Robust Visual Tracking by Segmentation Appendix

Matthieu Paul, Martin Danelljan, Christoph Mayer, and Luc Van Gool

Computer Vision Lab, ETH Zürich, Switzerland
{paulma, damartin, chmayer, vangool}@vision.ee.ethz.ch

In this Appendix, we provide further details on various aspects of our tracking pipeline. First, we provide additional architectural and inference details in Sections A and B. Second, we provide additional ablation studies, in particular on the loss weighting parameter η on different benchmarks to show the importance of the auxiliary instance localization loss in Section C. Then, we provide success plots for different Visual Object Tracking (VOT) benchmarks as well as a detailed analysis of our results on LaSOT [6] by comparing our approach against the other state-of-the-art methods for all the dataset attributes in Section D. Finally, we provide some additional visual comparison to other trackers in Section E.

A Additional Architecture details

Classification Scores Encoder H_θ First, we describe in Figure A1 the architecture of the Classification Scores Encoder H_θ. It takes as input the $H \times W$-dimensional scores predicted by the Instance Localization (*Classification*) branch and outputs a 16 channels deep representation of those scores. The score encoder consists of a convolutional layer followed by a max-pool layer with stride one and two residual blocks. The output of the residual blocks has 64 channels. Thus, the final convolutional layer reduces the number of channels of the output to 16 to match the encoded scores with the mask encoding. All the convolutional layers use (3×3) kernels with a stride of one to preserve the spatial size of the input classification scores.

Segmentation Decoder D_θ The segmentation decoder has the same structure has in LWL [2]. Together with the backbone, it shows a U-Net structure and mainly consists of four decoder blocks. It takes as input the extracted ResNet-50 backbone features and the combined encoding x_f from both the instance localization branch ($H_\theta(s_\mathrm{c})$) and the segmentation branch (x_m), with $x_\mathrm{f} = x_\mathrm{m} + H_\theta(s_\mathrm{c})$. Since the encoded instance localization scores have a lower spatial resolution than the mask encoding x_m, we upscale the encoded instance localization scores using a bilinear interpolation before adding it with the mask encoding x_m. We refer the reader to [2] for more details about the decoder structure.

Segmentation Branch We use the same architectures for the feature extractor F_θ, the label encoder E_θ, the weight predictor W_θ, the few-shot learner A_θ and the segmentation model T_τ as proposed in LWL [2]. Hence, we refer the reader to [2] for more details.

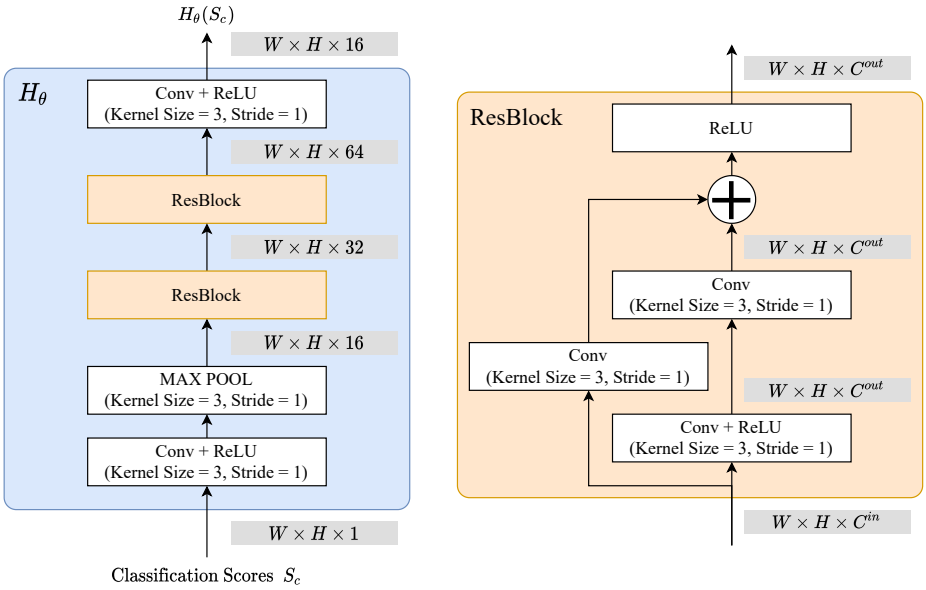

Fig. A1: Classification Scores Encoder H_θ.

Instance Localization Branch We use the same architectures for the feature extractor G_θ, the model predictor P_θ and the instance model T_κ as proposed in DiMP [1]. Hence, we refer the reader to [1] for more details.

B Additional Inference details

Search region selection The backbone does not extract features on the full image. Instead, we sample a smaller image patch for extraction, which is centered at the current target location and 6 times larger than the current estimated target size, when it does not exceed the size of the image. The estimation of the target state (position and size) is therefore crucial to ensure an optimal crop. In most situations, the segmentation output is used to determine the target state since it has a high accuracy. The *target center* is computed as the center of mass of the predicted per-pixel segmentation probability scores. The *target size* is computed as the variance of the segmentation probability scores.

If the segmentation branch cannot find the target (as described in the main paper), but the instance branch still outputs a high enough confidence score, we use it to update the target position. This is particularly important in sequences where the target is becoming too small for some time, but we can still track the target position.

When both branch cannot find the target, the internal state of the tracker is not updated. We upscale the search area based on the previous 60 valid predicted

Table A1: Ablation on the classification vs. segmentation loss weighting on different datasets in terms of AUC (area-under-the-curve) and AO (average overlap)

η	LaSOT [6] AUC	GOT-10k [9] AO	TrackingNet [13] AUC	NFS [8] AUC	UAV123 [12] AUC
0.0	67.7	84.0	81.2	63.7	64.7
0.4	69.8	84.0	81.4	66.2	67.4
10	69.7	85.2	81.6	65.4	67.6

scales. This is helpful in situations where the size of the object shrinks although its size does not change. This typically happens during occlusions, or if the target goes out of the frame partially or completely.

C Additional Ablations

In this section, we provide additional ablation studies related to our method, first on the weighting of the segmentation and classification losses used for training, second on the parameters that might make a difference specifically for Video Object Segmentation (VOS) benchmarks like Youtube-VOS [18].

Weighting segmentation and classification losses For this ablation, we study the weighting of the segmentation loss $\mathcal{L}_{\text{s}}$ and the instance localization loss $\mathcal{L}_{\text{c}}$ in the total loss $\mathcal{L}_{\text{tot}}$. It used to train our model and its influence on the overall performance during tracking. We recall that

$$\mathcal{L}_{\text{tot}} = \mathcal{L}_{\text{s}} + \eta \cdot \mathcal{L}_{\text{c}}. \tag{1}$$

Table A1 shows the results when training the tracker with three different values of η on five VOT datasets. First, we examine the case where we omit the auxiliary instance localization loss ($\eta = 0.0$). This means that the whole pipeline is trained for segmentation and the instance branch is not trained to produce specifically accurate localization scores. We observe that this setting leads to the lowest performance on all tested datasets, often by a large margin. Secondly, we test a dominant segmentation loss ($\eta = 0.4$), since the segmentation branch needs to be trained for a more complex task than the instance branch. We see a performance gain for almost all datasets. Thus, employing the auxiliary loss to train the instance localization branch helps to improve the tracking performance. We observe that using the auxiliary loss leads to localization scores generated during inference that are sharper, cleaner and localize the center of the target more accurately. Finally, we put an even higher weight on the classification term ($\eta = 10$). This setup leads to an even more accurate localization, and leads to the best results on average. Thus, we set $\eta = 10$ to train our tracking pipeline.

Fine-tuning on Youtube-VOS [6] In this section, we analyze whether we can gear our pipeline towards VOS benchmarks. To do that, we take our model and inference parameters, and modify them slightly. On the one hand, the model is fined-tuned for 50 epochs using Youtube-VOS [18] only for both training and

Table A2: Results on the Youtube-VOS 2019 [18] and DAVIS 2017 [15] datasets with a fined tuned model and inference parameters refered as *RTS (YT-FT)*.

	YouTube-VOS 2019 [18]					DAVIS 2017 [15]		
Method	$\mathcal{G}$	$\mathcal{J}_{\text{seen}}$	$\mathcal{J}_{\text{unseen}}$	$\mathcal{F}_{\text{seen}}$	$\mathcal{F}_{\text{unseen}}$	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$
RTS	79.7	77.9	75.4	82.0	83.3	80.2	77.9	82.6
RTS (YT-FT)	80.3	78.8	76.2	82.9	83.5	80.3	77.7	82.9
LWL [2]	81.0	79.6	76.4	83.8	84.2	81.6	79.1	84.1
STA [21]	80.6	-	-	-	-	-	-	-
STM [14]	79.2	79.6	73.0	83.6	80.6	81.8	79.2	84.3

validation. We also increase the initialization phase from 100 to 200 frames, and remove the relative target scale change limit from one frame to the next. In our standard model, we limit that scale change to 20% for increased robustness.

The results are presented in Table A2 for Youtube-VOS [18] and Davis [15]. We observe that the performances between both of our models stay very close for Davis, but that the fine-tuned model is getting closer to the baseline LWL [2] for Youtube-VOS. The more frequent updates seem to help, and not restricting the scale change of objects from frame to frames seems to play a role, since we get an improvement of 0.6 in $\mathcal{G}$ score.

D Additional Evaluation results

In this section we provide additional plots of our approach on different benchmarks, and a attribute analysis on LaSOT [6].

Success plots for LaSOT [6], NFS [8] and UAV123 [12] We provide in Figure A2 all the plots for the metrics we report for LaSOT [6] in the paper: *Success*, *Normalized Precision* and *Precision* plots. For completeness, we provide the success plots for NFS [8] and UAV123 [12] in Figure A3.

Attribute analysis on LaSOT [6] In this section, we focus on the dataset sequences attributes. We compare our approach to numerous other trackers, and provide the detailed results in Table A3. Furthermore, we highlight the strength of our approach in Figure A4 by focusing the comparison only to the two current state-of-the-art methods ToMP-101 and ToMP-50 [10].

There are 14 attributes provided for LaSOT [6] sequences, representing different kind of challenges the tracker has to deal with in different situations. Compared to existing trackers, our method achieves better AUC scores in 11 out of 14 attributes. In particular, we outperform ToMP-50 [10] and ToMP-101 [10] by a large margin for the following attributes: *Camera Motion* (+4.2% and +2.7%), *Scale Variation* (+1.8% and 0.9%), *Deformation* (+3.1% and +2.2%), *Motion Blur* (+3.1% and +2.5%) and *Aspect Ratio Change* (+1.7% and +1.0%). Our method is only outperformed on three attributes by KeepTrack [11] and ToMP [10] for *Fast Motion* (-2.3% to -4.1%) and for *Illumination Variation* (-0.3% to -1.4%). For *Background Clutter*, RTS outperforms ToMP-50 [10] by 2.3% and fall just behind ToMP-101 [10] (-0.1%).

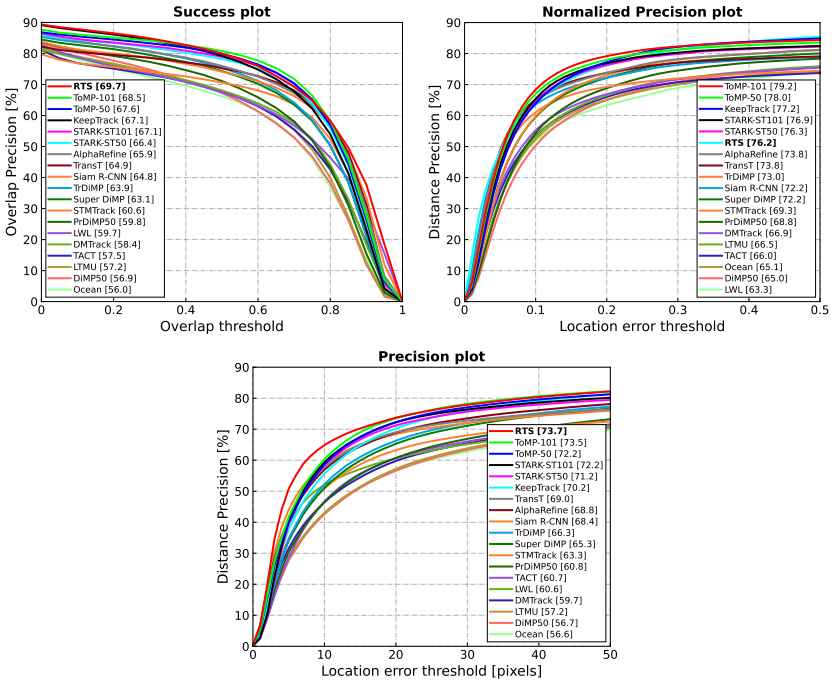

Fig. A2: Success, precision and normalized precision plots on LaSOT [6]. Our approach outperforms all other methods by a large margin in AUC, reported in the legend.

E Additional Content

Figure A5 shows additional visual results compared to other state-of-the-art trackers on 6 different sequences of LaSOT [6]. For more content, we refer the reader to: https://github.com/visionml/pytracking.

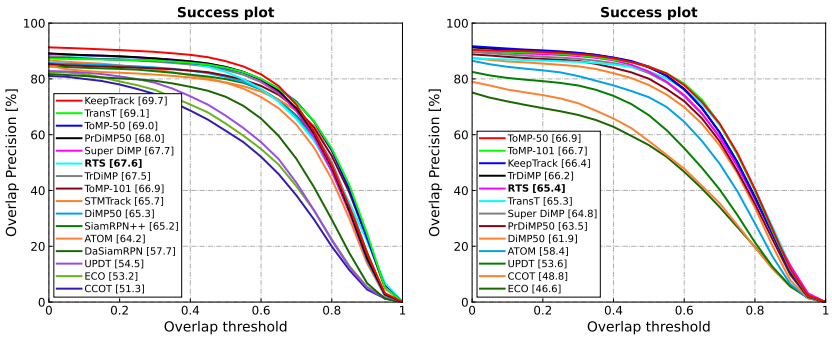

Fig. A3: Success plots on the UAV123 [12] (left) and NFS [8] (right) datasets in terms of overall AUC score, reported in the legend.

Table A3: LaSOT [6] attribute-based analysis. Each column corresponds to the results computed on all sequences in the dataset with the corresponding attribute. Our method outperforms all others in 12 out of 14 attributes.

	Illumination Variation	Partial Occlusion	Deformation	Motion Blur	Camera Motion	Rotation	Background Clutter	Viewpoint Change	Scale Variation	Full Occlusion	Fast Motion	Out-of-View	Low Resolution	Aspect Ratio Change	Total
LTMU [4]	56.5	54.0	57.2	55.8	61.6	55.1	49.9	56.7	57.1	49.9	44.0	52.7	51.4	55.1	57.2
LWL [2]	65.3	56.4	61.6	59.1	64.7	57.4	53.1	58.1	59.3	48.7	46.5	51.5	48.7	57.9	59.7
PrDiMP50 [5]	63.7	56.9	60.8	57.9	64.2	58.1	54.3	59.2	59.4	51.3	48.4	55.3	53.5	58.6	59.8
STMTrack [7]	65.2	57.1	64.0	55.3	63.3	60.1	54.1	58.2	60.6	47.8	42.4	51.9	50.3	58.8	60.6
SuperDiMP [1]	67.8	59.7	63.4	62.0	68.0	61.4	57.3	63.4	62.9	54.1	50.7	59.0	56.4	61.6	63.1
TrDiMP [17]	67.5	61.1	64.4	62.4	68.1	62.4	58.9	62.8	63.4	56.4	53.0	60.7	58.1	62.3	63.9
Siam R-CNN [16]	64.6	62.2	65.2	63.1	68.2	64.1	54.2	65.3	64.5	55.3	51.5	62.2	57.1	63.4	64.8
TransT [3]	65.2	62.0	67.0	63.0	67.2	64.3	57.9	61.7	64.6	55.3	51.0	58.2	56.4	63.2	64.9
AlphaRefine [19]	69.4	62.3	66.3	65.2	70.0	63.9	58.8	63.1	65.4	57.4	53.6	61.1	58.6	64.1	65.3
KeepTrack Fast [11]	**70.1**	63.8	66.2	65.0	70.7	65.1	60.1	67.6	66.6	59.2	57.1	63.4	62.0	65.6	66.8
KeepTrack [11]	**69.7**	64.1	67.0	**66.7**	71.0	65.3	61.2	66.9	66.8	**60.1**	**57.7**	**64.1**	62.0	65.9	67.1
STARK-ST101 [20]	67.5	65.1	68.3	64.5	69.5	66.6	57.4	68.8	66.8	58.9	54.2	63.3	59.6	65.6	67.1
ToMP-50 [10]	66.8	64.9	68.5	64.6	70.2	67.3	59.1	67.2	67.5	59.3	56.1	63.7	61.1	66.5	67.6
ToMP-101 [10]	69.0	**65.3**	**69.4**	65.2	**71.7**	**67.8**	**61.5**	**69.2**	**68.4**	59.1	**57.9**	**64.1**	**62.5**	**67.2**	**68.5**
RTS	68.7	**66.9**	**71.6**	**67.7**	**74.4**	**67.9**	**61.4**	**69.7**	**69.3**	**60.5**	53.8	**66.3**	**62.7**	**68.2**	**69.7**

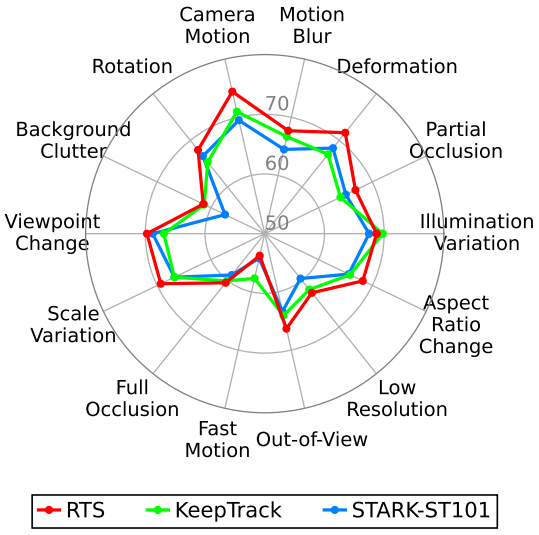

Fig. A4: Attributes comparison on LaSOT [6].

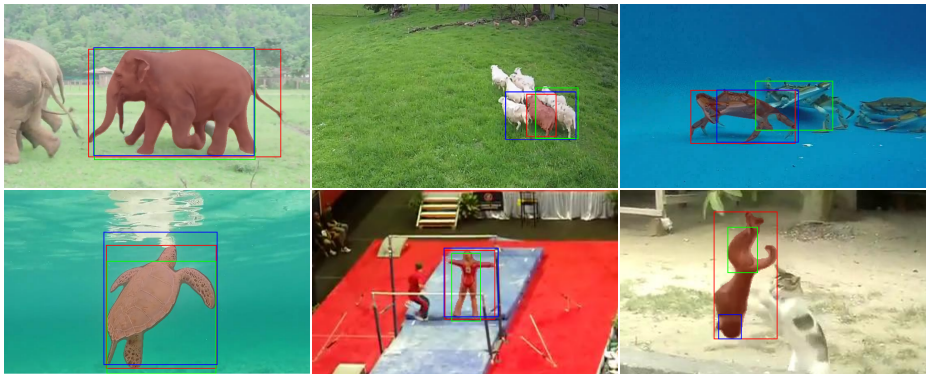

Fig. A5: Qualitative results on LaSOT [6] of our approach compared to the previous state-of-the-art methods KeepTrack [11] and STARK-ST101 [20]. As they do not produce segmentation masks, we represent ours as a red overlay and print for all methods the predicted bounding boxes with the following color code:
KeepTrack STARK-ST101 RTS

References

1. Bhat, G., Danelljan, M., Gool, L.V., Timofte, R.: Learning discriminative model prediction for tracking. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (October 2019) 2, 6
2. Bhat, G., Lawin, F.J., Danelljan, M., Robinson, A., Felsberg, M., Gool, L.V., Timofte, R.: Learning what to learn for video object segmentation. In: European Conference on Computer Vision ECCV (2020) 1, 4, 6
3. Chen, X., Yan, B., Zhu, J., Wang, D., Yang, X., Lu, H.: Transformer tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2021) 6
4. Dai, K., Zhang, Y., Wang, D., Li, J., Lu, H., Yang, X.: High-performance long-term tracking with meta-updater. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020) 6
5. Danelljan, M., Gool, L.V., Timofte, R.: Probabilistic regression for visual tracking. In: CVPR (2020) 6
6. Fan, H., Lin, L., Yang, F., Chu, P., Deng, G., Yu, S., Bai, H., Xu, Y., Liao, C., Ling, H.: Lasot: A high-quality benchmark for large-scale single object tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019) 1, 3, 4, 5, 6, 7
7. Fu, Z., Liu, Q., Fu, Z., Wang, Y.: Stmtrack: Template-free visual tracking with space-time memory networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2021) 6
8. Galoogahi, H.K., Fagg, A., Huang, C., Ramanan, D., Lucey, S.: Need for speed: A benchmark for higher frame rate object tracking. In: ICCV (2017) 3, 4, 6
9. Huang, L., Zhao, X., Huang, K.: Got-10k: A large high-diversity benchmark for generic object tracking in the wild. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **43**(5), 1562–1577 (2021) 3
10. Mayer, C., Danelljan, M., Bhat, G., Paul, M., Paudel, D.P., Yu, F., Van Gool, L.: Transforming model prediction for tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8731–8740 (June 2022) 4, 6
11. Mayer, C., Danelljan, M., Paudel, D.P., Van Gool, L.: Learning target candidate association to keep track of what not to track. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 13444–13454 (October 2021) 4, 6, 7
12. Mueller, M., Smith, N., Ghanem, B.: A benchmark and simulator for uav tracking. In: Proceedings of the European Conference on Computer Vision (ECCV) (October 2016) 3, 4, 6
13. Müller, M., Bibi, A., Giancola, S., Al-Subaihi, S., Ghanem, B.: Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In: ECCV (2018) 3
14. Oh, S.W., Lee, J.Y., Xu, N., Kim, S.J.: Video object segmentation using space-time memory networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (October 2019) 4
15. Pont-Tuset, J., Perazzi, F., Caelles, S., Arbeláez, P., Sorkine-Hornung, A., Van Gool, L.: The 2017 davis challenge on video object segmentation. arXiv:1704.00675 (2017) 4
16. Voigtlaender, P., Luiten, J., Torr, P.H., Leibe, B.: Siam R-CNN: Visual tracking by re-detection. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020) 6

17. Wang, N., Zhou, W., Wang, J., Li, H.: Transformer meets tracker: Exploiting temporal context for robust visual tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2021) 6
18. Xu, N., Yang, L., Fan, Y., Yue, D., Liang, Y., Yang, J., Huang, T.: Youtube-vos: A large-scale video object segmentation benchmark (2018) 3, 4
19. Yan, B., Wang, D., Lu, H., Yang, X.: Alpha-refine: Boosting tracking performance by precise bounding box estimation. In: CVPR (2021) 6
20. Yan, B., Peng, H., Fu, J., Wang, D., Lu, H.: Learning spatio-temporal transformer for visual tracking. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 10448–10457 (October 2021) 6, 7
21. Zhao, B., Bhat, G., Danelljan, M., Van Gool, L., Timofte, R.: Generating masks from boxes by mining spatio-temporal consistencies in videos. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 13556–13566 (October 2021) 4