

# Fast Online Object Tracking and Segmentation: A Unifying Approach

Qiang Wang\*  
CASIA

qiang.wang@nlpr.ia.ac.cn

Li Zhang\*  
University of Oxford

lz@robots.ox.ac.uk

Luca Bertinetto\*  
FiveAI

luca@robots.ox.ac.uk

Weiming Hu  
CASIA

wmhu@nlpr.ia.ac.cn

Philip H.S. Torr  
University of Oxford

philip.torr@eng.ox.ac.uk

## Abstract

*In this paper we illustrate how to perform both visual object tracking and semi-supervised video object segmentation, in real-time, with a single simple approach. Our method, dubbed SiamMask, improves the offline training procedure of popular fully-convolutional Siamese approaches for object tracking by augmenting their loss with a binary segmentation task. Once trained, SiamMask solely relies on a single bounding box initialisation and operates online, producing class-agnostic object segmentation masks and rotated bounding boxes at 55 frames per second. Despite its simplicity, versatility and fast speed, our strategy allows us to establish a new state of the art among real-time trackers on VOT-2018, while at the same time demonstrating competitive performance and the best speed for the semi-supervised video object segmentation task on DAVIS-2016 and DAVIS-2017. The project website is <http://www.robots.ox.ac.uk/~qwang/SiamMask>.*

## 1. Introduction

Tracking is a fundamental task in any video application requiring some degree of reasoning about objects of interest, as it allows to establish object correspondences between frames [34]. It finds use in a wide range of scenarios such as automatic surveillance, vehicle navigation, video labelling, human-computer interaction and activity recognition. Given the location of an arbitrary target of interest in the first frame of a video, the aim of *visual object tracking* is to estimate its position in all the subsequent frames with the best possible accuracy [48].

For many applications, it is important that tracking can be performed *online*, while the video is streaming. In other words, the tracker should not make use of future frames to

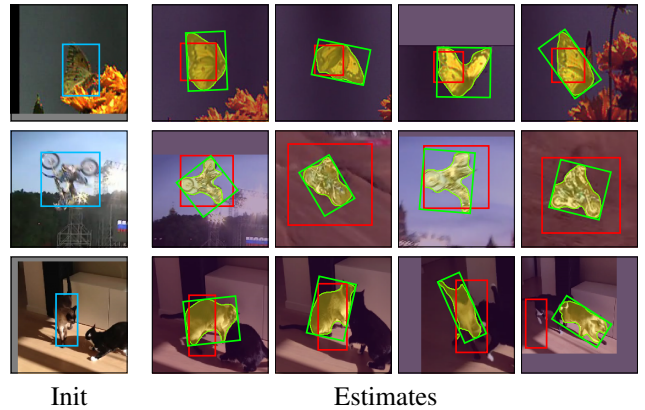


Figure 1. Our method aims at the intersection between the tasks of visual tracking and video object segmentation to achieve high practical convenience. Like conventional object trackers, it relies on a simple bounding box initialisation (blue) and operates online. Differently from state-of-the-art trackers such as ECO [12] (red), SiamMask (green) is able to produce binary segmentation masks, which can more accurately describe the target object.

reason about the current position of the object [26]. This is the scenario portrayed by visual object tracking benchmarks, which represent the target object with a simple axis-aligned (e.g. [56, 52]) or rotated [26, 27] bounding box. Such a simple annotation helps to keep the cost of data labelling low; what is more, it allows a user to perform a quick and simple initialisation of the target.

Similar to object tracking, the task of semi-supervised *video object segmentation* (VOS) requires estimating the position of an arbitrary target specified in the first frame of a video. However, in this case the object representation consists of a binary segmentation mask which expresses whether or not a pixel belongs to the target [40]. Such a detailed representation is more desirable for applications that require pixel-level information, like video editing [38] and rotoscoping [37]. Understandably, producing pixel-level estimates requires more computational re-

\*Equal contribution.

sources than a simple bounding box. As a consequence, VOS methods have been traditionally slow, often requiring several seconds per frame (e.g. [55, 50, 39, 1]). Very recently, there has been a surge of interest in faster approaches [59, 36, 57, 8, 7, 22, 21]. However, even the fastest still cannot operate in real-time.

In this paper, we aim at narrowing the gap between arbitrary object tracking and VOS by proposing *SiamMask*, a simple multi-task learning approach that can be used to address *both* problems. Our method is motivated by the success of fast tracking approaches based on fully-convolutional Siamese networks [3] trained offline on millions of pairs of video frames (e.g. [28, 63, 15, 60]) and by the very recent availability of YouTube-VOS [58], a large video dataset with pixel-wise annotations. We aim at retaining the offline trainability and online speed of these methods while at the same time significantly refining their representation of the target object, which is limited to a simple axis-aligned bounding box.

To achieve this goal, we simultaneously train a Siamese network on three tasks, each corresponding to a different strategy to establish correspondances between the target object and candidate regions in the new frames. As in the fully-convolutional approach of Bertinetto *et al.* [3], one task is to learn a measure of similarity between the target object and multiple candidates in a sliding window fashion. The output is a dense response map which only indicates the location of the object, without providing any information about its spatial extent. To refine this information, we simultaneously learn two further tasks: bounding box regression using a Region Proposal Network [46, 28] and class-agnostic binary segmentation [43]. Notably, binary labels are only required during offline training to compute the segmentation loss and *not* online during segmentation/tracking. In our proposed architecture, each task is represented by a different branch departing from a shared CNN and contributes towards a final loss, which sums the three outputs together.

Once trained, *SiamMask* solely relies on a single bounding box initialisation, operates online without updates and produces object segmentation masks and rotated bounding boxes at 55 frames per second. Despite its simplicity and fast speed, *SiamMask* establishes a new state-of-the-art on VOT-2018 for the problem of real-time object tracking. Moreover, the *same method* is also very competitive against recent semi-supervised VOS approaches on DAVIS-2016 and DAVIS-2017, while being the fastest by a large margin. This result is achieved with a simple bounding box initialisation (as opposed to a mask) and without adopting costly techniques often used by VOS approaches such as fine-tuning [35, 39, 1, 53], data augmentation [23, 30] and optical flow [50, 1, 39, 30, 8].

The rest of this paper is organised as follows. Section 2

briefly outlines some of the most relevant prior work in visual object tracking and semi-supervised VOS; Section 3 describes our proposal; Section 4 evaluates it on four benchmarks and illustrates several ablative studies; Section 5 concludes the paper.

## 2. Related Work

In this section, we briefly cover the most representative techniques for the two problems tackled in this paper.

**Visual object tracking.** Arguably, until very recently, the most popular paradigm for tracking arbitrary objects has been to train online a discriminative classifier exclusively from the ground-truth information provided in the first frame of a video (and then update it online).

In the past few years, the Correlation Filter, a simple algorithm that allows to discriminate between the template of an arbitrary target and its 2D translations, rose to prominence as particularly fast and effective strategy for tracking-by-detection thanks to the pioneering work of Bolme *et al.* [4]. Performance of Correlation Filter-based trackers has then been notably improved with the adoption of multi-channel formulations [24, 20], spatial constraints [25, 13, 33, 29] and deep features (e.g. [12, 51]).

Recently, a radically different approach has been introduced [3, 19, 49]. Instead of learning a discriminative classifier online, these methods train offline a similarity function on pairs of video frames. At test time, this function can be simply evaluated on a new video, once per frame. In particular, evolutions of the fully-convolutional Siamese approach [3] considerably improved tracking performance by making use of region proposals [28], hard negative mining [63], ensembling [15] and memory networks [60].

Most modern trackers, including all the ones mentioned above, use a rectangular bounding box both to initialise the target *and* to estimate its position in the subsequent frames. Despite its convenience, a simple rectangle often fails to properly represent an object, as it is evident in the examples of Figure 1. This motivated us to propose a tracker able to produce binary segmentation masks while still only relying on a bounding box initialisation.

Interestingly, in the past it was not uncommon for trackers to produce a coarse binary mask of the target object (e.g. [11, 42]). However, to the best of our knowledge, the only recent tracker that, like ours, is able to operate online and produce a binary mask starting from a bounding box initialisation is the superpixel-based approach of Yeo *et al.* [61]. However, at 4 frames per seconds (fps), its fastest variant is significantly slower than our proposal. Furthermore, when using CNN features, its speed is affected by a 60-fold decrease, plummeting below 0.1 fps. Finally, it has not demonstrated to be competitive on modern tracking or VOS benchmarks. Similar to us, the methods of Perazzi *et al.* [39] and Ci *et al.* [10] can also start from a rectangle and

output per-frame masks. However, they require fine-tuning at test time, which makes them slow.

**Semi-supervised video object segmentation.** Benchmarks for arbitrary object tracking (e.g. [48, 26, 56]) assume that trackers receive input frames in a sequential fashion. This aspect is generally referred to with the attributes *online* or *causal* [26]. Moreover, methods are often focused on achieving a speed that exceeds the ones of typical video framerates [27]. Conversely, semi-supervised VOS algorithms have been traditionally more concerned with an accurate representation of the object of interest [38, 40].

In order to exploit consistency between video frames, several methods propagate the supervisory segmentation mask of the first frame to the temporally adjacent ones via graph labeling approaches (e.g. [55, 41, 50, 36, 1]). In particular, Bao *et al.* [1] recently proposed a very accurate method that makes use of a spatio-temporal MRF in which temporal dependencies are modelled by optical flow, while spatial dependencies are expressed by a CNN.

Another popular strategy is to process video frames independently (e.g. [35, 39, 53]), similarly to what happens in most tracking approaches. For example, in OSVOS-S Maninis *et al.* [35] do not make use of any temporal information. They rely on a fully-convolutional network pre-trained for classification and then, at test time, they fine-tune it using the ground-truth mask provided in the first frame. MaskTrack [39] instead is trained from scratch on individual images, but it does exploit some form of temporality at test time by using the latest mask prediction and optical flow as additional input to the network.

Aiming towards the highest possible accuracy, at test time VOS methods often feature computationally intensive techniques such as fine-tuning [35, 39, 1, 53], data augmentation [23, 30] and optical flow [50, 1, 39, 30, 8]. Therefore, these approaches are generally characterised by low framerates and the inability to operate online. For example, it is not uncommon for methods to require minutes [39, 9] or even hours [50, 1] for videos that are just a few seconds long, like the ones of DAVIS.

Recently, there has been an increasing interest in the VOS community towards *faster* methods [36, 57, 8, 7, 22, 21]. To the best of our knowledge, the fastest approaches with a performance competitive with the state of the art are the ones of Yang *et al.* [59] and Wug *et al.* [57]. The former uses a meta-network “modulator” to quickly adapt the parameters of a segmentation network during test time, while the latter does not use any fine-tuning and adopts an encoder-decoder Siamese architecture trained in multiple stages. Both these methods run below 10 frames per second, while we are more than six times faster and only rely on a bounding box initialisation.

### 3. Methodology

To allow online operability and fast speed, we adopt the fully-convolutional Siamese framework [3]. Moreover, to illustrate that our approach is agnostic to the specific fully-convolutional method used as a starting point (e.g. [3, 28, 63, 60, 16]), we consider the popular SiamFC [3] and SiamRPN [28] as two representative examples. We first introduce them in Section 3.1 and then describe our approach in Section 3.2.

#### 3.1. Fully-convolutional Siamese networks

**SiamFC.** Bertinetto *et al.* [3] propose to use, as a fundamental building block of a tracking system, an offline-trained fully-convolutional Siamese network that compares an exemplar image  $z$  against a (larger) search image  $x$  to obtain a dense response map.  $z$  and  $x$  are, respectively, a  $w \times h$  crop centered on the target object and a larger crop centered on the last estimated position of the target. The two inputs are processed by the same CNN  $f_\theta$ , yielding two feature maps that are cross-correlated:

$$g_\theta(z, x) = f_\theta(z) \star f_\theta(x). \quad (1)$$

In this paper, we refer to each spatial element of the response map (left-hand side of Eq. 1) as *response of a candidate window* (**RoW**). For example,  $g_\theta^n(z, x)$ , encodes a *similarity* between the exemplar  $z$  and  $n$ -th candidate window in  $x$ . For SiamFC, the goal is for the maximum value of the response map to correspond to the target location in the search area  $x$ . Instead, in order to allow each RoW to encode richer information about the target object, we replace the simple cross-correlation of Eq. 1 with depth-wise cross-correlation [2] and produce a multi-channel response map. SiamFC is trained offline on millions of video frames with the logistic loss [3, Section 2.2], which we refer to as  $\mathcal{L}_{sim}$ .

**SiamRPN.** Li *et al.* [28] considerably improve the performance of SiamFC by relying on a region proposal network (RPN) [46, 14], which allows to estimate the target location with a bounding box of variable aspect ratio. In particular, in SiamRPN each RoW encodes a set of  $k$  anchor box proposals and corresponding object/background scores. Therefore, SiamRPN outputs box predictions in parallel with classification scores. The two output branches are trained using the smooth  $L_1$  and the cross-entropy losses [28, Section 3.2]. In the following, we refer to them as  $\mathcal{L}_{box}$  and  $\mathcal{L}_{score}$  respectively.

#### 3.2. SiamMask

Unlike existing tracking methods that rely on low-fidelity object representations, we argue the importance of producing per-frame binary segmentation masks. To this aim we show that, besides similarity scores and bounding box coordinates, it is possible for the RoW of a fully-

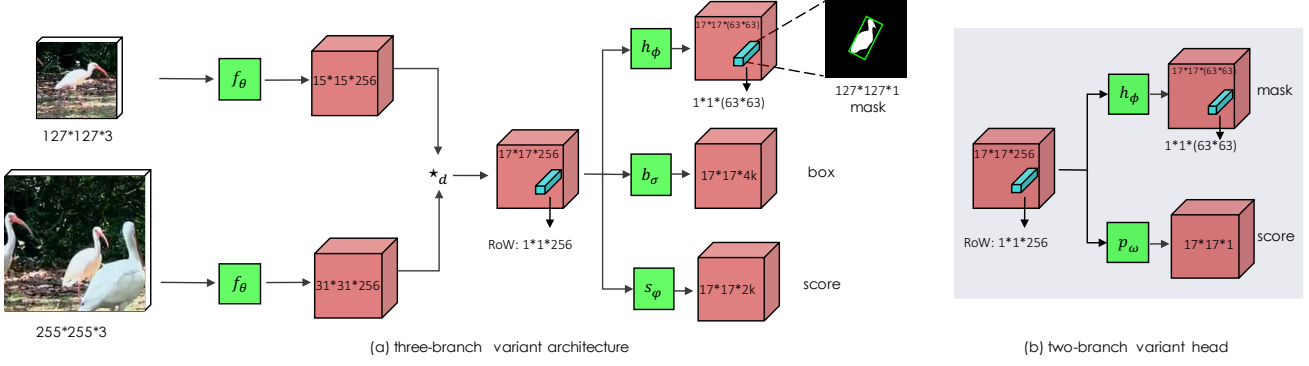


Figure 2. Schematic illustration of SiamMask variants: (a) *three-branch* architecture (full), (b) *two-branch* architecture (head).  $\star_d$  denotes depth-wise cross correlation. For simplicity, upsampling layer and mask refinement module are omitted here and detailed in Appendix A.

convolutional Siamese network to also encode the information necessary to produce a pixel-wise binary mask. This can be achieved by extending existing Siamese trackers with an extra branch and loss.

We predict  $w \times h$  binary masks (one for each RoW) using a simple two-layers neural network  $h_\phi$  with learnable parameters  $\phi$ . Let  $m_n$  denote the predicted mask corresponding to the  $n$ -th RoW,

$$m_n = h_\phi(g_\theta^n(z, x)). \quad (2)$$

From Eq. 2 we can see that the mask prediction is a function of *both* the image to segment  $x$  and the target object in  $z$ . In this way,  $z$  can be used as a reference to guide the segmentation process: given a different reference image, the network will produce a different segmentation mask for  $x$ .

**Loss function.** During training, each RoW is labelled with a ground-truth binary label  $y_n \in \{\pm 1\}$  and also associated with a pixel-wise ground-truth mask  $c_n$  of size  $w \times h$ . Let  $c_n^{ij} \in \{\pm 1\}$  denote the label corresponding to pixel  $(i, j)$  of the object mask in the  $n$ -th candidate RoW. The loss function  $\mathcal{L}_{mask}$  (Eq. 3) for the mask prediction task is a binary logistic regression loss over all RoWs:

$$\mathcal{L}_{mask}(\theta, \phi) = \sum_n \left( \frac{1 + y_n}{2wh} \sum_{ij} \log(1 + e^{-c_n^{ij} m_n^{ij}}) \right). \quad (3)$$

Thus, the classification layer of  $h_\phi$  consists of  $w \times h$  classifiers, each indicating whether a given pixel belongs to the object in the candidate window or not. Note that  $\mathcal{L}_{mask}$  is considered only for positive RoWs (*i.e.* with  $y_n = 1$ ).

**Mask representation.** In contrast to semantic segmentation methods in the style of FCN [32] and Mask R-CNN [17], which maintain explicit spatial information throughout the network, our approach follows the spirit of [43, 44] and generates masks starting from a flattened representation of the object. In particular, in our

case this representation corresponds to one of the  $(17 \times 17)$  RoWs produced by the depth-wise cross-correlation between  $f_\theta(z)$  and  $f_\theta(x)$ . Importantly, the network  $h_\phi$  of the segmentation task is composed of two  $1 \times 1$  convolutional layers, one with 256 and the other with  $63^2$  channels (Figure 2). This allows every pixel classifier to utilise information contained in the entire RoW and thus to have a complete view of its corresponding candidate window in  $x$ , which is critical to disambiguate between instances that look like the target (*e.g.* last row of Figure 4), often referred to as distractors. With the aim of producing a more accurate object mask, we follow the strategy of [44], which merges low and high resolution features using multiple *refinement* modules made of upsampling layers and skip connections (see Appendix A).

**Two variants.** For our experiments, we augment the architectures of SiamFC [3] and SiamRPN [28] with our segmentation branch and the loss  $\mathcal{L}_{mask}$ , obtaining what we call the *two-branch* and *three-branch* variants of SiamMask. These respectively optimise the multi-task losses  $\mathcal{L}_{2B}$  and  $\mathcal{L}_{3B}$ , defined as:

$$\mathcal{L}_{2B} = \lambda_1 \cdot \mathcal{L}_{mask} + \lambda_2 \cdot \mathcal{L}_{sim}, \quad (4)$$

$$\mathcal{L}_{3B} = \lambda_1 \cdot \mathcal{L}_{mask} + \lambda_2 \cdot \mathcal{L}_{score} + \lambda_3 \cdot \mathcal{L}_{box}. \quad (5)$$

We refer the reader to [3, Section 2.2] for  $\mathcal{L}_{sim}$  and to [28, Section 3.2] for  $\mathcal{L}_{box}$  and  $\mathcal{L}_{score}$ . For  $\mathcal{L}_{3B}$ , a RoW is considered positive ( $y_n = 1$ ) if one of its anchor boxes has IOU with the ground-truth box of at least 0.6 and negative ( $y_n = -1$ ) otherwise. For  $\mathcal{L}_{2B}$ , we adopt the same strategy of [3] to define positive and negative samples. We did not search over the hyperparameters of Eq. 4 and Eq. 5 and simply set  $\lambda_1 = 32$  like in [43] and  $\lambda_2 = \lambda_3 = 1$ . The task-specific branches for the box and score outputs are constituted by two  $1 \times 1$  convolutional layers. Figure 2 illustrates the two variants of SiamMask.

**Box generation.** Note that, while VOS benchmarks require binary masks, typical tracking benchmarks such as

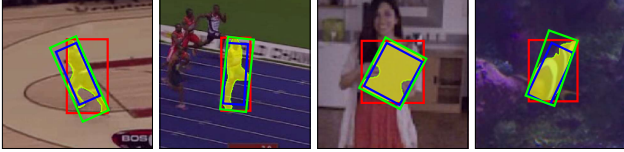


Figure 3. In order to generate a bounding box from a binary mask (in yellow), we experiment with three different methods. *Min-max*: the axis-aligned rectangle containing the object (red); *MBR*: the minimum bounding rectangle (green); *Opt*: the rectangle obtained via the optimisation strategy proposed in VOT-2016 [26] (blue).

VOT [26, 27] require a bounding box as final representation of the target object. We consider three different strategies to generate a bounding box from a binary mask (Figure 3): (1) axis-aligned bounding rectangle (*Min-max*), (2) rotated minimum bounding rectangle (*MBR*) and (3) the optimisation strategy used for the automatic bounding box generation proposed in VOT-2016 [26] (*Opt*). We empirically evaluate these alternatives in Section 4 (Table 1).

### 3.3. Implementation details

**Network architecture.** For both our variants, we use a ResNet-50 [18] until the final convolutional layer of the 4-th stage as our backbone  $f_\theta$ . In order to obtain a high spatial resolution in deeper layers, we reduce the output stride to 8 by using convolutions with stride 1. Moreover, we increase the receptive field by using dilated convolutions [6]. In our model, we add to the shared backbone  $f_\theta$  an unshared *adjust* layer ( $1 \times 1$  conv with 256 outputs). For simplicity, we omit it in Eq. 1. We describe the network architectures in more detail in Appendix A.

**Training.** Like SiamFC [3], we use exemplar and search image patches of  $127 \times 127$  and  $255 \times 255$  pixels respectively. During training, we randomly jitter exemplar and search patches. Specifically, we consider random translations (up to  $\pm 8$  pixels) and rescaling (of  $2^{\pm 1/8}$  and  $2^{\pm 1/4}$  for exemplar and search respectively).

The network backbone is pre-trained on the ImageNet-1k classification task. We use SGD with a first *warmup* phase in which the learning rate increases linearly from  $10^{-3}$  to  $5 \times 10^{-3}$  for the first 5 epochs and then decreases logarithmically until  $5 \times 10^{-4}$  for 15 more epochs. We train all our models using COCO [31], ImageNet-VID [47] and YouTube-VOS [58].

**Inference.** During tracking, SiamMask is simply evaluated once per frame, without any adaptation. In both our variants, we select the output mask using the location attaining the maximum score in the classification branch. Then, after having applied a per-pixel sigmoid, we binarise the output of the mask branch at the threshold of 0.5. In the *two-branch* variant, for each video frame after the first one, we fit the output mask with the *Min-max* box and use it as

reference to crop the next frame search region. Instead, in the *three-branch* variant, we find more effective to exploit the highest-scoring output of the box branch as reference.

## 4. Experiments

In this section, we evaluate our approach on two related tasks: visual object tracking (on VOT-2016 and VOT-2018) and semi-supervised video object segmentation (on DAVIS-2016 and DAVIS-2017). We refer to our *two-branch* and *three-branch* variants with SiamMask-2B and SiamMask respectively.

### 4.1. Evaluation for visual object tracking

**Datasets and settings.** We adopt two widely used benchmarks for the evaluation of the object tracking task: VOT-2016 [26] and VOT-2018 [27], both annotated with rotated bounding boxes. We use VOT-2016 to understand how different types of representation affect the performance. For this first experiment, we use mean intersection over union (IOU) and Average Precision (AP)@{0.5, 0.7} IOU. We then compare against the state-of-the-art on VOT-2018, using the official VOT toolkit and the Expected Average Overlap (EAO), a measure that considers both accuracy and robustness of a tracker [27].

#### How much does the object representation matter?

Existing tracking methods typically predict axis-aligned bounding boxes with a fixed [3, 20, 13, 33] or variable [28, 19, 63] aspect ratio. We are interested in understanding to which extent producing a per-frame binary mask can improve tracking. In order to focus on representation accuracy, for this experiment only we ignore the temporal aspect and sample video frames at random. The approaches described in the following paragraph are tested on randomly cropped search patches (with random shifts within  $\pm 16$  pixels and scale deformations up to  $2^{1 \pm 0.25}$ ) from the sequences of VOT-2016.

In Table 1, we compare our *three-branch* variant using the *Min-max*, *MBR* and *Opt* approaches (described at the end of Section 3.2 and in Figure 3). For reference, we also report results for SiamFC and SiamRPN as representative of the fixed and variable aspect-ratio approaches, together with three *oracles* that have access to per-frame ground-truth information and serve as upper bounds for the different representation strategies. (1) The fixed aspect-ratio oracle uses the per-frame ground-truth area and center location, but fixes the aspect ratio to the one of the first frame and produces an axis-aligned bounding box. (2) The *Min-max* oracle uses the minimal enclosing rectangle of the rotated ground-truth bounding box to produce an axis-aligned bounding box. (3) Finally, the *MBR* oracle uses the rotated minimum bounding rectangle of the ground-truth. Note that (1), (2) and (3) can be considered, respectively, the per-

	mIOU (%)	mAP@0.5 IOU	mAP@0.7 IOU
Fixed a.r. Oracle	73.43	90.15	62.52
<i>Min-max</i> Oracle	77.70	88.84	65.16
<i>MBR</i> Oracle	84.07	97.77	80.68
SiamFC [3]	50.48	56.42	9.28
SiamRPN [63]	60.02	76.20	32.47
<b>SiamMask-<i>Min-max</i></b>	65.05	82.99	43.09
<b>SiamMask-<i>MBR</i></b>	67.15	85.42	50.86
<b>SiamMask-<i>Opt</i></b>	<b>71.68</b>	<b>90.77</b>	<b>60.47</b>

Table 1. Performance for different bounding box representation strategies on VOT-2016.

formance upper bounds for the representation strategies of SiamFC, SiamRPN and SiamMask.

Table 1 shows that our method achieves the best mIOU, no matter the box generation strategy used (Figure 3). Albeit SiamMask-*Opt* offers the highest IOU and mAP, it requires significant computational resources due to its slow optimisation procedure [54]. SiamMask-*MBR* achieves a mAP@0.5 IOU of 85.4, with a respective improvement of +29 and +9.2 points w.r.t. the two fully-convolutional baselines. Interestingly, the gap significantly widens when considering mAP at the higher accuracy regime of 0.7 IOU: +41.6 and +18.4 respectively. Notably, our accuracy results are not far from the fixed aspect-ratio oracle. Moreover, comparing the upper bound performance represented by the oracles, it is possible to notice how, by simply changing the bounding box representation, there is a great room for improvement (e.g. +10.6% mIOU improvement between the fixed aspect-ratio and the *MBR* oracles).

Overall, this study shows how the *MBR* strategy to obtain a rotated bounding box from a binary mask of the object offers a significant advantage over popular strategies that simply report axis-aligned bounding boxes.

**Results on VOT-2018 and VOT-2016.** In Table 2 we compare the two variants of SiamMask with *MBR* strategy and SiamMask-*Opt* against five recently published state-of-the-art trackers on the VOT-2018 benchmark. Unless stated otherwise, SiamMask refers to our *three-branch* variant with *MBR* strategy. Both variants achieve outstanding performance and run in real-time. In particular, our *three-branch* variant significantly outperforms the very recent and top performing DaSiamRPN [63], achieving a EAO of 0.380 while running at 55 frames per second. Even without box regression branch, our simpler *two-branch* variant (SiamMask-2B) achieves a high EAO of 0.334, which is in par with SA\_Siam\_R [15] and superior to any other real-time method in the published literature. Finally, in SiamMask-*Opt*, the strategy proposed in [54] to find the optimal rotated rectangle from a binary mask brings the best overall performance (and a particularly high accuracy), but comes at a significant computational cost.

Our model is particularly strong under the accuracy met-

ric, showing a significant advantage with respect to the Correlation Filter-based trackers CSRDCF [33], STRCF [29]. This is not surprising, as SiamMask relies on a richer object representation, as outlined in Table 1. Interestingly, similarly to us, He *et al.* (SA\_Siam\_R) [15] are motivated to achieve a more accurate target representation by considering multiple rotated and rescaled bounding boxes. However, their representation is still constrained to a fixed aspect-ratio box.

Table 3 gives further results of SiamMask with different box generation strategies on VOT-2018 and -2016. SiamMask-box means the box branch of SiamMask is adopted for inference despite the mask branch has been trained. We can observe clear improvements on all evaluation metrics by using the mask branch for box generation.

## 4.2. Evaluation for semi-supervised VOS

Our model, once trained, can also be used for the task of VOS to achieve competitive performance without requiring any adaptation at test time. Importantly, differently to typical VOS approaches, ours can operate online, runs in real-time and only requires a simple bounding box initialisation.

**Datasets and settings.** We report the performance of SiamMask on DAVIS-2016 [40], DAVIS-2017 [45] and YouTube-VOS [58] benchmarks. For both DAVIS datasets, we use the official performance measures: the Jaccard index ( $\mathcal{J}$ ) to express region similarity and the F-measure ( $\mathcal{F}$ ) to express contour accuracy. For each measure  $\mathcal{C} \in \{\mathcal{J}, \mathcal{F}\}$ , three statistics are considered: mean  $\mathcal{C}_M$ , recall  $\mathcal{C}_O$ , and decay  $\mathcal{C}_D$ , which informs us about the gain/loss of performance over time [40]. Following Xu *et al.* [58], for YouTube-VOS we report the mean Jaccard index and F-measure for both seen ( $\mathcal{J}_S, \mathcal{F}_S$ ) and unseen categories ( $\mathcal{J}_U, \mathcal{F}_U$ ).  $\mathcal{O}$  is the average of these four measures.

To initialise SiamMask, we extract the axis-aligned bounding box from the mask provided in the first frame (*Min-max* strategy, see Figure 3). Similarly to most VOS methods, in case of multiple objects in the same video (DAVIS-2017) we simply perform multiple inferences.

**Results on DAVIS and YouTube-VOS.** In the semi-supervised setting, VOS methods are initialised with a binary mask [38] and many of them require computationally intensive techniques at test time such as fine-tuning [35, 39, 1, 53], data augmentation [23, 30], inference on MRF/CRF [55, 50, 36, 1] and optical flow [50, 1, 39, 30, 8]. As a consequence, it is not uncommon for VOS techniques to require several minutes to process a short sequence. Clearly, these strategies make the online applicability (which is our focus) impossible. For this reason, in our comparison we mainly concentrate on *fast* state-of-the-art approaches.

	SiamMask-Opt	SiamMask	SiamMask-2B	DaSiamRPN [63]	SiamRPN [28]	SA-Siam_R [15]	CSRDCF [33]	STRCF [29]
EAO $\uparrow$	<b>0.387</b>	<b>0.380</b>	0.334	0.326	0.244	0.337	0.263	0.345
Accuracy $\uparrow$	<b>0.642</b>	<b>0.609</b>	0.575	0.569	0.490	0.566	0.466	0.523
Robustness $\downarrow$	0.295	0.276	0.304	0.337	0.460	0.258	0.318	<b>0.215</b>
Speed (fps) $\uparrow$	5	55	60	160	<b>200</b>	32.4	48.9	2.9

Table 2. Comparison with the state-of-the-art under the EAO, Accuracy, and Robustness metrics on VOT-2018.

	VOT-2018			VOT-2016			
	EAO $\uparrow$	A $\uparrow$	R $\downarrow$	EAO $\uparrow$	A $\uparrow$	R $\downarrow$	Speed
SiamMask-box	0.363	0.584	0.300	0.412	0.623	0.233	<b>76</b>
SiamMask	0.380	0.609	<b>0.276</b>	0.433	0.639	<b>0.214</b>	55
SiamMask-Opt	<b>0.387</b>	<b>0.642</b>	0.295	<b>0.442</b>	<b>0.670</b>	0.233	5

Table 3. Results on VOT-2016 and VOT-2018.

	FT	M	$\mathcal{J}_M \uparrow$	$\mathcal{J}_O \uparrow$	$\mathcal{J}_D \downarrow$	$\mathcal{F}_M \uparrow$	$\mathcal{F}_O \uparrow$	$\mathcal{F}_D \downarrow$	Speed
OnAVOS [53]	✓	✓	<b>86.1</b>	<b>96.1</b>	5.2	<b>84.9</b>	<b>89.7</b>	5.8	0.08
MSK [39]	✓	✓	79.7	93.1	8.9	75.4	87.1	9.0	0.1
MSK <sub>b</sub> [39]	✓	✗	69.6	-	-	-	-	-	0.1
SFL [9]	✓	✓	76.1	90.6	12.1	76.0	85.5	10.4	0.1
FAVOS [8]	✗	✓	82.4	96.5	4.5	79.5	89.4	5.5	0.8
RGMP [57]	✗	✓	81.5	91.7	10.9	82.0	90.8	10.1	8
PML [7]	✗	✓	75.5	89.6	8.5	79.3	93.4	7.8	3.6
OSMN [59]	✗	✓	74.0	87.6	9.0	72.9	84.0	10.6	8.0
PLM [62]	✗	✓	70.2	86.3	11.2	62.5	73.2	14.7	6.7
VPN [22]	✗	✓	70.2	82.3	12.4	65.5	69.0	14.4	1.6
SiamMask	✗	✗	71.7	86.8	<b>3.0</b>	67.8	79.8	<b>2.1</b>	<b>55</b>

Table 4. Results on DAVIS 2016 (validation set). FT and M respectively denote if the method requires fine-tuning and whether it is initialised with a mask (✓) or a bounding box (✗).

	FT	M	$\mathcal{J}_M \uparrow$	$\mathcal{J}_O \uparrow$	$\mathcal{J}_D \downarrow$	$\mathcal{F}_M \uparrow$	$\mathcal{F}_O \uparrow$	$\mathcal{F}_D \downarrow$	Speed
OnAVOS [53]	✓	✓	<b>61.6</b>	<b>67.4</b>	27.9	<b>69.1</b>	<b>75.4</b>	26.6	0.1
OSVOS [5]	✓	✓	56.6	63.8	26.1	63.9	73.8	27.0	0.1
FAVOS [8]	✗	✓	54.6	61.1	<b>14.1</b>	61.8	72.3	<b>18.0</b>	0.8
OSMN [59]	✗	✓	52.5	60.9	21.5	57.1	66.1	24.3	8.0
SiamMask	✗	✗	54.3	62.8	19.3	58.5	67.5	20.9	<b>55</b>

Table 5. Results on DAVIS 2017 (validation set).

	FT	M	$\mathcal{J}_S \uparrow$	$\mathcal{J}_U \uparrow$	$\mathcal{F}_S \uparrow$	$\mathcal{F}_U \uparrow$	$\mathcal{O} \uparrow$	Speed
OnAVOS [53]	✓	✓	60.1	46.6	<b>62.7</b>	51.4	55.2	0.1
OSVOS [5]	✓	✓	59.8	<b>54.2</b>	60.5	<b>60.7</b>	<b>58.8</b>	0.1
OSMN [59]	✗	✓	60.0	40.6	60.1	44.0	51.2	8.0
SiamMask	✗	✗	<b>60.2</b>	45.1	58.2	47.7	52.8	<b>55</b>

Table 6. Results on YouTube-VOS (validation set).

Table 4, 5 and 6 show how SiamMask can be considered as a strong baseline for online VOS. First, it is almost two orders of magnitude faster than accurate approaches such as OnAVOS [53] or SFL [9]. Second, it is competitive with recent VOS methods that do not employ fine-tuning, while being four times more efficient than the fastest ones (*i.e.* OSMN [59] and RGMP [57]). Interestingly, we note that

	AN	RN	EAO $\uparrow$	$\mathcal{J}_M \uparrow$	$\mathcal{F}_M \uparrow$	Speed
SiamFC	✓		0.188	-	-	86
SiamFC		✓	0.251	-	-	40
SiamRPN	✓		0.243	-	-	<b>200</b>
SiamRPN		✓	0.359	-	-	76
SiamMask-2B w/o R		✓	0.326	62.3	55.6	43
SiamMask w/o R		✓	0.375	68.6	57.8	58
SiamMask-2B-score		✓	0.265	-	-	40
SiamMask-box		✓	0.363	-	-	76
SiamMask-2B		✓	0.334	67.4	63.5	60
SiamMask		✓	<b>0.380</b>	<b>71.7</b>	<b>67.8</b>	55

Table 7. Ablation studies on VOT-2018 and DAVIS-2016.

SiamMask achieves a very low *decay* [40] for both region similarity ( $\mathcal{J}_D$ ) and contour accuracy ( $\mathcal{F}_D$ ). This suggests that our method is robust over time and thus it is indicated for particularly long sequences.

Qualitative results of SiamMask for both VOT and DAVIS sequences are shown in Figure 4, 9 and 10. Despite the high speed, SiamMask produces accurate segmentation masks even in presence of distractors.

### 4.3. Further analysis

In this section, we illustrate ablation studies, failure cases and timings of our methods.

**Network architecture.** In Table 7, AN and RN denote whether we use AlexNet or ResNet-50 as the shared backbone  $f_\theta$  (Figure 2), while with “w/o R” we mean that the method does *not* use the refinement strategy of Pinheiro *et al.* [44]. From the results of Table 7, it is possible to make several observations. (1) The first set of rows shows that, by simply updating the architecture of  $f_\theta$ , it is possible to achieve an important performance improvement. However, this comes at the cost of speed, especially for SiamRPN. (2) SiamMask-2B and SiamMask considerably improve over their baselines (with same  $f_\theta$ ) SiamFC and SiamRPN. (3) Interestingly, the refinement approach of Pinheiro *et al.* [44] is very important for the contour accuracy  $\mathcal{F}_M$ , but less so for the other metrics.

**Multi-task training.** We conducted two further experiments to disentangle the effect of multi-task training. Results are reported in Table 7. To achieve this, we modified the two variants of SiamMask during inference so that, respectively, they report an axis-aligned bounding box from the score branch (SiamMask-2B-score) or the box branch (SiamMask-box). Therefore, despite having been trained,



Figure 4. **Qualitative results** of our method for sequences belonging to both object tracking and video object segmentation benchmarks. *Basketball* and *Nature* are from VOT-2018 [27]; *Car-Shadow* is from DAVIS-2016 [40]; *Dogs-Jump* and *Pigs* are from DAVIS-2017 [45]. Multiple masks are obtained from different inferences (with different initialisations).

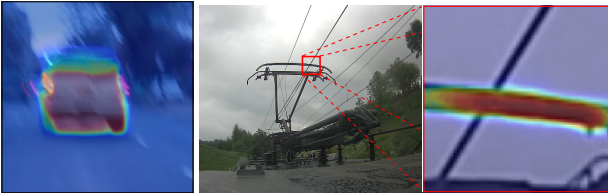


Figure 5. Failure cases: motion blur and ‘non-object’ instance.

the mask branch is *not* used during inference. We can observe how both variants obtain a modest but meaningful improvement with respect to their counterparts (SiamFC and SiamRPN): from 0.251 to 0.265 EAO for the *two-branch* and from 0.359 to 0.363 for the *three-branch* on VOT2018.

**Timing.** SiamMask operates online without any adaptation to the test sequence. On a single NVIDIA RTX 2080 GPU, we measured an average speed of 55 and 60 frames per second, respectively for the *two-branch* and *three-branch* variants. Note that the highest computational burden comes from the feature extractor  $f_\theta$ .

**Failure cases.** Finally, we discuss two scenarios in which SiamMask fails: motion blur and ‘non-object’ instance (Figure 5). Despite being different in nature, these two cases arguably arise from the complete lack of similar training samples in a training sets, which are focused on objects

that can be unambiguously discriminated from the foreground.

## 5. Conclusion

In this paper we introduced SiamMask, a simple approach that enables fully-convolutional Siamese trackers to produce class-agnostic binary segmentation masks of the target object. We show how it can be applied with success to both tasks of visual object tracking *and* semi-supervised video object segmentation, showing better accuracy than state-of-the-art trackers and, at the same time, the fastest speed among VOS methods. The two variants of SiamMask we proposed are initialised with a simple bounding box, operate online, run in real-time and do not require any adaptation to the test sequence. We hope that our work will inspire further studies that consider the two problems of visual object tracking and video object segmentation together.

**Acknowledgements.** This work was supported by the ERC grant ERC-2012-AdG 321162-HELIOS, EPSRC grant Seebibyte EP/M013774/1 and EPSRC/MURI grant EP/N019474/1. We would also like to acknowledge the support of the Royal Academy of Engineering and FiveAI Ltd. Qiang Wang is partly supported by the NSFC (Grant No. 61751212, 61721004 and U1636218).

## References

- [1] L. Bao, B. Wu, and W. Liu. Cnn in mrf: Video object segmentation via inference in a cnn-based higher-order spatio-temporal mrf. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2, 3, 6
- [2] L. Bertinetto, J. F. Henriques, J. Valmadre, P. H. S. Torr, and A. Vedaldi. Learning feed-forward one-shot learners. In *Advances in Neural Information Processing Systems*, 2016. 3
- [3] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr. Fully-convolutional siamese networks for object tracking. In *European Conference on Computer Vision workshops*, 2016. 2, 3, 4, 5, 6
- [4] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui. Visual object tracking using adaptive correlation filters. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010. 2
- [5] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool. One-shot video object segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 7
- [6] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. 5, 11
- [7] Y. Chen, J. Pont-Tuset, A. Montes, and L. Van Gool. Blazingly fast video object segmentation with pixel-wise metric learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2, 3, 7
- [8] J. Cheng, Y.-H. Tsai, W.-C. Hung, S. Wang, and M.-H. Yang. Fast and accurate online video object segmentation via tracking parts. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2, 3, 6, 7
- [9] J. Cheng, Y.-H. Tsai, S. Wang, and M.-H. Yang. Segflow: Joint learning for video object segmentation and optical flow. In *IEEE International Conference on Computer Vision*, 2017. 3, 7
- [10] H. Ci, C. Wang, and Y. Wang. Video object segmentation by learning location-sensitive embeddings. In *European Conference on Computer Vision*, 2018. 2
- [11] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2000. 2
- [12] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg. Eco: Efficient convolution operators for tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1, 2
- [13] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg. Learning spatially regularized correlation filters for visual tracking. In *IEEE International Conference on Computer Vision*, 2015. 2, 5
- [14] C. Feichtenhofer, A. Pinz, and A. Zisserman. Detect to track and track to detect. In *IEEE International Conference on Computer Vision*, 2017. 3
- [15] A. He, C. Luo, X. Tian, and W. Zeng. Towards a better match in siamese network based visual object tracker. In *European Conference on Computer Vision workshops*, 2018. 2, 6, 7
- [16] A. He, C. Luo, X. Tian, and W. Zeng. A twofold siamese network for real-time object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 3
- [17] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask rcnn. In *IEEE International Conference on Computer Vision*, 2017. 4
- [18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 5, 11
- [19] D. Held, S. Thrun, and S. Savarese. Learning to track at 100 fps with deep regression networks. In *European Conference on Computer Vision*, 2016. 2, 5
- [20] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015. 2, 5
- [21] Y.-T. Hu, J.-B. Huang, and A. G. Schwing. Videomatch: Matching based video object segmentation. In *European Conference on Computer Vision*, 2018. 2, 3
- [22] V. Jampani, R. Gadde, and P. V. Gehler. Video propagation networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2, 3, 7
- [23] A. Khoreva, R. Benenson, E. Ilg, T. Brox, and B. Schiele. Lucid data dreaming for object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition workshops*, 2017. 2, 3, 6
- [24] H. Kiani Galoogahi, T. Sim, and S. Lucey. Multi-channel correlation filters. In *IEEE International Conference on Computer Vision*, 2013. 2
- [25] H. Kiani Galoogahi, T. Sim, and S. Lucey. Correlation filters with limited boundaries. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 2
- [26] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Čehovin, T. Vojir, G. Häger, A. Lukežič, G. Fernández, et al. The visual object tracking vot2016 challenge results. In *European Conference on Computer Vision*, 2016. 1, 3, 5
- [27] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. C. Zajc, T. Vojir, G. Bhat, A. Lukežič, A. Eldesokey, G. Fernandez, and et al. The sixth visual object tracking vot-2018 challenge results. In *European Conference on Computer Vision workshops*, 2018. 1, 3, 5, 8, 12
- [28] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High performance visual tracking with siamese region proposal network. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2, 3, 4, 5, 7
- [29] F. Li, C. Tian, W. Zuo, L. Zhang, and M.-H. Yang. Learning spatial-temporal regularized correlation filters for visual tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2, 6, 7
- [30] X. Li and C. C. Loy. Video object segmentation with joint re-identification and attention-aware mask propagation. In *European Conference on Computer Vision*, 2018. 2, 3, 6
- [31] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 2014. 5

- [32] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 4
- [33] A. Lukezic, T. Vojir, L. C. Zajc, J. Matas, and M. Kristan. Discriminative correlation filter with channel and spatial reliability. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2, 5, 6, 7
- [34] T. Makovski, G. A. Vazquez, and Y. V. Jiang. Visual learning in multiple-object tracking. *PLoS One*, 2008. 1
- [35] K.-K. Maninis, S. Caelles, Y. Chen, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool. Video object segmentation without temporal information. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 2, 3, 6
- [36] N. Märki, F. Perazzi, O. Wang, and A. Sorkine-Hornung. Bilateral space video segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 2, 3, 6
- [37] O. Miksik, J.-M. Pérez-Rúa, P. H. Torr, and P. Pérez. Roam: a rich object appearance model with application to rotoscoping. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1
- [38] F. Perazzi. *Video Object Segmentation*. PhD thesis, ETH Zurich, 2017. 1, 3, 6
- [39] F. Perazzi, A. Khoreva, R. Benenson, B. Schiele, and A. Sorkine-Hornung. Learning video object segmentation from static images. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2, 3, 6, 7
- [40] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1, 3, 6, 7, 8, 13
- [41] F. Perazzi, O. Wang, M. Gross, and A. Sorkine-Hornung. Fully connected object proposals for video segmentation. In *IEEE International Conference on Computer Vision*, 2015. 3
- [42] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet. Color-Based Probabilistic Tracking. In *European Conference on Computer Vision*, 2002. 2
- [43] P. O. Pinheiro, R. Collobert, and P. Dollár. Learning to segment object candidates. In *Advances in Neural Information Processing Systems*, 2015. 2, 4
- [44] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár. Learning to refine object segments. In *European Conference on Computer Vision*, 2016. 4, 7, 11
- [45] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017. 6, 8, 13
- [46] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, 2015. 2, 3
- [47] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 2015. 5
- [48] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014. 1, 3
- [49] R. Tao, E. Gavves, and A. W. Smeulders. Siamese instance search for tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 2
- [50] Y.-H. Tsai, M.-H. Yang, and M. J. Black. Video segmentation via object flow. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 2, 3, 6
- [51] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. S. Torr. End-to-end representation learning for correlation filter based tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2
- [52] J. Valmadre, L. Bertinetto, J. F. Henriques, R. Tao, A. Vedaldi, A. Smeulders, P. H. S. Torr, and E. Gavves. Long-term tracking in the wild: A benchmark. In *European Conference on Computer Vision*, 2018. 1
- [53] P. Voigtlaender and B. Leibe. Online adaptation of convolutional neural networks for video object segmentation. In *British Machine Vision Conference*, 2017. 2, 3, 6, 7
- [54] T. Vojir and J. Matas. Pixel-wise object segmentations for the vot 2016 dataset. *Research Report CTU-CMP-2017-01, Center for Machine Perception, Czech Technical University, Prague, Czech Republic*, 2017. 6
- [55] L. Wen, D. Du, Z. Lei, S. Z. Li, and M.-H. Yang. Jots: Joint online tracking and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 2, 3, 6
- [56] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 1, 3
- [57] S. Wug Oh, J.-Y. Lee, K. Sunkavalli, and S. Joo Kim. Fast video object segmentation by reference-guided mask propagation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2, 3, 7
- [58] N. Xu, L. Yang, Y. Fan, J. Yang, D. Yue, Y. Liang, B. Price, S. Cohen, and T. Huang. Youtube-vos: Sequence-to-sequence video object segmentation. In *European Conference on Computer Vision*, 2018. 2, 5, 6
- [59] L. Yang, Y. Wang, X. Xiong, J. Yang, and A. K. Katsaggelos. Efficient video object segmentation via network modulation. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2018. 2, 3, 7
- [60] T. Yang and A. B. Chan. Learning dynamic memory networks for object tracking. In *European Conference on Computer Vision*, 2018. 2, 3
- [61] D. Yeo, J. Son, B. Han, and J. H. Han. Superpixel-based tracking-by-segmentation using markov chains. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2
- [62] J. S. Yoon, F. Rameau, J. Kim, S. Lee, S. Shin, and I. S. Kweon. Pixel-level matching for video object segmentation using convolutional neural networks. In *IEEE International Conference on Computer Vision*, 2017. 7
- [63] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu. Distractor-aware siamese networks for visual object tracking. In *European Conference on Computer Vision*, 2018. 2, 3, 5, 6, 7

## A. Architectural details

**Network backbone.** Table 8 illustrates the details of our *backbone* architecture ( $f_\theta$  in the main paper). For both variants, we use a ResNet-50 [18] until the final convolutional layer of the 4-th stage. In order to obtain a higher spatial resolution in deep layers, we reduce the output stride to 8 by using convolutions with stride 1. Moreover, we increase the receptive field by using dilated convolutions [6]. Specifically, we set the stride to 1 and the dilation rate to 2 in the  $3 \times 3$  conv layer of `conv4_1`. Differently to the original ResNet-50, there is no downsampling in `conv4_x`. We also add to the backbone an *adjust* layer (a  $1 \times 1$  convolutional layer with 256 output channels). Exemplar and search patches share the network’s parameters from `conv1` to `conv4_x`, while the parameters of the *adjust* layer are not shared. The output features of the adjust layer are then depth-wise cross-correlated, resulting a feature map of size  $17 \times 17$ .

**Network heads.** The network architecture of the branches of both variants are shown in Table 9 and 10. The `conv5` block in both variants contains a normalisation layer and ReLU non-linearity while `conv6` only consists of a  $1 \times 1$  convolutional layer.

**Mask refinement module.** With the aim of producing a more accurate object mask, we follow the strategy of [44], which merges low and high resolution features using multiple *refinement* modules made of upsampling layers and skip connections. Figure 6 gives an example of refinement module  $U_3$ , while Figure 8 illustrates how a mask is generated with stacked refinement modules.

block	exemplar output size	search output size	backbone
conv1	$61 \times 61$	$125 \times 125$	$7 \times 7, 64$ , stride 2
conv2_x	$31 \times 31$	$63 \times 63$	$3 \times 3$ max pool, stride 2 $\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3_x	$15 \times 15$	$31 \times 31$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$
conv4_x	$15 \times 15$	$31 \times 31$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$
adjust	$15 \times 15$	$31 \times 31$	$1 \times 1, 256$
xcorr	$17 \times 17$		depth-wise

Table 8. Backbone architecture. Details of each building block are shown in square brackets.

## B. Further qualitative results

**Different masks at different locations.** Our model generates a mask for each RoW. During inference, we rely on the

block	score	box	mask
conv5	$1 \times 1, 256$	$1 \times 1, 256$	$1 \times 1, 256$
conv6	$1 \times 1, 2k$	$1 \times 1, 4k$	$1 \times 1, (63 \times 63)$

Table 9. Architectural details of the *three-branch* head.  $k$  denotes the number of anchor boxes per RoW.

block	score	mask
conv5	$1 \times 1, 256$	$1 \times 1, 256$
conv6	$1 \times 1, 1$	$1 \times 1, (63 \times 63)$

Table 10. Architectural details of the *two-branch* head.

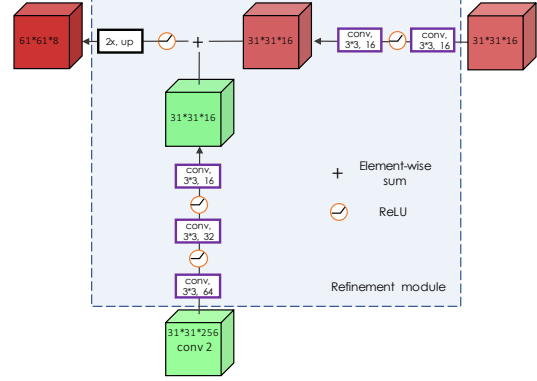


Figure 6. Example of a refinement module  $U_3$ .

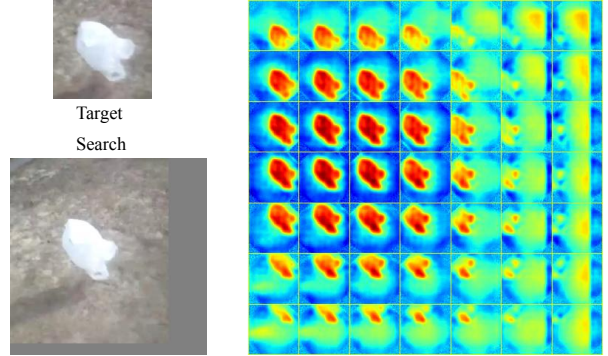


Figure 7. Score maps from the mask branch at different locations.

score branch to select the final output mask (using the location attaining the maximum score). The example of Figure 7 illustrates the multiple output masks produced by the mask branch, each corresponding to a different RoW.

**Benchmark sequences.** More qualitative results for VOT and DAVIS sequences are shown in Figure 9 and 10.

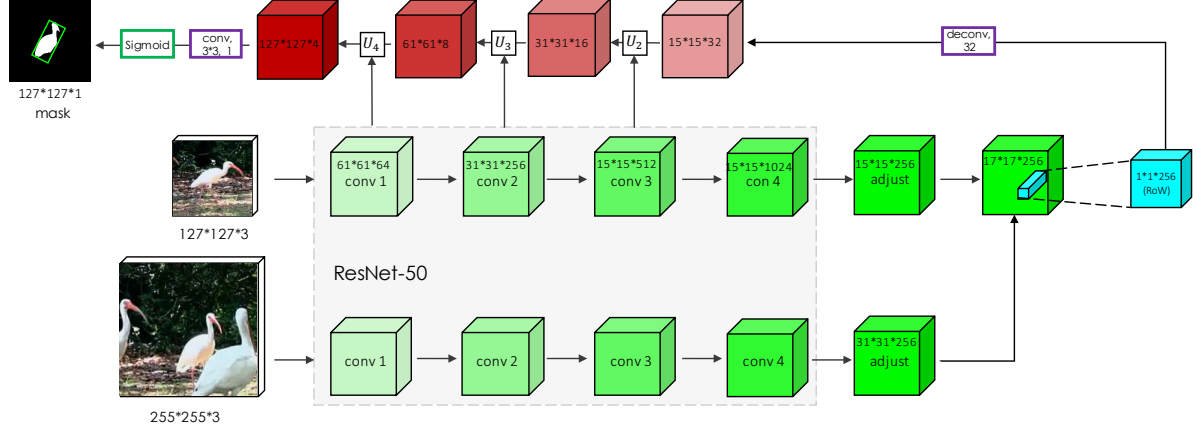


Figure 8. Schematic illustration of the stacked refinement modules.



Figure 9. Further qualitative results of our method on sequences from the visual object tracking benchmark VOT-2018 [27].

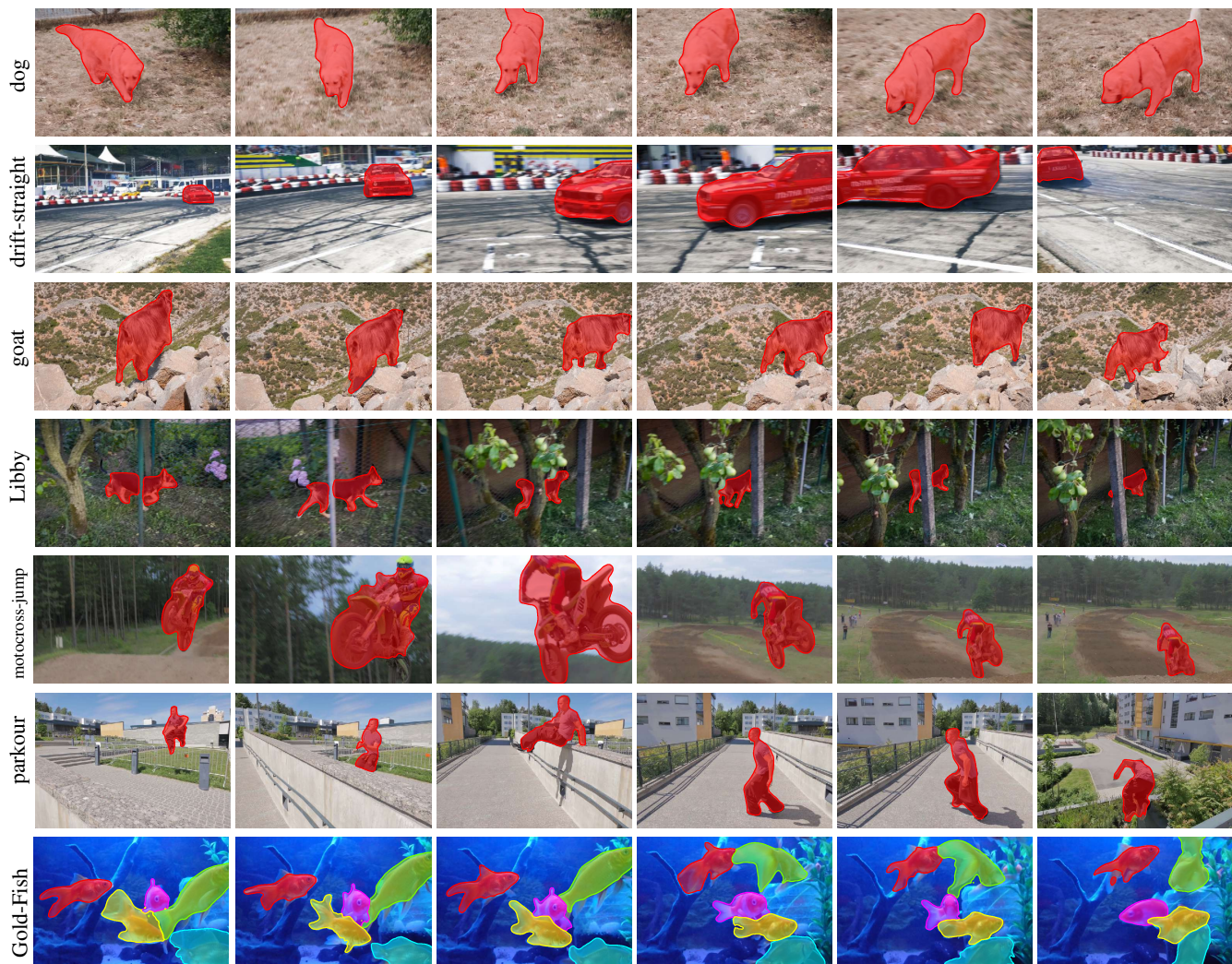


Figure 10. Further qualitative results of our method on sequences from the semi-supervised video object segmentation benchmarks DAVIS-2016 [40] and DAVIS-2017 [45]. Multiple masks are obtained from different inferences (with different initialisations).