DeVIS: Making Deformable Transformers Work for Video Instance Segmentation

Adrià Caelles[1], Tim Meinhardt[2], Guillem Brasó[2], and Laura Leal-Taixé[2]

[1] Technical University of Catalonia
adria.caelles@estudiantat.upc.edu
[2] Technical University of Munich
{tim.meinhardt,guillem.braso,leal.taixe}@tum.de

Abstract. Video Instance Segmentation (VIS) jointly tackles multi-object detection, tracking, and segmentation in video sequences. In the past, VIS methods mirrored the fragmentation of these subtasks in their architectural design, hence missing out on a joint solution. Transformers recently allowed to cast the entire VIS task as a single set-prediction problem. Nevertheless, the quadratic complexity of existing Transformer-based methods requires long training times, high memory requirements, and processing of low-single-scale feature maps. Deformable attention provides a more efficient alternative but its application to the temporal domain or the segmentation task have not yet been explored.
In this work, we present Deformable VIS (DeVIS), a VIS method which capitalizes on the efficiency and performance of deformable Transformers. To reason about all VIS subtasks jointly over multiple frames, we present temporal multi-scale deformable attention with instance-aware object queries. We further introduce a new image and video instance mask head with multi-scale features, and perform near-online video processing with multi-cue clip tracking. DeVIS reduces memory as well as training time requirements, and achieves state-of-the-art results on the YouTube-VIS 2021, as well as the challenging OVIS dataset.
Code is available at https://github.com/acaelles97/DeVIS.

Keywords: video instance segmentation, deformable transformers

1 Introduction

Video Instance Segmentation (VIS) simultaneously aims to detect, segment, and track multiple classes and object instances in a given video sequence. Thereby, VIS provide rich information for scene understanding in applications such as autonomous driving, robotics, and augmented reality.

Early methods [24,4,25] took inspiration from the multi-object tracking (MOT) field by applying tracking-by-detection methods to VIS, e.g., [24] extends Mask R-CNN [9] with an additional tracking head. The frame-by-frame mask prediction and track association allow for real-time processing but fail to capitalize on temporal consistencies in video data. Hence, more recent VIS methods [2,3,14,13,17] moved towards an offline or near-online processing of clips by treating instance segmentations as 3D spatio-temporal volumes.

The recent success of Transformers [20] in object recognition inspired a new generation of VIS approaches. Both VisTR [21] and IFC [11] deploy an encoder-decoder Transformer architecture, and closely mirror DETR's [5] approach to object detection by formulating the VIS task as a set-prediction problem. In this paradigm, instance masks are obtained in a single end-to-end trainable forward pass for all frames in a clip. While their formulation is simple and appealing, both methods are limited by the quadratic complexity of full attention. VisTR achieves communication between frames by concatenating and encoding the pixels of all frames jointly. Such a multi-frame processing only amplifies the expensive attention computation and makes [21] suffer from long training times and high memory requirements. IFC [11] tries to mitigate these issues by introducing inter-frame memory tokens to encode each frame individually. However, [11] still relies on full attention for single frames and hence inherits the limitations of [5,21] to only process low- and single-scale feature maps.

Deformable attention [27] resolves many of DETR's efficiency and performance issues, and circumvents the quadratic computational complexity for its inputs by subsampling attention keys around a spatial reference point assigned to each query. As shown in our experiments, a naive *deformabilization* of VisTR, i.e., replacing full attention with deformable attention in the encoder-decoder of Figure 1, does not achieve satisfactory train time nor segmentation performance. This is largely due to two problems: (i) learnable reference point offsets and attention weights introduce an unfeasible amount of new parameters for long clip sizes, and (ii) attention with spatially local reference points is not well-suited for the detection and tracking of objects moving through a sequence.

To make deformable attention work for VIS, we present *Deformable VIS* (DeVIS), a Transformer encoder-decoder which applies temporal deformable attention over multiple frames. The reduction in computational complexity reduces training time and makes the processing of high-res feature maps at multiple scales feasible. Furthermore, we motivate the alignment of reference points for decoder object queries individually for each object instance. Our newly proposed image and video instance segmentation head takes full advantage of multi-scale features and improves mask quality significantly. To run sequences with arbitrary lengths, we also introduce an improved multi-cue clip tracking. The presented DeVIS method achieves state-of-the-art results on the challenging YouTube-VIS [24] 2021 and OVIS [19] datasets and substantially reduces training time with respect to VisTR.

In summary, our key **contributions** are:

- We present Deformable VIS (DeVIS), a VIS method which introduces temporal multi-scale deformable attention and instance-aware object queries.
- We present a new instance mask prediction head which takes full advantage of deformable attention and encoded multi-scale feature maps.
- Our improved multi-cue clip tracking incorporates mask and class information to connect overlapping clips to sequences of arbitrary length.
- Our method provides efficient training and achieves state-of-the-art performance on YouTube-VIS 2021 and the challenging OVIS dataset.

2 Related work

We discuss VIS methods following the progression from tracking-by-detection to clip-level approaches culminating in modern Transformer-based architectures.

Tracking-by-detection. The inception of the VIS task with the YouTube-VIS 2019 [24] dataset also created Mask-Track R-CNN [24]. As it is common in the multi-object-tracking community, Mask-Track R-CNN processes sequences frame-by-frame in an online tracking-by-detection manner. To this end, [24] extends Mask R-CNN [9] with a tracking branch which allows it to not only detect and segment objects, but also assign instance identities via similarity matching of instance embeddings in the current frame and a memory queue. Sip-Mask [4] applies the same tracking head but with a single-stage detector and light-weight spatial mask preservation module. The crossover learning scheme of CrossVIS [25] allows for a localization of pixel instance features in other frames.

As a clip-level method, DeVIS processes sequences in clips, which greatly improves quality and robustness of instance mask as well as identity predictions.

Clip-level. Clip-level processing allows for offline or near-online VIS methods. The latter clip a sequence into multiple parts and hence rely on an additional clip tracking step. The STEm-Seg [2] method took inspiration from offline trackers and is the first to model object instances as 3D spatio-temporal volumes by predicting pixel embeddings with Gaussian variances. For a hybrid tracking-by-detection and clip-level method, the authors of MaskProp [3] extend Mask R-CNN with a mask propagation branch that operates between all frames in a video clip. The *Produce-Reduce* heuristic applied in SeqMask-RCNN [14] generates object instance proposals and reduces redundant identities based on multiple key frames. STMask [13] and SG-Net [17], on the other hand, move beyond Mask R-CNN by applying improved one-stage detection methods.

Albeit their early success, these clip-level methods usually tackle the detect, segment, and track subtasks with separately trainable multi-stage pipelines. Our DeVIS approach enjoys the advantages of clip-level methods but in a unified and end-to-end trainable manner through the application of Transformers.

Clip-level with Transformers. The VisTR [21] method introduced Transformers [20] to VIS by extending the DETR [5] object detector to the temporal domain. Its unified Transformer encoder-decoder architecture concatenates and encodes all frames in a clip by computing attention between all pixels. The decoder reasons about detection and tracking via multi-frame cross-attention between object queries and the encoded pixels, and produces instance masks with a subsequent segmentation head. To avoid the expensive computation multi-frame pixel attention, the authors of [11] encode each frame separately and introduce memory tokens for a high-level inter-frame communication.

Our proposed DeVIS method mitigates the aforementioned efficiency issues while still benefiting from the simultaneous encoding of multiple frames at once through the application of temporal multi-scale deformable attention with instance-aware reference point sampling. We further propose a new segmentation head which takes full advantage of the Transformer encoded multi-scale feature maps, and an improved multi-cue clip-tracking.
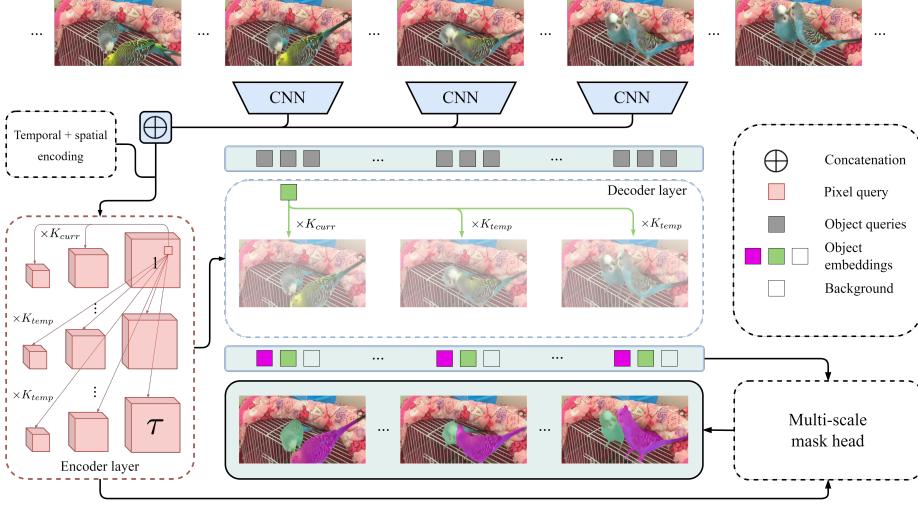
Fig. 1. An overview of our **DeVIS method** which applies temporal multi-scale deformable attention in a Transformer encoder-decoder architecture. The encoder computes deformable attention between pixels across scales and frames in a given clip without suffering from quadratic complexity of full attention. In the decoder, object queries attend to multiple frames, thereby providing consistent identity predictions.

3   DeVIS

In this section, we present *Deformable VIS* (DeVIS), a near-online end-to-end trainable Transformer encoder-decoder architecture for VIS. We details its key components: (i) Temporal multi-scale deformable attention on a (ii) Transformer architecture with instance-aware object queries, (iii) a new multi-scale deformable mask head, and (iv) multi-cue clip tracking.

3.1   Clip-level VIS with Transformers

In this section, we present an overview of our DeVIS method, shown in Figure 1, which follows the general Transformer-based clip processing pipeline of [21].

Clip-level VIS. Given a video with T frames, the goal of VIS is to detect, track, and segment all K objects in a sequence. This is achieved by providing a set of instance predictions $\Upsilon = \{y_{i,t}\}$ with $y_{i,t} = \{c_{i,t}, b_{i,t}, m_{i,t}\}$. A single prediction consists of the class c, bounding box b, and mask m for instance identity i at frame t. The VIS task expects a constant $c_{i,t}$ for all t. If an object instance with identity j is not present for the entire sequence, the final set Υ can include less than T predictions with $y_{j,t}$. A clip-level VIS method processes clips with τ frames and first provides subsets of instance predictions Υ_k with distinct sets of identities. Usually, clips include overlapping frames to perform a final clip tracking/stitching step, which is responsible for merging identities of overlapping clip instance predictions.

VIS with Transformers. To generate a set of clip predictions Υ_k, Transformer encoder-decoder methods first extract feature maps for each frame independently with a convolutional neural network (CNN). Feature maps are then concatenated to form a clip and temporal-positional encoding is added. Treating each pixel as an input query, the subsequent Transformer encoder shares information spatially and temporally between frames via self-attention [20]. A set of learned object queries [5] computes self-attention and cross-attention with all encoded pixels in the Transformer decoder. The total amount of object queries is equally distributed over the frames, hence, a query attends to all pixels in the clip but is responsible for the predictions on a fixed frame. The decoder outputs a set of object query embeddings which are passed through separate multi-layer perceptrons to predict $c_{i,t}$ and $b_{i,t}$ for each frame in the clip. The instance mask predictions $m_{i,t}$ are obtained by computing attention maps for each object embedding, and feeding these together with the backbone feature maps into an additional instance mask head. The entire model is trained end-to-end by matching the predicted outputs $y_{i,t}$ via a Hungarian cost matrix to the ground truth. We refer to [5] and [21] for more details on the matching and loss computation. Instance identities i are predicted by matching a fixed set of queries each from a different frame to the same identity during training. At inference, all objects detected and segmented by one of these query sets are then assumed to belong to the same identity. Intuitively, object queries in DETR learn to detect objects in certain regions of the image. For VIS, each set of queries is responsible for certain types of spatio-temporal object trajectories through the clip.

3.2   Roadmap to temporal deformable attention

The authors of [27] introduced deformable attention to DETR, thereby reducing the computational footprint and training time substantially. This allowed [27] to improve single-image object detection performance by running the Transformer encoder-decoder on multiple feature scales. While they only operated on single images, we present deformable attention for the *temporal domain* which simultaneously captures spatial and temporal dependencies across multiple scales.

Multi-Head Attention. The original Transformer [20] applies full attention between two sets of input queries Ω_q and keys Ω_k. An element of each of these sets is denoted by k and q with feature representations $\boldsymbol{z}_q \in \mathbb{R}^C$ and $\boldsymbol{x}_k \in \mathbb{R}^C$ of hidden size C, respectively. We denote the set of all $\boldsymbol{x}_k$ as $\mathbf{X}$. The Multi-Head Attention (MHA) for query q and M attention heads is then computed via:

$$\text{MHA}(\boldsymbol{z}_q, \mathbf{X}) = \sum_{m=1}^{M} \boldsymbol{W}_m \Big[\sum_{k \in \Omega_k} A_{mqk} \cdot \boldsymbol{W}'_m \boldsymbol{x}_k \Big], \qquad (1)$$

with learnable weight matrices $\boldsymbol{W}'_m \in \mathbb{R}^{C_v \times C}$ and $\boldsymbol{W}_m \in \mathbb{R}^{C \times C_v}$ where $C_v = C/M$. The attention weights A_{mqk} are computed via dot product between $\boldsymbol{z}_q$ and $\boldsymbol{x}_k$ and are normalized over all keys $\sum_{k \in \Omega_k} A_{mqk} = 1$. The case where $\Omega_q = \Omega_k$

is usually referred to as self-attention. The Transformer encoder in DETR [5] applies self-attention, where every entry in the feature map corresponds to a query. Due to the computation of A_{mqk}, which scales quadratically with the number of queries/keys, it is only feasible to run [5] on a single feature scale.

Deformable attention. To mitigate DETR's computational issues around attention, the authors of [27] suggest deformable attention which works on subsets of queries and weight computation of linear complexity. To this end, each query q is assigned a reference point $\boldsymbol{p}_q \in \mathbb{R}^2$ in the feature map domain. A subset of K queries is sampled around the reference point based on sample offsets $\Delta\boldsymbol{p}_{mqk}$ which are learnable via linear projection over the query feature $\boldsymbol{z}_q$. For simplicity we omit the summation over multiple attention heads and denote the resulting *Deformable Attention* (DA) for a single attention head m as:

$$\text{DA}(\boldsymbol{z}_q, \boldsymbol{p}_q, \mathbf{X}) = \boldsymbol{W}_m \Big[\sum_{k=1}^{K} A_{mqk} \cdot \boldsymbol{W}'_m \boldsymbol{x}(\boldsymbol{p}_q + \Delta\boldsymbol{p}_{mqk}) \Big]. \tag{2}$$

The attention weights are normalized over the sample points $\sum_{k\in\Omega_k} A_{mqk} = 1$ and also obtained via linear projection which avoids the expensive computation of dot product between queries. Furthermore, [27] present a multi-scale version of Equation 2 which computes deformable attention across feature maps.

Temporal multi-scale deformable attention. To encode spatio-temporal dependencies, which are crucial for for VIS, we present multi-scale deformable attention for the temporal domain. That is, the sets of queries and keys include pixels from multiple scales and all frames in a clip. Hence, we re-define $\mathbf{X} = \{\mathbf{X}^l\}_{l=1}^{L}$ to be the stack of L multi-scale backbone features with τ frames where $\mathbf{X}^l \in \mathbb{R}^{C\times\tau\times H_l\times W_l}$. Intuitively, a query q from frame t has the ability to compute attention with sampled keys from all frames and feature levels in a clip. In comparison to full attention between all pixels, the sampling with offsets around a query's reference point $\hat{\boldsymbol{p}}_q$ reduces the computational effort substantially. The number of keys is independent of the input resolution and only scales linearly with the number of feature scales and frames. We define Temporal Multi-Scale Deformable Attention (TMSDA) module for a single m over a clip as:

$$\text{TMSDA}(\boldsymbol{z}_q, \hat{\boldsymbol{p}}_q, \mathbf{X}) = \boldsymbol{W}_m \Big[\sum_{t=1}^{\tau} \sum_{l=1}^{L} \sum_{k=1}^{K(t)} A_{mtlqk} \cdot \boldsymbol{W}'_m \boldsymbol{x}^{lt}(\phi_l(\hat{\boldsymbol{p}}_q) + \Delta\boldsymbol{p}_{mtlqk}) \Big]. \tag{3}$$

As in [27], each reference point is represented with normalized coordinates $\hat{\boldsymbol{p}}_q \in [0,1]^2$ and re-scaled by ϕ_l to allow for a sampling across feature maps l with different resolutions. The scalar attention weight A_{mtlqk} is normalized by $\sum_{t=1}^{\tau} \sum_{l=1}^{L} \sum_{k=1}^{K(t)} A_{mtlqk} = 1$. We introduce $K(t)$ which adapts the number of keys sampled from a given frame and present two scenarios depending on whether
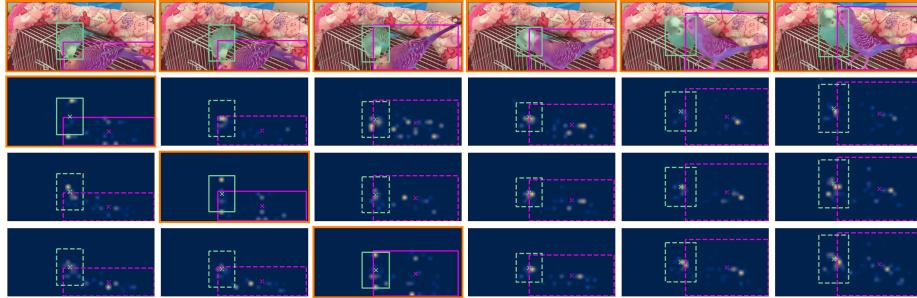
Fig. 2. Attention map visualization of all frames in a clip for two object queries assigned to detect and segment object on the first, second and third frame. The temporal attention computed on other frames successfully follows each object and hence provides consistent identity predictions. Furthermore, we visualize the instance-aware reference point alignment (red dot) which adjusts the reference point of each first-frame query to the position of the respective object in the other frames.

a query z_q samples keys from its corresponding or other temporal frames:

$$K(t) = \begin{cases} K_{curr} & \text{if } z_q \in \mathbf{Z}^{lt} \\ K_{temp} & \text{else.} \end{cases} \tag{4}$$

Adding the temporal dimension allows each query to simultaneously sample keys from all feature scales as well as its current, and temporal frames. Such a design is particularly beneficial for a consistent detection and identity prediction of objects moving and changing size over the sequence. $K_{temp} = 0$ removes all temporal connections and reverts back to the original deformable attention from [27].

In the following paragraphs, we give further details on how our temporal deformable attention is applied in our Transformer encoder-decoder architecture.

3.3 TMSDA for VIS Transformers

Following the formal introduction of Temporal Multi-Scale Deformable Attention (TMSDA), we detail its integration into our Transformer encoder-decoder and present a novel design of instance-aware reference points for object queries.

Transformer encoder. We replace the common full self-attention between all pixels in the encoder with TMSDA as in Equation 3. The sampling design of deformable attention allows each pixel to only connect to a subset of other pixels close to spatial location of the reference points p_q for the current and other frames. The amount of temporal information being considered is controlled by the clip size τ and the number of temporal sampling points K_{temp}. Computing deformable attention allows us to encode all $L = 4$ ResNet [10] backbone feature scales, thereby replacing the role of a FPN [15]. We apply an additive encoding

to each pixel which indicates it spatial, temporal, and feature scale position. The spatial locality of sampled keys limits the available temporal information potentially required for long clips with large object motions. We believe our deformable approach is superior to [21,11] as it allows for a more fine-grained inter-frame communication on multiple scales in a unified formulation.

Transformer decoder. Our Transformer decoder computes two attention steps for its object queries: self-attention and cross-attention between queries and the encoded frame features. For single-image detection [5], the decoder self-attention helps to avoid duplicate object detections. But for VIS Transformers, object queries also need to communicate about instance identities. As explained in Section 3.1, a subset of object queries are assigned to each frame of the clip. To extract object information with an object query from its corresponding and future as well as past frames, we compute temporal deformable multi-scale cross-attention. In contrast to the Transformer encoder, the reference points of object queries required for the cross-attention are learnable. Each object query can leverage not only meaningful, local information from its particular object on its assigned frame, but from each of the other frames of the input clip. This helps improve consistent mask and identity predictions over the sequence. Furthermore, we apply the same bounding box refinement as [27] at each layer of the decoder. This allows the initial reference point, i.e., sampling area, of a query to be adapted to the currently assumed coordinates of the object bounding box.

Instance-aware object queries. Each object query of the decoder is by design able to learn different sampling reference points for different frames in the clip. This not only allows to distinguish between a query's assigned frame and other frames, but to adjust the reference points on other frames according to its instance identity. Hence, we introduce *Instance-aware object queries* which exploit the identity consistency across queries to adapt their reference points on other frames to the predicted bounding boxes belonging to their respective object identity. This instance-aware reference point sampling is applied before each Transformer decoder layer, see Figure 1, and works in conjunction with the bounding box refinement of the reference point on the query frame. If an object query successfully predicts the object bounding box on its own frame t, other object queries belonging to the same instance but from other frames will benefit from an improved sampling on frame t. Instance-aware object queries provide an additional communication between object queries of different frames and improve track consistency.

3.4   Multi-scale deformable mask head

Processing multi-scale features currently only benefits the detection and tracking performance of the Transformer encoder-decoder. We further explore the potential of Transformer encoded feature maps for mask prediction, and present a
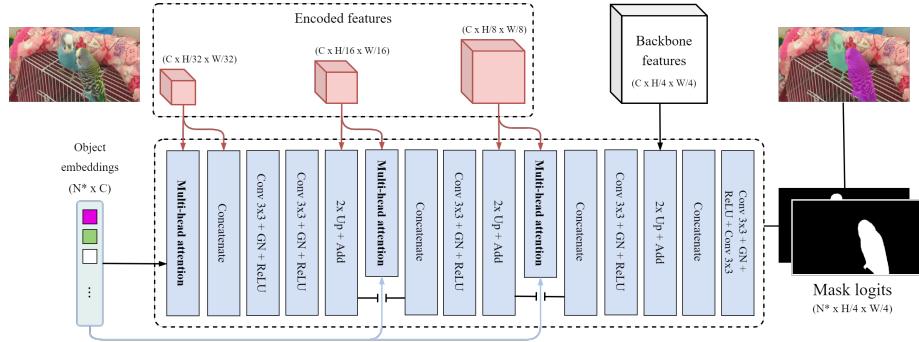
Fig. 3. Overview of our new **multi-scale mask head** for video and image instance segmentation. The upsampling of Transformer-encoded feature maps and multi-scale attention maps boosts performance significantly. Attention maps are generated by computing multi-head attention between feature maps and object queries. We indicate the hidden size and reduced set of object queries with N* and C, respectively. New connections to the decoder (blue) and encoder (red) are shown colored.

new multi-scale deformable instance mask segmentation head applicable to both image and video segmentation.

In Figure 3, we present an overview of its architecture. Our encoder-decoder architecture outputs encoded backbone feature maps and object embeddings per clip. As shown in Figure 1, the mask head predicts binary instance masks by generating attention maps for each of the embeddings and applying a series of convolutions and upsampling operations. Attention maps are generated by computing Multi-Head Attention (MHA), as in Equation 1, between an embedding an the encoded feature maps of its corresponding frame in the clip. At different scales of the upsampling process, corresponding backbone feature maps are added to improve segmentation performance. The mask head predicts all instance masks in a clip at once and does not consider any explicit temporal information. In particular, each query only computes attention maps for its own frame as opposed to the deformable cross-attention computation in the decoder.

To take full advantage of the encoded multi-scale features, we propose the following design changes in order to improve training time and segmentation performance for VIS and image instance segmentation:

Training only positive matches. During the training of DETR, the detection loss is computed via Hungarian matching between the ground truth and the object embeddings produced by the decoder. We reduce the mask head training time by only computing masks and their corresponding losses embeddings which receive a positive matching with a ground truth object. Since object queries are designed to exceed the number of objects per frame by a large margin, this results in a substantial reduction of training time.

Encoded feature maps. The deformable attention allows us to encode all but the largest (H/4 x W/4) backbone feature maps. While [5] only encodes the

lowest (H/32 x W/32) feature map and hence must use raw backbone features in its mask head, we are able to add and upsample multiple Transformer-encoded feature maps as visualized in Figure 3. The application of encoded feature maps allows for a direct connection between the mask head and Transformer encoder.

Multi-scale attention maps. To obtain informative attention maps, the object embeddings must compute MHA with the feature maps in the preceding Transformer encoder-decoder. Therefore, we are able to generate attention maps not only for a single but multiple scales and concatenate these at the corresponding stages of the upsampling process. The original mask head only generates attention maps for the smallest scale (H/32 x W/32) and is not able to benefit from the additional connections to the object embeddings, i.e., the Transformer decoder, during the upsampling process. It should be noted, that although our Transformer encoder-decoder computes deformable attention as in Equation 3, the attention maps are generated via regular attention as in Equation 1.

MDC. Furthermore, we replace the convolutions in the mask head with Modulated Deformable Convolutions (MDC) [26]. This not only boosts performance and convergence time but presents a more unified deformable approach.

End-to-end training. The additional connections of our mask head in to the preceding encoder-decoder blocks, see colored lines in Figure 3, result in a significant performance boost for an end-to-end training of the full model, i.e., including the backbone and encoder-decoder. Without these connections, the authors of [5] did not observe any improvement for a similar end-to-end training.

3.5   Multi-cue clip tracking

For DeVIS to run on sequences with arbitrary length, we apply a near-online clip tracking/stitching similar to [11,2,3]. To this end, we sequentially match instance identities from consecutive and overlapping clips. The final set of tack predictions Υ is computed by matching the current Υ with the next Υ_k via the Hungarian algorithm [12]. Instances in Υ_k without any match in Υ start a new instance identity. For tracking-by-detection/online methods it is common to perform the data association with multiple cues [24]. We are the first to extend this idea to clip-level tracking and compute multiple cost terms for identities i and j in Υ and Υ_k, respectively:

- Mask cost via negative volumetric soft IoU [11] between consecutive overlapping sets of masks $m_{i,t}$ and $m_{j,t}$.
- Class cost $\mathcal{C}(c_i, c_j) = -1$ if $c_i = c_j$, and else 0, rewards consistent categories.
- Score cost $\mathcal{S}(s_i, s_j) = |s_i - s_j|$ matches clips with similar confidence.

The contribution of each cue is controlled by their corresponding weights σ_{mask}, σ_{class} and σ_{score}. In cases of strong occlusion and imperfect mask predictions, the identity matching can additionally rely on consistent class and score information which further improves the identity preservation across a sequence.

4   Experiments

This section provides the most relevant details of our implementation, and the experimental setup for the following ablation studies and benchmark evaluations. If not otherwise specified all results in the paper are obtained with a ResNet-50 [10] backbone and follow the hyperparameters of [27]. For additional implementation and training details we refer to the appendix.

Multi-scale deformable mask head. We train our mask head jointly with a pre-trained Deformable DETR [27] on COCO [16] for 24 epochs, decaying the learning by 0.1 after epoch 15. For batch size 2, we use 8 GPUs with 32GB memory for 2 days (345 GPU hours). The initial learning rates of the backbone, encoder-decoder, and mask head are $1e^{-5}$, $2e^{-5}$, and $2e^{-4}$, respectively.

DeVIS. We initialize our DeVIS model from the preceding end-to-end instance mask head training and then fine-tune for additional 10 epochs on the respective VIS dataset. With one clip per GPU, we use 4 GPUs for 1.5 days (120 GPU hours) with 18GB memory for YouTube-VIS 2019 [24] dataset. The increased number of objects per frame in both YouTube-VIS 2021 [24] and OVIS [19] require 24GB of memory. In contrast to [21,11], we are able to train with data augmentations on different input scales and apply a learned additive temporal encoding. If not otherwise specified, all models use clip size $\tau = 6$, stride $S = 4$ and reference point keys $K_{curr} = K_{temp} = 4$. We run multi-cue clip tracking with $\sigma_{mask} = \sigma_{class} = \sigma_{score} = 1$. For the ablations, we report best validation scores after training 10 epochs.

Datasets and metrics. We evaluate results on the *YouTube-VIS 2019/2021* [24] datasets which contain 2883 and 3859 high quality videos with 40 unique object categories, respectively. The latter provides improved ground truth annotations. The OVIS [19] dataset includes severe occlusion scenarios on 901 sequences. We measure the Average Precision (AP) as well as Average Recall (AR). For instance mask prediction on images, we report the AP based on mask IoU on the common COCO [16] dataset.

4.1   Ablation studies

We present ablations demonstrating the effectiveness of our contributions for DeVIS and provide detailed insights on the new mask head and clip tracking.

Making DeVIS work for VIS. In Table 1, we demonstrate the shortcomings of a naive deformabilization, i.e., replacement of VisTR's [21] full attention with deformable attention. We indicate full attention over all pixels wit $K_{temp} = $ All. Running temporal deformable attention offline, i.e., with clip size $\tau = 36$, reduces training time compared to VisTR, but converges with an unsatisfactory performance of 34.2. This is due to the incapability of spatially restricted reference points to compute meaningful temporal attention connections over large clip sizes. This assumption is supported by the 1.1 improvement of an offline Deformable VisTR without any temporal connections, i.e., $K_{temp} = 0$. A reduction of the clip size to $\tau = 6$ in a near-online fashion mitigates the reference point issues while also requiring only a fraction of the original training time (155 vs. 48

Table 1. Ablation of our main **DeVIS contributions** and the incremental built from a naive Deformable VisTR to our final model. *Increase spatial inputs* denotes multi-scale training on higher input resolutions.

Method	Clip size τ	K_{curr}	K_{temp}	Feature scales	AP	Δ AP	Training GPU hours	#params
VisTR	36	All	All	1	36.1	–	350	57M
	6	All	All	4	–	–	OOM	64M
Deformable VisTR	36	4	4	1	34.2	–	260	47M
	36	4	0	1	35.3	–	155	36M
	6	4	4	1	34.0	–	48	41M
	6	4	0	1	32.4	–	30	36M
DeVIS								
+ Increase spatial inputs	6	4	4	4	35.9	+1.9	93	48M
+ Instance-aware object queries	6	4	4	4	37.0	+1.1	112	48M
+ Multi-scale mask head	6	4	4	4	40.2	+3.2	120	48M
+ Multi-cue clip tracking	6	4	4	4	41.9	+1.7	120	48M
+ Auxiliary loss weighting	6	4	4	4	44.0	+2.1	120	48M

GPU hours). To further support our hypothesis, we demonstrate how removing temporal connections $K_{temp} = 0$ for smaller clip sizes $\tau = 6$ does indeed deteriorate the performance. However, without fully capitalizing on the efficiency of deformable attention its best version with 34.0 is still inferior to VisTR.

The first DeVIS row demonstrates the potential gains (1.9) from increasing the number of feature scales to $L = 4$ and training on higher input resolutions. The transition to multiple scales is a prerequisite for the application of our new mask head and only feasible to train on smaller clip sizes. The same model with full attention (second row) results in out-of-memory (OOM) errors. Naturally, this increases the total number of parameters and training time but both remain far below VisTR. Instance-aware object queries come with a neglectable increase in training time but provide a 1.1 AP boost. The individual contributions of the mask head and clip tracking additions are ablated in Table 2 and 5, respectively. Both result in additional performance boosts without substantially increasing the computational costs. A cascaded weighting of the decoder auxiliary loss terms as applied in [1] increases results further. Our final model benefits from deformable attention with low training times and parameter counts surpassing a naive Deformable VisTR approach by 11.1 points.

In Table 3, we ablate different clip sizes τ and number of temporal sampling keys K_{temp} for our final configuration. We used the optimal clip size $\tau = 6$ for our DeVIS ablations in Table 1. Both smaller and larger clip sizes resulted in worse performance either due to the lack of temporal connections or the aforementioned problem of spatially local reference points in the encoder, respectively. Running all $L = 4$ feature scales was only possible for a clip length of up to $\tau = 12$. Furthermore, we ablate the removal of temporal connections which resulted in a large relative drop for the challenging OVIS [19] dataset.

Multi-scale mask head. We evaluate our contributions on the mask head in Table 2 on COCO [16] instance segmentation. The baseline represents a straightforward application of the original DETR [5] mask head with the De-

Table 2. Ablation for the **multi-scale mask head** on COCO [16]. The baseline applies the original mask head as in DETR with Deformable DETR [27].

Mask head	Mask mAP	Training GPU hours
Baseline as in [5] with [27]	23.7	242
+ Train only positive matches	24.5	58
+ Encoded feature maps	25.0	56
+ Multi-scale attention maps	29.2	59
+ MDC	31.1	78
+ End-to-end training	38.0	345

Table 3. Removing **temporal connections** with $K_{temp} = 0$ results in performance drops across all datasets. We observe an optimal clip size of $\tau = 6$.

Clip size τ	K_{temp}	YT-VIS 19 [24]		YT-VIS 21 [24]		OVIS [19]	
		AP	Δ AP	AP	Δ AP	AP	Δ AP
6	4	44.4	–	43.1	–	23.8	–
6	0	41.2	-3.2	39.5	-3.6	19.7	-4.1
3	4	41.0	-3.4	–	–	–	–
9	4	42.4	-2.0	–	–	–	–
12	4	41.6	-2.8	–	–	–	–

Table 4. Comparison of instance segmentation results on **COCO** [16]. Mask R-CNN is from detectron2 [23].

Methods	AP	AP_{50}	AP_{75}	AP_l	AP_m	AP_s	FPS
DETR [5]	33.3	56.5	33.9	53.1	36.8	13.5	–
IFC [11]	35.1	–	–	–	–	–	–
Mask R-CNN [9]	37.2	58.5	39.8	53.3	39.4	18.6	21.4
Mask2Former [6]	43.7	–	–	64.8	47.2	23.4	13.5
Ours	38.0	61.4	40.1	59.8	41.4	17.9	12.1

Table 5. Contribution of the additional class and score cost terms in our **multi-cue clip tracking**.

Clip tracking cues	AP	AP_{50}	AP_{75}
Vol. soft mask IoU	42.1	63.2	46.7
Vol. soft mask IoU + Score	42.8	65.1	46.9
Vol. soft mask IoU + Class	43.1	66.0	47.2
Vol. soft mask IoU + Score + Class	44.4	66.8	48.5

formable DETR [27] detector, which is not only 9.6 points worse than [5] (see Table 4), but suffers from an unfeasible long training time largely due to the increased number of object queries in [27]. By computing instance masks only for queries positively matched with a ground truth object, we are able to reduce the training time 4-fold. The following two additions take full advantage of the encoded multi-scale features of [27] and result in a mask AP of 29.2. For top performance, we further add MDC [26] and train the entire model end-to-end. Our mask head without end-to-end training is still inferior to DETR. This can be attributed to the sparse computation of deformable attention which makes the generated attention maps less suitable for pixel-level segmentation. However, the end-to-end training fully realizes the potential of the additional connections between our mask head and the encoder-decoder. The increased training time is justified by the overall 4.7 point improvement over DETR.

Multi-cue clip tracking. To improve the track consistency between clips, we introduce additional cues to the common mask-based clip tracking. The clip tracking row in Table 1 replaces the original VisTR mask IoU cost term with a combination of volumetric soft mask IoU, class, and score costs. After tuning the cost weighting parameters, we ablate their individual contributions in Table 5. The new class and score terms provide an overall boost of 2.3 AP points.

4.2   Benchmark evaluation

Video instance segmentation. To demonstrate the effectiveness of DeVIS in comparison with other VIS methods, we report results on the YouTube-VIS 2019 and 2021 [24] dataset in Table 6. Our method achieves state-of-the-art performance with respect to all previous methods on the YouTube-VIS 2021 dataset by

Table 6. Comparison of VIS methods on the **YouTube-VIS 2019/2021** [24] validation sets. FPS measurements denoted with * are extracted from [8]. With ** and †the we denote a joint training with COCO [16] and unpublished methods, respectively.

Method	Backbone	YT-VIS 19 [24]						YT-VIS 21 [24]					OVIS [19]		
		FPS	AP	AP_{50}	AP_{75}	AR_1	AR_{10}	AP	AP_{50}	AP_{75}	AR_1	AR_{10}	AP	AP_{50}	AP_{75}
Online MaskTrack-RCNN [24]	R50	*26.1	30.3	51.1	32.6	31.0	35.5	28.6	48.9	29.6	26.5	33.8	15.4	33.9	13.1
SipMask [4]	R50	*35.5	33.7	54.1	35.8	35.4	40.1	31.7	52.5	34.0	30.8	37.8	14.3	29.9	12.5
SG-Net [17]	R50	*23.0	34.8	56.1	36.8	35.8	40.8	–	–	–	–	–	–	–	–
CompFeat [7]	R50	–	35.3	56.0	38.6	33.1	40.3	–	–	–	–	–	–	–	–
CrossVIS [25]	R50	39.8	36.3	56.8	38.9	35.6	40.7	34.2	54.4	37.9	30.4	38.2	18.1	35.5	16.9
STMask [13]	R50-DCN	28.6	33.5	52.1	36.9	31.1	39.2	30.6	49.4	32.0	26.4	36.0	–	–	–
VISOLO[8]	50	40.0	38.6	56.3	43.7	35.7	42.5	36.9	54.7	40.2	30.6	40.9	–	–	–
Offline VisTR [21]	R50	69.9	36.2	59.8	36.9	37.2	42.4	–	–	–	–	–	–	–	–
SeqMask-RCNN [14]	R50	*3.8	40.4	63.0	43.8	41.1	49.7	–	–	–	–	–	–	–	–
IFC [11]	R50	107.1	41.2	65.1	44.6	42.3	49.6	35.2	57.2	37.5	–	–	–	–	–
IFC [11]	R101	89.4	42.6	66.6	46.3	43.5	51.4	–	–	–	–	–	–	–	–
SeqFormer† [22]	R50	12	45.1	66.9	50.5	45.6	54.6	40.5	62.4	43.7	36.1	48.1	–		
SeqFormer†** [22]	Swin-L	–	59.3	82.1	66.4	51.7	64.4	51.8	74.6	58.2	42.8	58.1	–		
Mask2Former† [6]	R50	–	**46.4**	**68.0**	**50.5**	–	–	40.6	60.9	41.8	–	–	–	–	–
Mask2Former† [6]	Swin-L	–	**60.4**	**84.4**	**67.0**	–	–	52.6	76.4	57.2	–	–	–	–	–
Near-online MaskProp [3] (T=12)	R50	–	40.0	–	42.9	–	–	–	–	–	–	–	–	–	–
STEm-Seg [2]	R50	*3.0	34.6	55.8	37.9	34.4	41.6	–	–	–	–	–	13.8	32.1	11.9
IFC [11] (T=5, S=1)	R50	46.5	39.0	60.4	42.7	41.7	51.6	–	–	–	–	–	–	–	–
DeVIS (T=6, S=4)	R50	18.4	44.4	66.7	48.6	42.4	51.6	**43.1**	**66.8**	**46.6**	**38.0**	**50.1**	**23.8**	**48.0**	**20.8**
DeVIS (T=6, S=4)	SwinL	18.4	57.1	80.8	66.3	50.8	61.0	**54.4**	**77.7**	**59.8**	**43.8**	**57.8**	**34.6**	**58.7**	**36.8**

significant margins of 2.5 and 1.8 for ResNet-50 [10] and Swin-L [18] backbones, respectively. The benefits of our temporal connections and multi-scale feature encoding in conjunction with the mask head are most apparent by our 7.9 improvement over IFC [11]. Both VisTR and IFC achieve lower runtimes mirroring the relation between Deformable DETR [27] and DETR [5]. Furthermore, both methods rely on an expensively pretrained DETR limiting their adaptability dramatically. In addition to YouTube-VIS, we are the first Transformer-based method to present results on the OVIS [19] dataset. We surpass the previous best method [25] by 5.7 points for ResNet-50. Due to its pixel-level encoding of image features, DeVIS excels on OVIS' challenging occlusions scenarios.

Image instance segmentation. The new multi-scale mask head does not only boost VIS performance but also excels on image segmentation. In Table 4, we evaluate performance on COCO [16] even surpassing Mask R-CNN [9]. We present this as contribution beyond the VIS community and regard our mask head as a valuable completion of Deformable DETR. Interestingly, DeVIS is superior to Mask2Former despite the inferiority of our mask head for image segmentation. We regard this as a strong sentiment for our DeVIS video approach.

5   Conclusion

We have proposed a novel VIS method which applies a Transformer encoder-decoder architecture to clips of frames in a near-online fashion. To mitigate the efficiency issues of previous Transformer-based methods suffering from quadratic input complexity, we propose temporal deformable attention with instance-aware object queries. Deformable attention allows us to benefit from multi-scale feature maps and led to the introduction of a new powerful instance mask head. Furthermore, we present multi-cue tracking with class and score terms. Our DeVIS method achieves state-of-the-art results on two VIS datasets and hopefully paves the way for future applications of deformable attention for VIS.

Appendix

This section provides additional material for the main paper: §A contains further implementation details for our DeVIS method and its training. In §B, we discuss further ablations on the effect of the clip size and temporal sampling points. Furthermore, we complement the qualitative results of the main paper with selected illustrations in (§C). These include qualitative results, more detailed attention maps and failure cases.

A   Implementation details

Multi-scale mask head To improve convergence of the end-to-end full model training, we increase the dice and mask loss weights to $\lambda_{DICE} = \lambda_{MASK} = 8$. Furthermore, we add both loss terms to the auxiliary losses of the 3rd decoder layer. For trainings of the new mask head with DeVIS, we keep the original mask loss weights but also add the corresponding terms to the 3rd auxiliary loss. To further speed up the inference, we reduce the *top k* from 100 as in [27] to 50.

DeVIS We train our model with a total of 60 object queries for YouTube-VIS 2019 and 180 object queries for YouTube-VIS 2021 and OVIS. With a clip size of $\tau = 6$ this assigns 10 or 30 queries to each frame. OVIS includes sequences with up to 44 unique instances and hence requires an increased number of object queries. The class head is solely responsible for categorization of each object query. This means, the mask head predicts a single mask for each object query and does not produce per-category outputs as Mask R-CNN [9]. In order to associate a single class with a trajectory of queries, we compute the mean score over each class. This results in a total of number of classes times number of object queries per frame trajectories. The final output is selected in a top-k manner from the total set of trajectories. It should be noted, if k is larger than the number of queries per frame, a single trajectory is associated with multiple labels. To work as similar as possible to VisTR [21], our ablation experiments apply $k = 10$. For our benchmark experiments, we increase the value to $k = 20$ and $k = 30$ for YouTube-VIS 2019 and YouTube-VIS 2019/OVIS, respectively.

For the auxiliary loss weighting [1], we use the following incremental weights from the first to the final layer: 1/2, 5/30, 4/30, 3/30, 2/30 and 1/30. Such a weighting decreases the influence of early decoder layers with an emphasized focus on optimizing the outputs of the final layer. In contrast to [1], we keep the same weighting during the entire training. However, the weighting is not applied to the mask auxiliary loss on the third layer. As the total contribution from non-mask losses is heavily reduced, we use $\lambda_{DICE} = \lambda_{MASK} = 1$, since we did not observe any benefit in this case from increasing its contribution. Furthermore, we use $\lambda_{CLASS} = 1$ weight for the class cost and matching, following [21]. In comparison to [27], our encoder-decoder model introduces only changes to the dimensions of a few parameters, namely, object queries and linear projections for sample offset and attention weight. We train the object queries, the introduced temporal learned embedding and the classification head from scratch. For the

linear projections, we duplicate the existing pre-trained weights for each new temporal position, adapted to the number of points K_{temp}. We apply initial learning rates of $1e^{-5}$ and $1e^{-4}$ for the backbone, and rest of the model including the mask head, respectively. We drop these by 0.1 at epoch 3 and 7 for YouTube-VIS 2019, 4 and 8 for YouTube-VIS 2021 and 6 and 10 for OVIS.

Learnable sample offset and attention weight parameters To circumvent the quadratic input complexity of regular attention, deformable attention [27] learns linear projections which infer the sample offsets $\Delta \boldsymbol{p}_{mqk}$ and attention weights A_{mqk} for a query q, sampling point k and attention head m. In our temporal deformable attention formulation, we separately learn linear projections for the current and temporal frames, see Table A.1 for their dimensions. This allows to individually set K_{curr} and K_{temp} and ablate configurations without temporal connections. Given a clip with $\tau = 6$, the queries assigned to a frame t apply their $\tau - 1$ learned temporal projections to the remaining frames in the following frame order:

We follow VIS convention to run inference on reduced input resolutions with 360 pixels but upsample to the required benchmark resolution before the clip tracking. The runtime frames per second (FPS) measurement do not include this upsampling.

Query frame t	Temporal frame indices
0	[1, 2, 3, 4, 5]
1	[0, 2, 3, 4, 5]
2	[0, 1, 3, 4, 5]
3	[0, 1, 2, 4, 5]
4	[0, 1, 2, 3, 5]
5	[0, 1, 2, 3, 4]

This requires the same projection parameters to predict offsets/weights with different temporal distances to the current frame which is possible due to the learned temporal encoding added to each query. Furthermore, the explicit discrimination between the current and temporal frames with respect to a query allows decoder object queries to focus on predictions for their frame while taking additional temporal information under consideration. It should also be noted that all the terms in Table A.1 scale linearly with the number of frames in a clip and and do not depend on the input resolution.

Table A.1. Dimensions of learnable sample offset and attention weight parameters with object queries N, attention heads m, feature scales L and clip size τ.

	Current frame	Temporal frames
Attention weights	$N \times M \times L \times K_{curr} \times 1 \times 1$	$N \times m \times L \times K_{temp} \times \tau - 1 \times 1$
Sampling offsets	$N \times M \times L \times K_{curr} \times 1 \times 2$	$N \times m \times L \times K_{temp} \times \tau - 1 \times 2$

Table A.2. Removing **temporal connections** with $K_{temp} = 0$ results in performance drops across all datasets. We observe an optimal clip size of $\tau = 6$.

Clip size τ	K_{temp}	YT-VIS 19 [24]		YT-VIS 21 [24]		OVIS [19]	
		AP	Δ AP	AP	Δ AP	AP	Δ AP
6	4	44.4	–	43.1	–	23.8	–
3	4	41.0	-3.4	–	–	–	–
9	4	42.4	-2.0	–	–	–	–
12	4	41.6	-2.8	–	–	–	–
6	0	41.2	-3.2	39.5	-3.6	19.7	-4.1
6	1	41.2	-3.2	–	–	–	–
6	2	43.4	-1.0	–	–	–	–
6	3	43.6	-0.8	–	–	–	–

B   Ablation studies

In Table A.2, we extend the analysis of the main paper on the effect of adding more or less temporal information by altering the number of K_{temp} for the encoder and decoder separately. The ablation of different K_{temp} require individually trained models with differing number of parameters. Given a trained DeVIS model, the final runtime and performance can be modulated by adjusting the clip stride S, i.e., instance overlap, during inference. For experiments with differing clip size τ, we adjust the stride to keep the number of overlapping frames between constant. We set $K_{curr} = 4$ as in [27] for all experiments and obtain the same optimal value for K_{temp}.

C   Qualitative results

In Figure A.1 and A.2, we present additional qualitative results for the YouTube-VIS 2019/2021 and OVIS, respectively. In Figure A.5, we demonstrate the instance-aware object queries and their reference point alignment. To this end, we visualize the attention maps with reference points (cross) at the first, third and last layer for the query from the third frame. The reference point from that query on other frames aligns with the bounding box positions on these respective frames. Finally, we present failure cases in Figure A.3 and A.4.

Fig. A.1. Qualitative results on the YouTube-VIS 2019/2021 [24] datasets.

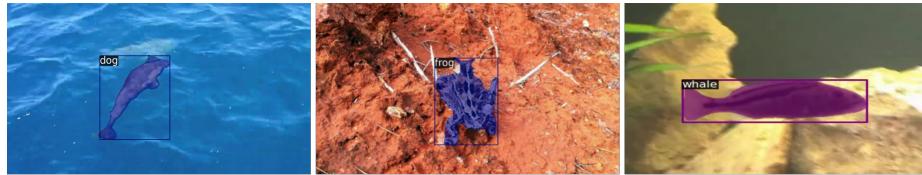**Fig. A.2.** Qualitative results on the OVIS [19] dataset.

Fig. A.3. Failure predictions due to the category. Most of our errors on YouTube-VIS 2019/2021 [24] datasets are from failed categories, specially the ones that are more under-represented on the training data.

Fig. A.4. Segmentation failure on YouTube-VIS 2021 [24] dataset. We also struggle sometimes to segment overlapping instances from the same category on YouTube-VIS dataset. We do a better job on similar scenarios A.2 on OVIS, and we argue it is because the model sees a lot more of these examples during training in this other dataset.

Fig. A.5. Visualization of the instance-aware object queries and their alignment of reference points for temporal frames on example sequences from [24]. To this end, we plot the not yet aligned reference points applied in the first decoder layer (first row) and their subsequent alignment to the predicted bounding box centers after the 2nd and 6th layer.

References

1. Al-Rfou, R., Choe, D., Constant, N., Guo, M., Jones, L.: Character-level language modeling with deeper self-attention. AAAI **33** (2019) 12, 15
2. Athar, A., Mahadevan, S., Ošep, A., Leal-Taixé, L., Leibe, B.: Stem-seg: Spatio-temporal embeddings for instance segmentation in videos. In: Eur. Conf. Comput. Vis. (2020) 1, 3, 10, 14
3. Bertasius, G., Torresani, L.: Classifying, segmenting, and tracking object instances in video with mask propagation. IEEE Conf. Comput. Vis. Pattern Recog. (2020) 1, 3, 10, 14
4. Cao, J., Anwer, R.M., Cholakkal, H., Khan, F.S., Pang, Y., Shao, L.: Sipmask: Spatial information preservation for fast image and video instance segmentation. Eur. Conf. Comput. Vis. (2020) 1, 3, 14
5. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. Eur. Conf. Comput. Vis. (2020) 2, 3, 5, 6, 8, 9, 10, 12, 13, 14
6. Cheng, B., Choudhuri, A., Misra, I., Kirillov, A., Girdhar, R., Schwing, A.G.: Mask2former for video instance segmentation (2021). https://doi.org/10.48550/ARXIV.2112.10764, https://arxiv.org/abs/2112.10764 13, 14
7. Fu, Y., Yang, L., Liu, D., Huang, T.S., Shi, H.: Compfeat: Comprehensive feature aggregation for video instance segmentation. AAAI (2021) 14
8. Han, S.H., Hwang, S., Oh, S.W., andHyunwoo Kim, Y.P., Kim, M., Kim, S.J.: VISOLO: grid-based space-time aggregation for efficient online video instance segmentation. CoRR (2021) 14
9. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: IEEE Conf. Comput. Vis. Pattern Recog. (2017) 1, 3, 13, 14, 15
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conf. Comput. Vis. Pattern Recog. (2016) 7, 11, 14
11. Hwang, S., Heo, M., Oh, S.W., Kim, S.J.: Video instance segmentation using inter-frame communication transformers. Adv. Neural Inform. Process. Syst. (2021) 2, 3, 8, 10, 11, 13, 14
12. Kuhn, H.W., Yaw, B.: The hungarian method for the assignment problem. Naval Res. Logist. Quart pp. 83–97 (1955) 10
13. Li, M., Li, S., Li, L., Zhang, L.: Spatial feature calibration and temporal fusion for effective one-stage video instance segmentation. In: CVPR (2021) 1, 3, 14
14. Lin, H., Wu, R., Liu, S., Lu, J., Jia, J.: Video instance segmentation with a propose-reduce paradigm. Int. Conf. Comput. Vis. (2021) 1, 3, 14
15. Lin, T.Y., Dollár, P., Girshick, R.B., He, K., Hariharan, B., Belongie, S.J.: Feature pyramid networks for object detection. IEEE Conf. Comput. Vis. Pattern Recog. (2017) 7
16. Lin, T.Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C.L., Dollár, P.: Microsoft coco: Common objects in context. arXiv:1405.0312 (2014) 11, 12, 13, 14
17. Liu, D., Cui, Y., Tan, W., Chen, Y.: Sg-net: Spatial granularity network for one-stage video instance segmentation. IEEE Conf. Comput. Vis. Pattern Recog. (2021) 1, 3, 14
18. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: ICCV (2021) 14

19. Qi, J., Gao, Y., Hu, Y., Wang, X., Liu, X., Bai, X., Belongie, S., Yuille, A., Torr, P., Bai, S.: Occluded video instance segmentation: A benchmark. arXiv preprint arXiv:2102.01558 (2021) 2, 11, 12, 13, 14, 17, 18

20. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. In: Adv. Neural Inform. Process. Syst. (2017) 2, 3, 5

21. Wang, Y., Xu, Z., Wang, X., Shen, C., Cheng, B., Shen, H., Xia, H.: End-to-end video instance segmentation with transformers. In: Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2021) 2, 3, 4, 5, 8, 11, 14, 15

22. Wu, J., Jiang, Y., Zhang, W., Bai, X., Bai, S.: Seqformer: a frustratingly simple model for video instance segmentation (2021). https://doi.org/10.48550/ARXIV.2112.08275, https://arxiv.org/abs/2112.08275 14

23. Wu, Y., Kirillov, A., Massa, F., Lo, W.Y., Girshick, R.: Detectron2. https://github.com/facebookresearch/detectron2 (2019) 13

24. Yang, L., Fan, Y., Xu, N.: Video instance segmentation. CoRR (2019) 1, 2, 3, 10, 11, 13, 14, 17, 18, 19

25. Yang, S., Fang, Y., Wang, X., Li, Y., Fang, C., Shan, Y., Feng, B., Liu, W.: Crossover learning for fast online video instance segmentation. In: Int. Conf. Comput. Vis. (October 2021) 1, 3, 14

26. Zhu, X., Hu, H., Lin, S., Dai, J.: Deformable convnets v2: More deformable, better results. arXiv preprint arXiv:1811.11168 (2018) 10, 13

27. Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. Int. Conf. Learn. Represent. (2021) 2, 5, 6, 7, 8, 11, 13, 14, 15, 16, 17