

To generate the query vectors q_{τ_t} for the current frame τ_t, the context features are passed to the transformer-decoder in a round robin fashion, where they are used to modulate the object queries (as discussed in Sec. 3.2). Importantly, the object queries attend to the multi-level image features for all the T frames at once for predictions in the current frame τ_t. The use of context better captures spatio-temporal details than using only the current frame features.

The query vectors q_{τ_t} for frame τ_t are used to obtain the predicted class and segmentation masks for all objects in frame τ_t using linear layers (omitted in Fig. 2 for readability). The output from the class head is the matrix $C_{\tau_t} \in [0,1]^{N\times(M+1)}$, which indicates for each of the N object proposals a probability distribution over the M categories (with an additional no-object category) which are

					OVIS					Youtube-VIS 2019					Youtube-VIS 2021				
Method	Bb.	Steps	FPS	Onl.	AP	AP_{50}	AP_{75}	AR_1	AR_{10}	AP	AP_{50}	AP_{75}	AR_1	AR_{10}	AP	AP_{50}	AP_{75}	AR_1	AR_{10}
MaskTrack [56]	R50	D+A	26.1	✓	10.8	25.3	8.5	7.9	14.9	30.3	51.1	32.6	31.0	35.5	28.6	48.9	29.6	26.5	33.8
IFC [25] (T=5)	R50	D+A	46.5	✓	-	-	-	-	-	41.0	62.1	45.4	43.5	52.7	-	-	-	-	-
DeVIS [5] (T=6, S=4)	R50	D+A	18.4	✓	23.7	47.6	20.8	12.0	28.9	44.4	66.7	48.6	42.4	51.6	43.1	66.8	46.6	38.0	50.1
MinVIS [22]	R50	D+A	-	✓	25.0	45.5	24.0	13.9	29.7	**47.4**	69.0	**52.1**	**45.7**	**55.7**	**44.2**	66.0	48.1	**39.2**	**51.7**
IDOL [52]	R50	D+A	30.6	✓	**30.2**	**51.3**	**30.0**	**15.0**	**37.5**	46.4	**70.7**	51.9	44.8	54.9	43.9	**68.0**	**49.6**	38.0	50.9
VisTR [49]	R50	1-step	30.0	×	-	-	-	-	-	36.2	59.8	36.9	37.2	42.4	-	-	-	-	-
PCAN [27]	R50	1-step	15.0	✓	-	-	-	-	-	36.1	54.9	39.4	36.3	41.6	-	-	-	-	-
InsPro [15]	R50	1-step	26.3	✓	-	-	-	-	-	43.2	65.3	48.0	38.8	49.0	37.6	58.7 4	0.9	32.7	41.4
SeqFormer [51]	R50	1-step	12.0	×	-	-	-	-	-	45.1	66.9	50.5	45.6	54.6	40.5	62.4	43.7	36.1	48.1
Mask2Former-VIS [9]	R50	1-step	-	×	-	-	-	-	-	46.4	68.0	50.0	-	-	40.6	60.9	41.8	-	-
VMT [26]	R50	1-step	-	✓	16.9	36.4	13.7	10.4	22.7	47.9	52.0	45.8	-	-	-	-	-	-	-
InstanceFormer [30]	R50	1-step	-	✓	20.0	40.7	18.1	12.0	27.1	45.6	68.6	49.6	42.1	53.5	40.8	62.4	43.7	36.1	48.1
VITA [18]	R50	1-step	-	×	19.6	41.2	17.4	11.7	26.0	**49.8**	**72.6**	**54.5**	**49.4**	**61.0**	**45.7**	**67.4**	**49.5**	**40.9**	**53.6**
Ours	R50	1-step	40.2	✓	**25.8**	**47.9**	**25.4**	**14.2**	**33.9**	46.7	70.4	50.9	45.7	55.9	43.3	64.9	47.1	39.3	52.7
DeVIS [5] (T=6, S=4)	SL	D+A	18.4	✓	35.5	59.3	38.3	16.6	39.8	57.1	80.8	66.3	50.8	61.0	54.4	77.7	59.8	43.8	57.8
MinVIS [22]	SL	D+A	-	✓	39.4	61.5	41.3	**18.1**	43.3	**61.6**	83.3	68.6	**54.8**	**66.6**	55.3	76.6	62.0	**45.9**	**60.8**
IDOL [52]	SL	D+A	17.6	✓	**42.6**	**65.7**	**45.2**	17.9	**49.6**	61.5	**84.2**	**69.3**	53.3	65.6	**56.1**	**80.8**	**63.5**	45.0	60.1
VMT [26]	SL	1-step	8.2	✓	19.8	39.6	17.2	11.2	26.3	59.7	66.7	52.0	-	-	-	-	-	-	-
SeqFormer [51]	SL	1-step	-	×	-	-	-	-	-	59.3	82.1	66.4	51.7	64.4	51.8	74.6	58.2	42.8	58.1
Mask2Former-VIS [9]	SL	1-step	-	×	-	-	-	-	-	60.4	84.4	67.0	-	-	52.6	76.4	57.2	-	-
MS-STS [45]	SL	1-step	-	×	-	-	-	-	-	61.0	85.2	**68.6**	54.7	66.4	-	-	-	-	-
VITA [18]	SL	1-step	-	×	27.7	51.9	24.9	14.9	33.0	**63.0**	**86.9**	67.9	**56.3**	**68.1**	**57.5**	**80.6**	**61.0**	**47.7**	**62.6**
Ours	SL	1-step	20.5	✓	**38.2**	**60.7**	**39.5**	**17.7**	**44.1**	61.4	82.8	**68.6**	55.2	**68.1**	54.5	75.4	60.5	45.5	61.4

Table 1. Different methods on Video Instance Segmentation (OVIS, Youtube-VIS 2019 and Youtube-VIS 2021 validation data) using Resnet-50 (R50) and Swin-L (SL) backbones. We categorize the methods based on backbones: R50 and SL, and based on steps: 1-step or 2-steps (Detection followed by association (D+A)). 'Onl.' indicates whether a method is online or near-online (✓) or offline (×). VITA [18] performs best on the Youtube-VIS data, but is sub-optimal for the more complex OVIS data with longer videos.

of interest in a given dataset. The output of the mask head is multiplied with the high resolution image features of the current frame to get the final segmentation masks of all objects $S_{\tau_t} \in [0,1]^{N \times H \times W}$. Note, H and W represent the height and width of frame τ_t, N is the maximum number of considered objects in a given video.

4. Experiments

We evaluate our proposed approach on three challenging tasks: VIS, VPS and MOTS. We first discuss the datasets and evaluation metrics. In Sec. 4.1 we then present the main results using the following datasets: OVIS, Youtube-VIS, Cityscapes VPS, MOTS 2020, and KITTI-MOTS. Next, in Sec. 4.2, we present ablation studies. We show the importance of our query vector propagation approach as opposed to a [38]-like approach. We then show that relative positional encodings are better at associating objects as opposed to absolute encodings. We also show the importance of having a temporal context length T greater than 1 to incorporate temporally rich information while processing a single frame. Lastly, we show some qualitative results.

Datasets and Evaluation Metrics. We evaluate our approach on the VIS, VPS and MOTS task. For VIS, we use the challenging OVIS [41] dataset. We also evaluate our approach on Youtube-VIS 2019 and 2021 [56] data. We test our approach on the VPS task using the Cityscapes-VPS [28] data and on the MOTS task using the KITTI-MOTS and MOTS 2020 data.

	Cityscapes-VPS				
Method	2-Br.	Dep.	VPQ	VPQ_{th}	VPQ_{st}
ViP-DeepLab [42]	✓	✓	63.1	49.5	73.0
VPS-Net [28]	✓		57.5	44.8	66.7
Ours			**63.0**	**48.0**	**72.8**

Table 2. Results on VPS. Our method, while being general, performs better than VPS-Net [28], which has 2 separate branches (2-Br.) for semantic and instance segmentation. ViP-DeepLab [42], also uses a 2-branch network and also depth for training.

For VIS, we use the standard evaluation metrics of average precision (AP, AP_{50}, AP_{75}) and average recall (AR_1, AR_{10}). For MOTS, we use the sMOTSA (soft MOTS Accuracy) [47], MOTSA (MOTS Accuracy) and MOTSP (MOTS Precision). For ablation studies with KITTI-MOTS, we use the higher order tracking accuracy (HOTA), detection accuracy (DetA) and association accuracy (AssA) [36] because they better capture the detection and tracking aspects of the MOTS task. For VPS, we use video panoptic quality metrics (VPQ, VPQ_{th}, VPQ_{st}) to evaluate the overall performance, performance on the 'thing' category and on the 'stuff' category, as proposed before [28].

4.1. Main Results

We discuss the results of different approaches in this section. We use context-length, $T = 2$ in all experiments

MOTS 2020			
Method	sMOTSA	MOTSA	MOTSP
TrackRCNN [47]	52.7	66.9	80.2
PointTrack [54]	58.1	70.6	-
TrackFormer [38]	58.7	-	-
Ours	**60.2**	**73.2**	**84.3**

Table 3. Results of different methods on the MOTS 2020 dataset.

KITTI-MOTS							
Method	Ext.	sMOTSA		MOTSA		MOTSP	
		car	ped.	car	ped.	car	ped.
MOTSFusion [35]	Dep.	82.8	59.4	90.5	72.6	-	-
PointTrack [54]	Flow.	**85.5**	62.4	**94.9**	77.3	-	-
TrackRCNN [47]		76.2	46.8	87.8	65.1	87.2	75.7
PCAN [27]		-	-	89.6	66.4	88.3	76.1
Ours		84.5	**62.8**	94.0	**77.6**	**92.3**	**84.5**

Table 4. Results of different methods on the KITTI-MOTS dataset.

unless otherwise stated. We use the Resnet-50 [17] (R50) and Swin-L [32] (SL) backbones when comparing our approach to other methods.

Evaluation on VIS Data. Tab. 1 summarizes the results of different methods on the OVIS, Youtube-VIS 2019 and 2021 validation data. We categorize the methods based on backbone and whether a post processing step is required. 'D+A' refers to detection followed by a post processing association step. Our online approach improves upon other 1-step approaches (which are offline) by 5.8% and 10.5% on the OVIS validation data using the Resnet-50 (R50) and Swin-L (SL) backbones. However, we find a reduction in performance for the Youtube-VIS data as compared to VITA [18]. Note, VITA [18] is an offline method that uses independent object queries per frame which are then combined for predictions. Although VITA [18] performs better on the Youtube-VIS data with simpler scenes and shorter videos, it performs worse than the proposed method on the more challenging OVIS data. Further note, the OVIS data contains long video sequences (often with 500 frames). Hence, recent transformer-based offline approaches [8, 18, 51] can't be evaluated on this dataset without predictions on short clips and heuristic merging. Note that our 1-step method underperforms as compared to the 2-step IDOL [52], possibly due to the disentangled detection and association steps in IDOL. However, we highlight our method's generalizability across a wide range of tasks (VIS, VPS, MOTS). IDOL has only been tested on VIS, and the absolute encodings used in IDOL are seen to be sub-optimal for MOTS, as highlighted in Tab. 6.

Evaluation on VPS Data. Tab. 2 shows results of different methods on the VPS task. Our proposed method improves upon VPS-Net [28] and performs similar to ViP-DeepLab [42], which uses a specialized 2-branch architecture particularly developed for panoptic segmentation: the 2 branches perform semantic and instance segmentation separately. ViP-DeepLab [42] also uses additional depth data for training and has an additional branch to estimate monocular depth. The results show that the proposed general and simple approach is effective enough to replace specialized 2-branch architectures for the VPS task.

Method	AP	AP_{50}	AP_{75}	AR_1	AR_{10}
Ours	**25.8**	**47.9**	**25.4**	**14.2**	**33.9**
w. QC	16.9	34.2	15.2	11.9	23.9
w/o. QP	9.8	24.6	5.8	9.4	14.7

Table 5. Importance of our query propagation approach. Abbreviations 'w. QC' and 'w/o. QP' represent with query concatenation and without query vector propagation.

Method	Car			Pedestrian		
	HOTA	DetA	AssA	HOTA	DetA	AssA
Ours	**83.2**	**84.5**	**85.0**	**64.1**	**64.4**	**63.7**
Ours (absolute pos.)	70.4	78.6	62.7	52.0	58.0	46.4

Table 6. Importance of relative positional encodings.

T	Bb.	AP	AP_{50}	AP_{75}	AR_1	AR_{10}	AP_{so}	AP_{mo}	AP_{ho}
1	R50	19.1	36.0	17.8	11.1	27.0	31.2	22.3	5.9
2	R50	**25.8**	47.9	**25.4**	**14.2**	**33.9**	**39.5**	29.1	**9.7**
4	R50	**25.8**	**49.5**	24.4	13.8	33.0	39.1	**29.4**	**9.7**
1	SL	34.0	56.0	34.3	15.9	41.5	47.4	37.9	16.2
2	SL	**38.2**	**60.7**	**39.5**	**17.7**	**44.1**	**57.5**	**43.2**	**18.7**

Table 7. Ablation to show how context length T affects performance. R50 and Swin-L (SL) backbones are used.

Evaluation on MOTS Data. Tab. 3 and Tab. 4 show the results on the MOTS 2020 and KITTI-MOTS validation data. We obtain the best results on the MOTS 2020 data and perform similar to the highly specialized PointTrack [54] that uses optical flow on the KITTI-MOTS data.

4.2. Ablation Studies

We now study the importance of each component in our approach. First we show how our method of query propagation is superior to [38]-like propagation or heuristic association. We then show how relative positional encodings improve results and how temporal context length T affects performance. We use the OVIS dataset for these analyses unless mentioned otherwise. Note, the performance changes are more drastic when using a challenging dataset like OVIS that contains complex scenes and severe occlusions.

Effect of query propagation. The 2nd row of Tab. 5 shows a [38]-like setting for query vector propagation. We call this setting 'w. QC', i.e., with query concatenation. Instead of using our query-vector propagation approach where a single set of object queries is refined repeatedly, we use the learnt query embeddings q_e for each frame (to represent new objects) and concatenate them with the queries from the previous frame (to retain old objects). We pass the concatenated queries through a MLP to retain the original number of N total queries. Note that none of the heuristics from [38] were used in this setting. We observe a performance drop in AP from 25.8 (our approach, row 1 in Tab. 5) to 16.9

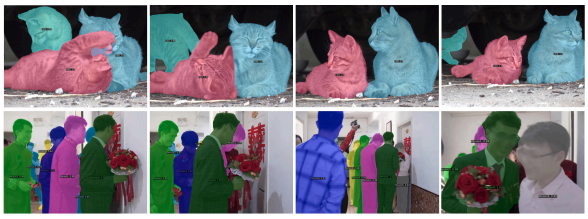

Figure 3. Examples from the OVIS data. Our method can retain identities of objects despite absence (first example), occlusion and viewpoint changes (second example). In the first example, the green cat leaves the scene and reappears 54 frames later. Our approach correctly identifies the cat when it reappears. In the second example (crowded scene), our method correctly retains the identities of all people, despite heavy occlusion and viewpoint changes.

Figure 4. Video panoptic segmentation on Cityscapes-VPS data. The segmentation masks are overlaid on the images.

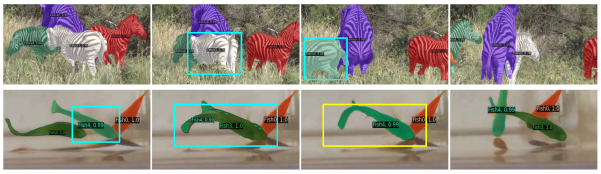

Figure 5. Failure modes of our method. In the first example (top row) the green and white zebras are swapped as highlighted by cyan boxes. In the second example, the light and dark green fish are swapped as shown with cyan boxes (although the identities are successfully recovered later as highlighted with a yellow box).

using this setting. The last row of Tab. 5 shows results when object queries aren't propagated. We call this setting 'w/o. QP': inference is performed frame-by-frame using only q_e as object proposals. The final objects are matched based on mask-overlap using the Hungarian algorithm [39]. We observe a drop in AP to 9.8 with this setting.

Effect of relative positional encodings. We use the KITTI-MOTS dataset for this analysis. The official metrics for KITTI-MOTS, detection accuracy (DetA) and association accuracy (AssA), effectively capture the performance gaps in detection and association, providing better analysis. Notably, objects in this dataset often have fast motion, which better highlights the relevance of relative positional encodings. The performance gap isn't as severe on OVIS data. Tab. 6 shows the performance differences. Using absolute positional encodings (row 2 – 'Ours (absolute pos.)'), DetA drops by 6% for both cars and pedestrians, and AssA drastically decreases by 22% for cars and 17% for pedestrians. The overall HOTA is lower by 13% for cars and 12% for pedestrians. This clearly indicates the benefit of using relative positional encodings (also highlighted in Fig. 1). We show the effect of relative positional encodings on the VIS task in the supplementary material.

Effect of context-length. Tab. 7 shows how the length T of the context affects performance. $T = 1$ defaults to frame-by-frame inference. We clearly observe that frame-by-frame inference is sub-optimal as compared to larger temporal context. As expected, temporal context is important to generate object queries. However, the performance improvement is marginal with $T > 2$. This suggests that the object queries already retain long-term temporal information and additional context-features are no longer significant.

4.3. Qualitative Results

Fig. 3 shows 2 examples from the OVIS dataset. They represent 4 non-consecutive frames from 2 videos. Our approach retains identities of objects despite long-term absence and heavy occlusion. In the first example, the green cat leaves the scene and reappears after multiple frames (last frame). Our approach correctly identifies the cat when it reappears. The second example shows a crowded scene, where our method correctly retains the identities of all persons, despite heavy occlusions and viewpoint changes. Fig. 4 shows images overlaid with segmentation masks generated by the proposed method for VPS (Cityscapes-VPS dataset). We highlight the compelling quality of the generated masks.

Failure cases. Fig. 5 shows two cases, where our method exhibits identity switches. In the first case (top row) the green and white zebras are swapped as shown with cyan boxes. In the second case, the light and dark green fish are swapped as shown with cyan boxes (although the identities are successfully recovered later, as shown with a yellow box). These cases show that results are promising, but context-aware relative object queries need to be improved further.

5. Conclusion

In this work, we introduce context-aware relative object queries for online video instance and panoptic segmentation. The object queries are continuously refined every frame by a transformer decoder and propagated across video frames to seamlessly predict segmentations. We reach or surpass the current state-of-the-art on three challenging video-segmentation tasks. We demonstrate in ablation studies that each of our developed components contributes to the success.

Acknowledgements: This work is supported in party by Agriculture and Food Research Initiative (AFRI) grant no. 2020-67021-32799/project accession no.1024178 from the USDA National Institute of Food and Agriculture: NSF/USDA National AI Institute: AIFARMS. We also thank the Illinois Center for Digital Agriculture for seed funding for this project. Work is also supported in part by NSF under Grants 2008387, 2045586, 2106825, MRI 1725729.

References

[1] 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. 1, 2

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

[3] Gedas Bertasius and Lorenzo Torresani. Classifying, segmenting, and tracking object instances in video with mask propagation. In *ICCV*, 2020. 1, 2

[4] Guillem Braso and Laura Leal-Taixe. Learning a neural solver for multiple object tracking. In *CVPR*, June 2020. 1, 3

[5] Adrià Caelles, Tim Meinhardt, Guillem Brasó, and Laura Leal-Taixé. Devis: Making deformable transformers work for video instance segmentation. *arXiv preprint arXiv:2207.11103*, 2022. 2, 6

[6] 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. 1, 2

[7] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 1, 3, 4, 5, 11, 14

[8] Bowen Cheng, Anwesa Choudhuri, Ishan Misra, Alexander Kirillov, Rohit Girdhar, and Alexander G Schwing. Mask2former for video instance segmentation. *arXiv preprint arXiv:2112.10764*, 2021. 1, 2, 5, 7, 11, 12, 14

[9] Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *CVPR*, 2022. 1, 2, 4, 5, 6, 12, 13, 14

[10] Bowen Cheng, Alexander G. Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. In *NeurIPS*, 2021. 1, 4

[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. 4

[12] Anwesa Choudhuri, Girish Chowdhary, and Alexander G. Schwing. Assignment-space-based multi-object tracking and segmentation. In *ICCV*, 2021. 1, 3

[13] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019. 2, 5, 11

[14] Mohsen Fayyaz, Mohammad Hajizadeh Saffar, Mohammad Sabokrou, Mahmood Fathy, Reinhard Klette, and Fay Huang. Stfcn: spatio-temporal fcn for semantic video segmentation. *arXiv preprint arXiv:1608.05971*, 2016. 4

[15] Fei He, Haoyang Zhang, Naiyu Gao, Jian Jia, Yanhu Shan, Xin Zhao, and Kaiqi Huang. Inspro: Propagating instance query and proposal for online video instance segmentation. *arXiv preprint arXiv:2301.01882*, 2023. 2, 6

[16] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 1, 2

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 7

[18] Miran Heo, Sukjun Hwang, Seoung Wug Oh, Joon-Young Lee, and Seon Joo Kim. Vita: Video instance segmentation via object token association. *arXiv preprint arXiv:2206.04403*, 2022. 1, 3, 6, 7

[19] Andrea Hornakova, Roberto Henschel, Bodo Rosenhahn, and Paul Swoboda. Lifted disjoint paths with application in multiple object tracking. In *ICML*, 2020. 1, 3

[20] Ping Hu, Fabian Caba, Oliver Wang, Zhe Lin, Stan Sclaroff, and Federico Perazzi. Temporally distributed networks for fast video semantic segmentation. In *CVPR*, 2020. 4

[21] Yuan-Ting Hu, Jia-Bin Huang, and Alexander G Schwing. Videomatch: Matching based video object segmentation. In *ECCV*, 2018. 4

[22] De-An Huang, Zhiding Yu, and Anima Anandkumar. Minvis: A minimal video instance segmentation framework without video-based training. In *NeurIPS*, 2022. 1, 2, 3, 6, 14

[23] Xuhua Huang, Jiarui Xu, Yu-Wing Tai, and Chi-Keung Tang. Fast video object segmentation with temporal aggregation network and dynamic template matching. In *CVPR*, 2020. 4

[24] Xuhua Huang, Jiarui Xu, Yu-Wing Tai, and Chi-Keung Tang. Fast video object segmentation with temporal aggregation network and dynamic template matching. In *CVPR*, 2020. 4

[25] Sukjun Hwang, Miran Heo, Seoung Wug Oh, and Seon Joo Kim. Video instance segmentation using inter-frame communication transformers. *NeuRIPS*, 2021. 1, 2, 6

[26] Lei Ke, Henghui Ding, Martin Danelljan, Yu-Wing Tai, Chi-Keung Tang, and Fisher Yu. Video mask transfiner for high-quality video instance segmentation. In *ECCV*, 2022. 6

[27] 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. *NeurIPS*, 2022. 3, 6, 7

[28] Dahun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Video panoptic segmentation. In *CVPR*, 2020. 3, 6, 7

[29] Dahun Kim, Jun Xie, Huiyu Wang, Siyuan Qiao, Qihang Yu, Hong-Seok Kim, Hartwig Adam, In So Kweon, and Liang-Chieh Chen. Tubeformer-deeplab: Video mask transformer. In *ICCV*, 2022. 4

[30] Rajat Koner, Tanveer Hannan, Suprosanna Shit, Sahand Sharifzadeh, Matthias Schubert, Thomas Seidl, and Volker Tresp. Instanceformer: An online video instance segmentation framework. *arXiv preprint arXiv:2208.10547*, 2022. 6

[31] Yule Li, Jianping Shi, and Dahua Lin. Low-latency video semantic segmentation. In *CVPR*, 2018. 4

[32] 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

[33] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 14

[34] Zhichao Lu, Vivek Rathod, Ronny Votel, and Jonathan Huang. Retinatrack: Online single stage joint detection and tracking. In *CVPR*, 2020. 3

[35] Jonathon Luiten, Tobias Fischer, and Bastian Leibe. Track to reconstruct and reconstruct to track. *RAL*, 2020. 1, 3, 7

[36] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixe, and Bastian Leibe. Hota: A higher order metric for evaluating multi-object tracking. *IJCV*, 2020. 6

[37] Wenjie Luo, Bin Yang, and Raquel Urtasun. Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In *CVPR*, 2018. 3

[38] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. Trackformer: Multi-object tracking with transformers. In *CVPR*, 2022. 1, 3, 4, 6, 7

[39] James Munkres. Algorithms for the assignment and transportation problems. *J-SIAM*, 1957. 8, 11, 12

[40] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *ICCV*, 2019. 4

[41] 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. *arXiv preprint arXiv:2102.01558*, 2021. 1, 2, 6

[42] 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. 3, 6, 7

[43] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 1

[44] Peize Sun, Jinkun Cao, Yi Jiang, Rufeng Zhang, Enze Xie, Zehuan Yuan, Changhu Wang, and Ping Luo. Transtrack: Multiple object tracking with transformer. *arXiv preprint arXiv:2012.15460*, 2020. 1, 3

[45] Omkar Thawakar, Sanath Narayan, Jiale Cao, Hisham Cholakkal, Rao Muhammad Anwer, Muhammad Haris Khan, Salman Khan, Michael Felsberg, and Fahad Shahbaz Khan. Video instance segmentation via multi-scale spatio-temporal split attention transformer. In *ECCV*, 2022. 6

[46] 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. 4

[47] Paul Voigtlaender, Michael Krause, Aljoša Ošep, Jonathon Luiten, Berin Balachandar Gnana Sekar, Andreas Geiger, and Bastian Leibe. MOTS: Multi-object tracking and segmentation. In *CVPR*, 2019. 3, 6, 7

[48] Qiang Wang, Li Zhang, Luca Bertinetto, Weiming Hu, and Philip HS Torr. Fast online object tracking and segmentation: A unifying approach. In *CVPR*, 2019. 4

[49] 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. 1, 2, 5, 6

[50] Zhongdao Wang, Liang Zheng, Yixuan Liu, Yali Li, and Shengjin Wang. Towards real-time multi-object tracking. In *ECCV*, 2020. 1, 3

[51] Junfeng Wu, Yi Jiang, Wenqing Zhang, Xiang Bai, and Song Bai. Seqformer: a frustratingly simple model for video instance segmentation. *arXiv preprint arXiv:2112.08275*, 2021. 1, 2, 6, 7, 14

[52] 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. 1, 2, 3, 6, 7, 14

[53] Ning Xu, Linjie Yang, Yuchen Fan, Jianchao Yang, Dingcheng Yue, Yuchen Liang, Brian Price, Scott Cohen, and Thomas Huang. Youtube-vos: Sequence-to-sequence video object segmentation. In *ECCV*, 2018. 4

[54] Zhenbo Xu, Wei Zhang, Xiao Tan, Wei Yang, Huan Huang, Shilei Wen, Errui Ding, and Liusheng Huang. Segment as points for efficient online multi-object tracking and segmentation. In *ECCV*, 2020. 3, 7

[55] Bin Yan, Yi Jiang, Peize Sun, Dong Wang, Zehuan Yuan, Ping Luo, and Huchuan Lu. Towards grand unification of object tracking. In *ECCV*, 2022. 4

[56] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *ICCV*, 2019. 1, 2, 6

[57] Shusheng Yang, Yuxin Fang, Xinggang Wang, Yu Li, Chen Fang, Ying Shan, Bin Feng, and Wenyu Liu. Crossover learning for fast online video instance segmentation. In *ICCV*, 2021. 1, 2

[58] Fangao Zeng, Bin Dong, Tiancai Wang, Xiangyu Zhang, and Yichen Wei. Motr: End-to-end multiple-object tracking with transformer. *arXiv preprint arXiv:2105.03247*, 2021. 1, 3

[59] Lu Zhang, Zhe Lin, Jianming Zhang, Huchuan Lu, and You He. Fast video object segmentation via dynamic targeting network. In *ICCV*, 2019. 4

[60] Yifu Zhang, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. Fairmot: On the fairness of detection and re-identification in multiple object tracking. *IJCV*, 2021. 3