# Supplementary Material for Learning Fast and Robust Target Models for Video Object Segmentation

Andreas Robinson[1]* Felix Järemo Lawin[1]* Martin Danelljan[2]
Fahad Shahbaz Khan[1,3] Michael Felsberg[1]
[1]CVL, Linköping University, Sweden [2]CVL, ETH Zurich, Switzerland [3]IIAI, UAE

In this supplementary material we provide a description of the initial sample generation, referred in section 3.1 in the paper. We also provide more detailed quantitative results and ablative analysis of parameters.

## 1. Initial sample generation

In the first frame of the sequence, we train the target model on the initial dataset $\mathcal{M}_0$, created from the given target mask $\mathbf{y}_0$ and the extracted features $\mathbf{x}_0$. To add more variety in the initial frame, we generate additional augmented samples. Based on the initial label $\mathbf{y}_0$, we cut out the target object and apply a fast inpainting method [3] to restore the background. We then apply a random affine warp and blur before pasting the target object back onto the image, creating a set of augmented images $\tilde{\mathbf{I}}_k$ and corresponding label masks $\tilde{\mathbf{y}}_k$. After feature extraction, we insert the unmodified first frame and the augmented frames into the dataset $\mathcal{M}_0 = \{(\tilde{\mathbf{x}}_k, \tilde{\mathbf{y}}_k, \gamma_k)\}_{k=0}^{K-1}$ and set the sample weights $\gamma_k$ such that the original sample carries twice the weight of the other samples. Example augmentations performed in the initial frame are shown in Figure 1.

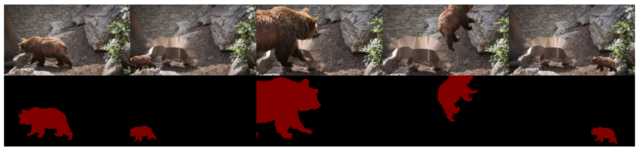
Figure 1. Example of data augmentation for the initial sample generation with the image (top row) and corresponding label mask (bottom row). The original first frame sample is shown to the left and the augmented samples follows from left to right.

## 2. Detailed Quantitative Results

In this section we report some additional quantitative results.

*Authors contributed equally.

| Method | $\mathcal{G}$ overall | $\mathcal{J}$ seen \| unseen | $\mathcal{F}$ seen \| unseen | data |
|---|---|---|---|---|
| Ours | 72.1 | 72.3 \| 65.9 | 76.2 \| 74.1 | 100% |
| Ours | 70.6 | 71.4 \| 63.7 | 75.5 \| 71.8 | 50% |
| Ours | 66.7 | 69.7 \| 58.5 | 73.0 \| 65.6 | 25% |
| Ours D-only | 59.9 | 60.1 \| 57.0 | 58.6 \| 63.8 | 0% |

Table 1. YouTubeVos 2018 test-dev results for different amount of training data, sample. Ours with 100% data is the same instance as in the comparison in Table 2 in the main paper. The Ours D-only is our approach without the segmentation network as described in Section 5.1 in the main paper. It thus requires no training data at all.

### 2.1. Training data

We analyze how the amount of training data impacts the performance of our approach. For this purpose we train our model on subsets of the YouTube-VOS training set containing 100%, 50%, 25% and 0% of the YouTube-VOS 2018 training split (excluding the validation split used to analyzing our approach as in Section 4 in the paper). For the version using 0% of the data, called "Ours D-only", we only apply or target appearance model, which is trained during inference, thus requiring no offline training. As shown in Table 1, the performance improves as we increase the amount of training data from 0 to 100 percent of the YouTubeVOS training split. Already at 25 percent our approach outperforms recent methods such as AGAME [2] (see Table 2 in the paper). At 50 percent, our approach surpasses all compared methods in Table 2 in the paper, that are trained only on the full YouTube-VOS training set. Remarkably, our target model without the segmentation (Ours D-only), consisting of a linear filter that requires no pretraining, obtains a $\mathcal{G}$-score superior to the methods OSVOS [1], OnAVOS [5] and the recent RVOS [4] (see Table 2 in the paper).

## 2.2. Algorithm runtime analysis

We investigate the runtime for the different steps in our proposed VOS approach in Algorithm 1 in the main paper. All runtimes have been computed by averaging over the DAVIS 2016 evaluation split.

Figure 2 shows how execution in frame 1 (the *init phase*, steps 1 and 2 in Algorithm 1) changes vs the size of the initial sample memory (or dataset) $\mathcal{M}_0$, when segmenting a single object. We present relative runtimes of the maximum time spent using all steps with $|\mathcal{M}_0| = 20$ The time spent during data augmentation is dominated by the inpainting which is performed only once, on the first frame, and hence it is appears constant.

Figure 3 shows how execution in frames 2 and onward (the *forward phase*, steps 4-9 in Algorithm 1) when fixating $|\mathcal{M}_0| = 20$ and varying the maximum dataset size $K_{\max}$. Again, we present timings in relation to the full execution time using $|\mathcal{M}| = 80$. Note that most DAVIS 2016 videos are only a few seconds long and will never fill $\mathcal{M}$ entirely. This will reduce the apparent runtimes for large dataset sizes.

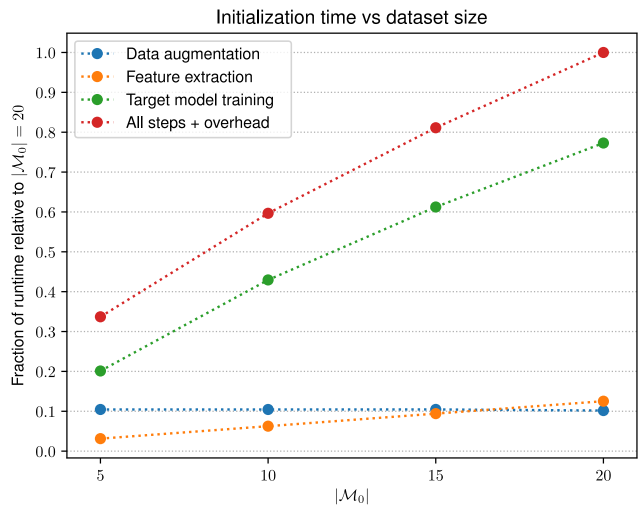

Figure 2. Initialization runtime (frame 1), relative to the initial dataset size $|\mathcal{M}_0| = 20$ with a fixed maximum dataset size $K_{\max} = 80$ (section 3.4).

In addition, Table 2 shows the distribution of average time spent on each step in one frame in the forward phase. This is in the steady-state situation, after the sample memory is filled (here $|\mathcal{M}_i| = 80$), averaged over the last $t_s = 8$ frames of the sequence.

Since the DAVIS videos are quite short, the init phase accounts for 41 percent of the total runtime when evaluating the **Ours** variant on DAVIS2016. On a per video basis, the initialization requires between 31 (for "cows" with 104 frames) and 60 percent (for "car-shadow" with 40 frames) of the total runtime.

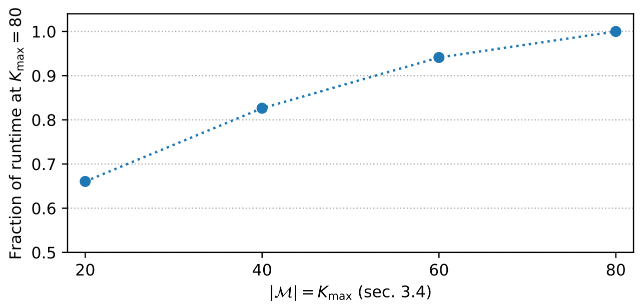

Figure 3. Per-frame (forward phase) runtime relative to the maximum dataset size $K_{\max} = 80$ with a fixed initial size $|\mathcal{M}_0| = 20$.

| Algorithm step | Percent |
|---|---|
| 4. Feature extraction | 24.0 |
| 6. Segmentation | 10.0 |
| 9. Target model update training | 65.5 |
| Other | 0.6 |

Table 2. Distribution of time spent on steps in the frame loop of algorithm 1. The target prediction (step 5) is wrapped into "Other".

From figure 2, we conclude that the first-frame initialization (algorithm steps 1-2) scales approximately linearly with $|\mathcal{M}_0|$. The per-frame (forward phase) processing (Algorithm 1 steps 3-9) is dominated by the model update training and feature extraction. Theoretically, the complexity of both phases scale linearly with the number of iterations in their respective optimization steps (step 2 and 9) as well as linearly with the number of targets.

## 2.3. Parameter sensitivity

Figure 4 reports the mean $\mathcal{J}$ as functions of the memory learning rate $\eta$ and target model update interval $t_s$ (defined in Section 3.4 in the paper). The experiments are performed on the YouTubeVOS validation split, defined in Section 4 in the paper. It is apparent that the method is rather insensitive to either parameter.

In addition, Table 3 shows the mean $\mathcal{J}$ as functions of the size of the initial training dataset $\mathcal{M}_0$. We test two variants of our method, one trained on YouTubeVOS data and one trained on both YouTubeVOS and DAVIS data. We evaluate on our own YouTubeVOS validation split and the DAVIS validation set. We observe that the YouTubeVOS evaluation is insensitive to the choice of $|\mathcal{M}_0|$. While still achieving a competitive $\mathcal{J}$-score without initial data augmentation, our approach obtains the best performance using four additional augmented samples in $\mathcal{M}_0$.

## References

[1] Sergi Caelles, K-K Maninis, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. One-shot video object segmentation. In *2017 IEEE Conference on Computer*

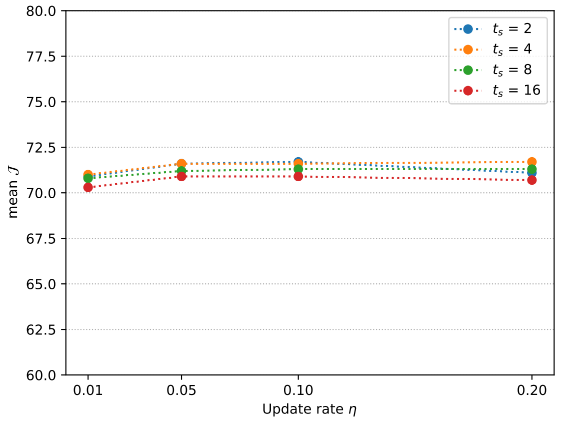

Figure 4. Mean intersection-over-union results on our YouTubeVOS validation split (defined in Section 4), as a function of the memory update rate $\eta$ and retraining interval $t_s$ hyper parameters, detailed in section 3.4.

| | | $\lvert\mathcal{M}_0\rvert$ | | | |
|---|---|---|---|---|---|
| **Method** | **Eval** | 1 | 5 | 10 | 20 |
| Ours (yt) | ytv | 71.5 | 71.4 | 71.2 | 71.4 |
| Ours (yt+dv) | dvv | 69.4 | 73.8 | 72.6 | 73.1 |

Table 3. The influence on mean $\mathcal{J}$ with varying $|\mathcal{M}_0|$ during inference. We test two variants, trained on either YouTubeVOS only (yt) or both YouTubeVOS and DAVIS2017 (yt+dv17). Results shown are from evaluating on our YoutubeVOS validation split (ytv) and the DAVS2017 validation split (dvv).

*Vision and Pattern Recognition (CVPR)*, pages 5320–5329. IEEE, 2017. 1

[2] Joakim Johnander, Martin Danelljan, Emil Brissman, Fahad Shahbaz Khan, and Michael Felsberg. A generative appearance model for end-to-end video object segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[3] Alexandru Telea. An image inpainting technique based on the fast marching method. *Journal of graphics tools*, 9(1):23–34, 2004. 1

[4] Carles Ventura, Miriam Bellver, Andreu Girbau, Amaia Salvador, Ferran Marques, and Xavier Giro-i Nieto. Rvos: End-to-end recurrent network for video object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5277–5286, 2019. 1

[5] Paul Voigtlaender and Bastian Leibe. Online adaptation of convolutional neural networks for video object segmentation. In *BMVC*, 2017. 1