In this document, we present more experimental results and analyses to complement the manuscript:

• Running speed analysis in Section 1.
• More comparisons. We compare the tracking accuracy vs. speed performance on both the OTB-2013 [23] and OTB-2015 [24] datasets with more trackers in Section 2.
• More ablation tests. We present the experiments by directly fine-tuning a deep network with the initial information in Section 3.

1. Running Speed Analysis

The initialization time of the proposed model is about 2 seconds on the machine specified on L575. The final average running speed is related to the length of test sequences as shown in the figure below. The average running speed on the OTB-2015 dataset is 33.7 FPS.

2. Tracking Accuracy vs. Speed

Figure 1 and Figure 2 show the tracking accuracy vs. speed performance on the OTB-2013 and OTB-2015 datasets, respectively. We add more trackers (annotated with gray square) using the reported tracking speed as their source code is not available. While the comparisons based on the reported results without using the source code on the same machine do not truly reveal the run time performance, we add these results for completeness.

The trackers with the reported speed include DRT [19], LSART [18], FlowT [28], SACF [25], SA-Siam [9], RASNet [22], SiamRPN [11], RT-MDNet RT-MDNet, and CFnet2-tri [7]. The other trackers including CCOT [6], VITAL [17], DAT [15], MetaSDNet [14], ECO [5], STR-CF [12], DSLT [13], MCFC [26], CREST [16], MCC-T-H [21], ACT [3], DSiamM [8], ECO-HC [5], BACF [10], CFNet [20], SiamFC [2], Staple [1], Trace [4], and DaSi-amRPN [27] are evaluated on the same machine.

Compared with all these state-of-the-art trackers, the proposed algorithm achieves a favorable performance in terms of speed and accuracy. The proposed tracker achieves the best performance among the real-time trackers, which can be attributed to the proposed target-active and scale-sensitive features. The proposed features are generated by the filters which are active to specific patterns of the target and sensitive to scale changes. As such, the target-specific and scale-sensitive features distinguish the target from the background well and are sensitive to scale changes. On the other hand, the proposed algorithm uses a simple tracking framework directly comparing the features of the target template and the search region without a complicated inference model or an online update model. Therefore, the proposed tracker achieve fast performance and prevents over-fitting and model drifting.

3. More Ablation Tests

According to suggestions of reviewers, we compare the proposed algorithm with the model which fine-tunes a deep feature network directly. We conduct the experiments by directly fine-tuning a deep network with the initial information and concatenate 3 or 4 CNN layers as features. Table 1 shows that directly fine-tuning performs poorly and concatenating more layers does not improve tracking performance. Concatenating low-level layers does not always improve tracking performance as it makes the network model complicated. In addition, using more layer features does not solve the uncertainty problem because uncertainty comes
Figure 1. Tracking accuracy vs. speed on the OTB2013 dataset. The proposed algorithm achieves a favorable performance against the state-of-the-art trackers.

Figure 2. Tracking accuracy vs. speed on the OTB2015 dataset. The proposed algorithm achieves a favorable performance against the state-of-the-art trackers.
from an arbitrary target in the tracking task, which is unknown before tracking.

Table 1. **Comparison with the baseline by directly fine-tuning and concatenating more features.**

<table>
<thead>
<tr>
<th>Tracker</th>
<th>OTB-2013 Precision</th>
<th>OTB-2013 AUC</th>
<th>OTB-2015 Precision</th>
<th>OTB-2015 AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finetuning</td>
<td>0.848</td>
<td>0.653</td>
<td>0.826</td>
<td>0.639</td>
</tr>
<tr>
<td>Three layers</td>
<td>0.850</td>
<td>0.656</td>
<td>0.827</td>
<td>0.642</td>
</tr>
<tr>
<td>Four layers</td>
<td>0.817</td>
<td>0.635</td>
<td>0.817</td>
<td>0.633</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>0.896</strong></td>
<td><strong>0.680</strong></td>
<td><strong>0.866</strong></td>
<td><strong>0.660</strong></td>
</tr>
</tbody>
</table>

Our method significantly differs from the approach by directly fine-tuning a deep network with the initial information in two aspects. First, we do not modify or change the activations of existing pre-trained deep models. Instead, we only select the target-active and the scale-sensitive filters (channels). Second, the original deep models designed for image classification is not effective for tracking due to the use of spatial pooling, which decreases the valuable spatial information that helps to precisely localize target objects.

Note that fine-tuning a deep network with the initial frame would easily result in over-fitting due to limited samples in one image. In contrast, we aim to identify the target-aware features. We use the given target information to compute gradients of the proposed losses and identify the target-specific channels according to the gradients, rather than using the given target information to fine-tune a network directly. In addition, a classification network usually needs a large feature space to separate all possible (hundreds to thousands) classes, which is redundant for tracking as only two classes (foreground and background) are essential for a test sequence. As such, we propose to exploit features specific to the target of interest in a test sequence.
References


