Model Decay in Long-Term Tracking

Gavves, E.; Tao, R.; Gupta, D.K.; Smeulders, A.W.M.

Link to publication

Model Decay in Long-Term Tracking

Efstratios Gavves
QUVA Lab, University of Amsterdam
e.gavves@uva.nl

Ran Tao[1]
Kepler Vision Technologies
r.tao@keplervision.eu

Deepak K. Gupta
QUVA Lab, University of Amsterdam
d.k.gupta@uva.nl

Arnold W. M. Smeulders
QUVA Lab, University of Amsterdam
a.w.m.smeulders@uva.nl

Abstract

Updating the tracker model with adverse bounding box predictions adds an unavoidable bias term to the learning. This bias term, which we refer to as model decay, offsets the learning and causes tracking drift. While its adverse affect might not be visible in short-term tracking, accumulation of this bias over a long-term can eventually lead to a permanent loss of the target. In this paper, we look at the problem of model bias from a mathematical perspective. Further, we briefly examine the effect of various sources of tracking error on model decay, using a correlation filter (ECO) and a Siamese (SINT) tracker. Based on observations and insights, we propose simple additions that help to reduce model decay in long-term tracking. The proposed tracker is evaluated on four long-term and one short term tracking benchmarks, demonstrating superior accuracy and robustness, even in 30 minute long videos.

1. Introduction

Fueled by the availability of standard datasets [17, 36, 30], tracking has made great progress over the last few years. These datasets have mainly been designed to tackle challenges encountered in short-term tracking. For example, the average lengths of videos in ALOV [30] and OTB [36] are only about 10 seconds and 20 seconds, respectively. The rationale behind designing these datasets was to select hard moments such as changes in illumination conditions, abrupt motion, clutter, large deformations, sudden occlusions, among others.

In general, performing well on the above datasets has been interpreted as overcoming the major challenges of tracking. However, in practice additional and different problems occur when the duration of tracking is longer, *e.g.*
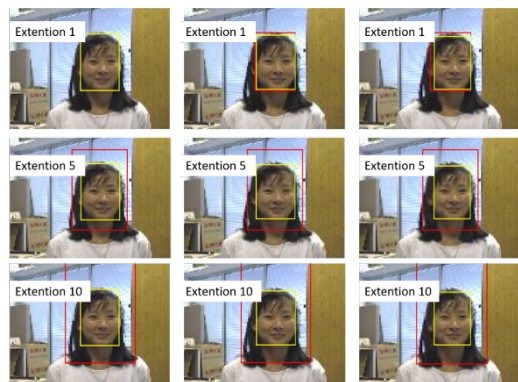

Figure 1: Predictions from ECO [6] on an artificially extended video created from OTB50 data (red box: tracker prediction, yellow box: ground truth prediction). *Model decay* is prevalent here although the appearance variation remains intact. Due to heavy updating involved, model decay is noticeable from very early stages itself, even for clearly visible target objects moving slowly .

half an hour. When considering the practical applications of tracking, long durations are much more frequent than short videos, as in human interactions, sports, ego-documents and TV shows.

To account for tracking challenges and appearance changes in short and long videos, state-of-art trackers must update their internal model frequently. However, too many model updates can eventually decay the inherent tracker model and the target might get lost. While the decay might not be noticeable in short-term videos, the accumulated effect is very prominent in long-term cases. In this paper, we study model decay from a theoretical perspective with the goal of identifying underline reasons that cause it, and explore possible remedies to circumvent this problem.

To demonstrate the severe effects of model decay on long videos, results from a simple numerical experiment are presented here. A sample

---

[1]During this research, R. Tao was affiliated with QUVA Lab at University of Amsterdam.

video is chosen randomly from the OTB50 dataset, and extended artificially in the following manner: $[x_1, x_2, ..., x_{T-1}, x_T, x_T, x_{T-1}, ..., x_2, x_1, x_1, x_2, ..., x_T, ...]$.. This way of extending ensures that any differences in tracking accuracy observed in the later parts of the long sequence will be caused solely due to increased video length.

Fig. 1 shows the predictions obtained using ECO, one among the state-of-art trackers, for 3 different frames observed in 3 different repetitions of the original video. The 3 frames per repetition are chosen such that they lie close in the sequence and the appearance of the target does not change much among them. It can be seen that the tracker predictions become less and less accurate over time due to model decay caused by the gradual but erroneous, heavy updating. The tracker drift caused by model decay has been a long known phenomenon, but, in the context of short-term tracking, this issue has not been very relevant. To develop circumventing measures against it for long-term tracking, a theoretical understanding is needed that can provide a mathematical definition to the underlying mechanism.

In this paper, we mathematically study the learning dynamics of general tracker models. We show that in the presence of prediction errors, any model update adds always an unavoidable bias to the learning. Second, based on numerical experiments, the correlation between model decay and various tracking challenges is studied on two different trackers: a correlation filter tracker and a siamese tracker. Finally, based on the identified limitations of the trackers and gained insights, some simple modifications are proposed that allow a siamese tracker to deal better with model decay, thereby making it amenable to long-term tracking.

2. Model Decay in Tracking

The most popular strategy for learning object trackers can be summarized in the following equations:

$$\phi_{t+1} = \arg\min_{\phi} \mathcal{L}(x_{1:t}, y_{1:t}) \tag{1}$$

$$y_{t+1} = f(x_{t+1}; \phi_{t+1}), \tag{2}$$

where f is the ϕ-parameterized tracker model that minimizes the tracker loss $\mathcal{L}$ over the dataset $D = [x_{1:t}, y_{1:t}]$ at the timestep $t + 1$. The dataset is composed of frames $x_{1:t} = [x_1, ..., x_t]$, and the tracker model f returns as output the bounding box predictions $y_{1:t} = [y_1, ..., y_t]$. To reduce notation clutter, we use $f_{i,t}$ to refer to the output of the tracker model with parameters ϕ_t applied on frame x_i. The model parameters are updated by taking small steps towards the gradient direction of the loss surface, namely using gradient descent approach (or its variants),

$$\frac{\partial \phi}{\partial t} = -\eta \nabla_\phi \mathcal{L}_t \Rightarrow \tag{3}$$

$$\phi_{t+1} = \phi_t - \eta \nabla_\phi \mathcal{L}_t \tag{4}$$

Central to the tracking learning problem, therefore, is the gradient of the tracking loss with respect to the model parameters. Extending on $\nabla_\phi \mathcal{L}$ and using the expectation over t timesteps, $\mathbb{E}[\cdot] = \frac{1}{t} \sum_{i=1}^{t} [\cdot]$, we have

$$\nabla_\phi \mathcal{L}_t = \nabla_\phi \mathbb{E}[(y_i - f_{i,t})^2] \tag{5}$$

$$= 2\mathbb{E}[f_t \nabla_\phi f_t] - 2\mathbb{E}[y_t \nabla_\phi f_t], . \tag{6}$$

To go from Eq. (5) to (6) we rely on that the bounding box coordinates y_i, predicted in previous frames, become dataset input variables with constant values. Thus, they are independent of ϕ and $\mathbb{E}[\nabla_\phi y_i^2] = 0$. By substituting Eq. (6) in (4), the model parameters update can be described as

$$\phi_{t+1} - \phi_t = -2\eta \Big[\mathbb{E}[f_{i,t} \nabla_\phi f_{i,t}] - \mathbb{E}[y_i \nabla_\phi f_{i,t}] \Big] \tag{7}$$

An interesting –but often overlooked reality– is that while tracking is casted as a supervised learning problem, there is only one data sample in the learning dataset that is definitely correct. This one and only correct sample is the pair (x_1, y_1^*) defined by the user in the first frame, where y_1^* represents the coordinates of the user specified bounding box describing the object. While all other bounding boxes $y_i \; \forall \; i > 1$ are used for re-training and fine-tuning the tracker, there is no guarantee that the bounding boxes y_i are indeed correct or even good enough for learning. In fact, had the predictions y_i been good enough for re-training the tracker, the tracker would not need to be retrained.

Based on the argument presented above, it is reasonable to expect that the predictions, which also serve as future training samples for the re-training of the tracker, are noisy measurements of the true bounding box coordinates y_i^*. Assuming Gaussian noise with variance σ_i^2, we can state:

$$y_i = y_i^* + \delta_i, \;\; \text{and} \;\; \delta_i \sim N(0, \sigma_i^2), \tag{8}$$

With the substitution of Eq. (8) in (7) and some rearrangement of terms, we have

$$\phi_{t+1} - \phi_t = -2\eta \Big[\mathbb{E}[f_{i,t} \nabla_\phi f_{i,t}] - \mathbb{E}[(y_i^* + \delta_i) \nabla_\phi f_{i,t}] \Big] \tag{9}$$

$$= \underbrace{-2\eta \mathbb{E}[(f_{i,t} - y_i^*) \cdot \nabla_\phi f_{i,t}]}_{\text{Perfect parameter update}} + \underbrace{2\eta \mathbb{E}[\delta_i \cdot \nabla_\phi f_{i,t}]}_{\text{Parameter bias}} \tag{10}$$

It is easy to recognize the two components in the parameter update of the tracker. The first term in Eq. (10) corresponds to the perfect model update component, as it corrects the error made by the model prediction $f_{i,t}$ as compared to the perfect box y_i^*. The second term corresponds to the parameter bias component, as this term depends directly on the error made by past predictions δ_i. If $\delta_i = 0$, then there would be no error, and the parameter updates would also be perfect.

Model dynamics. Having computed the effect of the past errors on the parameter updates of the tracker, we can next examine the effect on the model dynamics $\frac{\partial f}{\partial t}$ over time. Specifically, after updating the parameters, the relation between the past $f_{i,t}$ and the next model $f_{i,t+1}$ is

$$\frac{\partial f}{\partial t} \propto f_{i,t+1} - f_{i,t} = \frac{\partial f}{\partial \phi}\frac{\partial \phi}{\partial t} \Rightarrow \qquad (11)$$

$$f_{i,t+1} = f_{i,t} + \frac{\partial f}{\partial \phi}\frac{\partial \phi}{\partial t} \qquad (12)$$

As $\frac{\partial \phi}{\partial t} \propto \phi_{t+1} - \phi_t$, combining eq. (12) and (10) we have that

$$f_{i,t+1} = f_{i,t} \underbrace{-2\eta\mathbb{E}[(f_{i,t} - y_i^*) \cdot \|\nabla_\phi f_{i,t}\|^2]}_{\text{Perfect model update}} \qquad (13)$$

$$\underbrace{+2\eta\mathbb{E}[\delta_{i,t} \cdot \|\nabla_\phi f_{i,t}\|^2]}_{\text{Model decay}}$$

From Eq. (13), we make the following observation. Due to the continuous updates, the tracker model offshoots its predictions by a quantity that is linearly proportional to past errors. We refer to this quantity as model decay.

Long-term tracking & model decay. As the model dynamics in eq. (13) are recursive, it is implied that the bias term accumulates and, in fact, worsens over time. For as long as the cumulative model decay is small enough, usually in the early iterations, the model dynamics is sufficiently accurate. The early errors δ_i, are small not only because the tracker is still accurate, but also because the number of summands t is small. This is the reason why model decay is not a problem, and often goes even unnoticed, in short videos up to 10-20 sec. However, in longer videos where the number of summand t grows and the $\delta_{i,t}$ errors grow as well, the cumulative model decay become noticeable. The tracking paradox is more and more relevant when considering longer videos.

Dynamics of correlation filter and siamese trackers. Based on the model dynamics described, we further analyze two trackers chosen from correlation filter (ECO [6]) and siamese (SINT [32]) categories.

Regarding the initial model, correlation filter trackers rely on training using the first frame sample (and variants) for training. They also perform many model updates, even as frequently as every frame. These trackers are known to perform well on short-term videos. However, in longer videos the cumulative model bias should be expected to grow very large, and this will have a significant impact on the model dynamics, thereby causing tracker drift.

In contrast, siamese trackers rely for their initial model on a similarity function trained externally on large datasets

of matching versus non-matching visual pairs. Since siamese trackers can perform well even with no updates [32] their cumulative model decay is 0. The reason is that although prediction errors δ_i occur, the model us never updated, so $\nabla_\phi f_{i,t} = 0$. This means, however, that the tracker relies for its total model dynamics on the initial model only. All in all, in long videos siamese trackers are expected to show little or no model decay. However, their performance is sensitive in short videos and is only as good as their initial matching model, as empirically observed in [33].

An applied perspective. The consequence of the theoretical framework provided above is that now the aptitude of updating can be measured, and the presented mathematical study lays the ground work for designing remedies to reduce the model bias. While a clear solution to avoid model decay is not yet known, the obtained insights provide us a simple yet effective approach to reduce model bias to a certain extent.

From Eq. 10, it can be seen that an accurate estimation of model bias would require computing the term $\mathbb{E}[\delta_i\nabla_\phi f_{i,t}]$. However, the state parameters ϕ_{t+1} are not known beforehand, and redetecting the target in the earlier frames for every model update step (to obtain $f_{i,t}$) will be computationally very expensive. While a feasible approximation to the bias term in Eq. 10 is left for future research, we use a very simplified version of the bias term (by controlling δ). To demonstrate the effectiveness of the process, we present LT-SINT (a long-term variant of the siamese tracker SINT). Note that although SINT has been chosen to demonstrate the concept, the underlying idea is general, and can be combined with any existing tracker without much modification.

Under the scenarios of abrupt motion, SINT can lose the target and the false positives encountered thereafter increase δ quickly, adding remarkable residual to the model bias. To reduce this, LT-SINT employs a heuristic scheme where global search is performed every T frames. This reduces the chances of a target being lost for more than T frames at once. Compared to the local search, the global search results are more reliable and tracker model can be updated at these instances. However, occasional large appearance variations of the target coinciding with the global search step can lead to an even severe false positive getting identified during this step. This is definitely not desired since using this prediction to update the model would lead to large values of δ as well as adversely affect all terms defined by $f_{i,t+1}$. To circumvent this issue, a motion model (referred to as *decay recognition network*) is introduced, which ensures that unreliable estimates during global search are prevented from contributing towards model updates. More details related to LTSINT are presented in Section 4.2.

3. Tracking Challenges & Model Decay

3.1. Long OTB

In [35, 30] the different tracking challenges have been systematically studied. As we would like to study the contribution of these sources of errors in model decay, we extend the videos from OTB-50 [35] dataset as follows. For a video with frames $x = [x_1, ..., x_T]$ we artificially increase its length by $x = [x_1, ..., x_T, x_T - 1, ..., x_1, x_2, ...]$, where a forward and reverse pass, $x_1, ..., x_T, x_{T-1}, ..., x_1$ counts as a single repetition. As the metadata annotations for the different tracking challenges are video-wide, the same annotations apply also for the extended videos. We refer to this dataset as the *Long OTB* and we propose to use it to monitor model decay in visual object trackers.

3.2. Effect of Tracking Challenges on Model Decay

Next, how model decay is influenced by the various tracking challenges documented in tracking surveys [35, 30]. Specifically, we focus on the tracking challenges reported in OTB-50 [35]: illumination variation, scale variation, occlusion, deformation, motion blur, fast motion, in-plane rotation, out-of-plane rotation, out-of-view, background clutters, low resolution.

In long videos, some of these tracking challenges may become extreme, leading to stronger pressure towards model decay. We identify three additional, extreme challenges: sampling drift, target disappearance and reappearance, and pixel-level errors. Sampling drift relates to the fast motion and scale variation challenges. It is caused by imperfect local scale-space search [11, 12, 5, 4, 32, 2, 27], often due to the target exceeding a user-defined maximum speed. Target disappearance and reappearance is an extreme version of the out-of-view and occlusion challenges, where the target object leaves the frame completely for an unknown length of time. As a consequence, there is no guarantee for the location of the re-entrance, compromising the assumption of motion continuity. Pixel-level errors refer to the label errors. They happen especially during generation of the "pseudo-positives" based on *ad hoc* rules like maximum overlap over time, which can be satisfied even with wrong detections. Although a < 0.7 IoU threshold is good enough for evaluating tracker accuracy, the added background pixels start counting as positive appearances, compounding slowly but consistently and leading to serious model decay. Pixel-level errors is a challenge relates to learning and optimization of the tracker and connects to all tracking challenges.

We plot the average AUC curves for the eleven annotated tracking challenges in Fig. 2 (up) for ECO [6] and Fig. 2 (bottom) for SINT. Although we have no annotations for sampling drift, target disappearance and re-appearance, and pixel-level errors, they connect to the eleven annotated
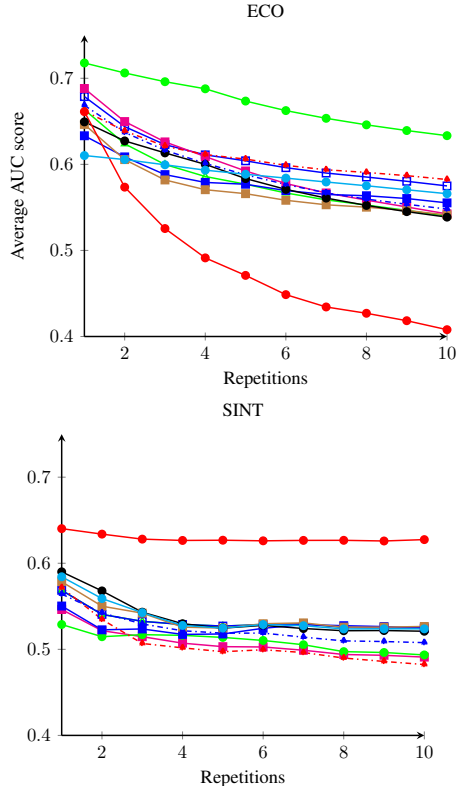

Figure 2: The effect of the 11 sources of tracking difficulty in Long OTB-50 videos. The legend goes as follows: illumination variation-IV (—▲—), scale variation-SV (—☐—), occlusion-OCC (—■—), deformation-DEF (—●—), motion blur-MB (—■—), fast motion-FM (—■—), in-plane rotation-IPR (—●—), out-of-plane rotation-OPR (·-▲-·), out-of-view-OV (—●—), background clutter-BC (·-▲-·), low resolution-LR (—●—).

tracking challenges and we can thus derive insights indirectly.

There are several interesting observations. First, ECO – relying on heavy model updating– pays a particularly high price when the target is occluded ($\sim 10\%$ drop) or goes out-of-view ($\sim 25\%$ drop) throughout the video. The other tracking challenges also contribute to model decay in the order of $\sim 3 - 8\%$. The only tracking challenges that appears to not contribute much to model decay is low resolution, probably because of the multi-scale search of ECO.

In contrast, SINT seems to be less affected by model decay. Since there is no model update mechanism in SINT, one would expect observing no model bias with it. However, due to the inherent local search strategy SINT can occasionally lose the target and this extent of losing can var across repetitions. Due to this reason, a slight drop in AUC score can be observed over several repetitions. The biggest

problem appears to be in-plane and out-of-plane rotation[1]. ($\sim 5\%$ drop) and background clutter ($\sim 7\%$ drop), while for the rest of the challenges the drop is in the order of $2 - 5\%$. That said, it is also clear that SINT scores considerably lower that that of ECO.

While the siamese tracker seems to perform well, it is also on a disadvantage as the first frame cannot alone describe the full appearance of the target object. Since model decay impacts long-term tracking significantly, siamese trackers with careful model updating is a reasonable forward direction. Last, we emphasize this analysis contemplates the best case scenario of having the same video frames repeated again and again. In reality, long videos change their appearance significantly and the above tracking challenges are expected to be even harsher.

4. Minimizing Model Decay, Simply

4.1. Predicting Model Decay

While eq. (13) explains the effect of the model decay on the dynamics of the model, it is only a theoretical tool. The reason is that it requires the coordinates of the optimal bounding box around the target, which is what we are looking for in the first place. by re-arranging the terms of eq. (13) and setting $\omega_t = (1 - \frac{\delta_{i,t}}{f_{i,t} - y_i^*})$, however, we have that

$$f_{i,t+1} = f_{i,t} - 2\eta \cdot \mathbb{E}[\omega_t \cdot (f_{i,t} - y_i^*)\|\nabla_\phi f_{i,t}\|^2] \quad (14)$$

One way to interpret eq. (14) is that $\omega_t \in \{0,1\}$ is a binary weight variable, which allows for model updates for only a fraction of the frames when $\omega_t = 1$. A logical step then is to measure ω_t by an independent model, *e.g.*, a second neural network, trained to recognize if the next frame is resembles a perfect or biased model update. We will refer to this as the *decay recognition network*.

To make sure any bias in $\nabla_\phi f_{i,t}$ has a minimal effect on ω_t, the decay recognition network must share as little correlation as possible with the tracker model. This means the decay recognition network has as little correlation as possible to the tracker model should not share any parameters with the tracker model. Also, it should not receive overly correlated input patches.

The latter is clearly a hard requirement, as by definition tracker predictions usually relate to each other in nearby frames. A solution is to rely on siamese trackers [32], whose tracking predictions are independent over time and suffer less from model decay already. Although the focus of this work is not to present a fully-fledged tracker that solves model decay, we, therefore, present two simple modifications on SINT [32].

---

[1]Without any provisions standard convolutional neural networks are not rotation invariant, so in-plane rotations generate different features

4.2. A Simple Long-term Tracker

SINT [32] performs no model updates and local search. So, the tracker follows the target only for as long as its appearance does not deviate too much from the first frame and the local search space contains the target. To account for significant appearance changes and sampling drift, two improvements are applied over SINT to design LTSINT.

First, we introduce a decay recognition network (DRN) that regulates whether the next model update will contribute bias or not, similar to self-awareness [21]. With the DRN module, we expect to obtain a qualitative approximation of ω_t. While there is no fixed choice for it, our DRN conditions on the similarity map in the current frame as well as the previous $K - 1$ frames. An LSTM-based binary classifier is used, and similarity maps from the history are incorporated to capture the temporal dynamics of the similarity distributions. Since the DRN operates on the predictions from the previous frames, the risk of being getting infected by model bias still pertains. To make sure no such additional bias is introduced, we simply train the LSTM using similarity feature maps from the original siamese similarity function only. Specifically, training is performed using tracks z from the standard siamese tracker [32] on a held out set deemed positive (or negative) when having an > 0.5 (or < 0.5) IoU threshold. At test time, when the LSTM scores above a highly conservative threshold, the siamese similarity function is updated.

The second change is to perform hybrid search to locate the most likely next position of the target. This involves performing local search, as in [32], as well as searching globally after every T frames. The addition of global search reduces the extent to which a target can be lost in one instance. Moreover, since it is performed only sparsely, the increase in computational time is limited. Global search involves a hierarchical search strategy of three levels. First, the $N(= 10)$ mos probable locations are identified. Around each of the N locations, search is then performed at M different scales during the second stage. For better localization in spatial and scale spaces, the third stage involves analysis over L finer scales around the $M \cdot N$ selected locations. The candidate with the highest value on the similarity map is then chosen. Local search is similar to the third stage of global search, and target is searched around the prediction obtained in the previous frame.

All said, we emphasize that no matter how careful we are with our model updates, the risk of model decay still exists. In practice, with the suggested modifications we did not witness significant model decay for videos even up to 30 minutes long.

Model Decay on LT-SINT We run ECO, SINT, LT-SINT on eight Long OTB videos, four videos with significant
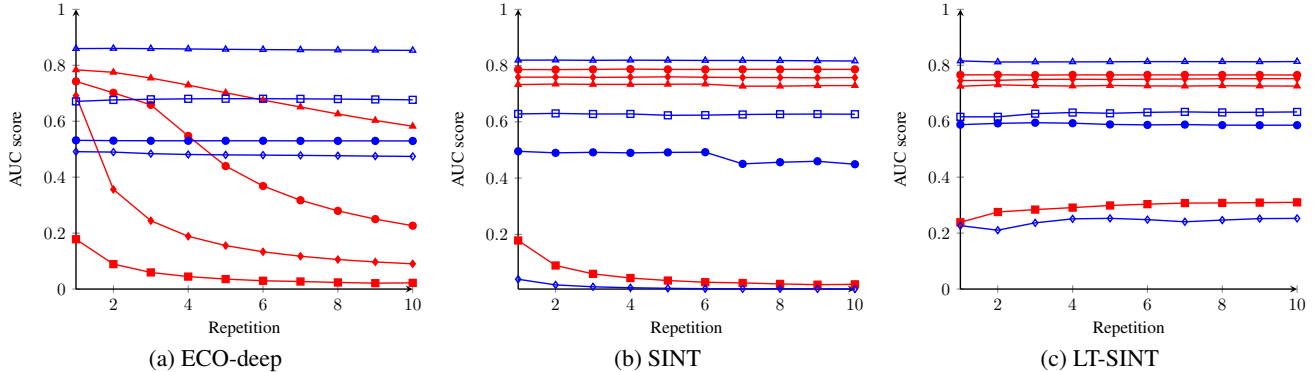
Figure 3: Running ECO, SINT, and LT-SINT on eight Long OTB videos, four videos with significant model decay (——) and four videos with small model decay (——), for ECO and SINT. The videos are: Girl: SV, OCC, IPR, OPR (—▲—), Freeman4: SV, OCC, IPR, OPR (—■—), Lemming: IV, SV, OCC, FM, OPR, OV (—◆—), SUV: OCC, IPR, OV (—●—), Car4: IV, SV (—△—), Couple: SV, DEF, FM, OPR, BC (—□—), Skiing: IV, SV, DEF, IPR, OPR (—◇—), Freeman1: SV, IPR, OPR (—●—).

model decay with red and four videos with small model decay with blue, and show results in Fig. 3. We observe that LT-SINT maintains top accuracy and is impervious to model decay, maintaining tracking throughout the whole duration of the video. In fact for some videos where ECO takes a toll due to model decay, like Freeman4, LT-SINT even improves. As LT-SINT appears to cope better with model decay, we focus on LT-SINT for the final benchmarking.

5. Related Work

Long-term tracking. To our knowledge, only limited works exist that focus on long-term tracking. A seminal work on long-term tracking is the TLD [14] tracker. It is a multi-component method that combines an optical flow tracker with a detector, and the detector part is slowly updated. However, TLD is very sensitive to tracking circumstances. SPL [31] follows the TLD paradigm for long-term tracking, and employs a SVM-based detector which is updated using the frames that produce the lowest SVM-objective for the training set. The repeated evaluation of the SVM objective, however, has a noticeable computational footprint.

LTCT [21] is a correlation filter tracker paying particular attention to long-term tracking, by integrating an online detector to re-detect the target in case of tracking failures. Furthermore, in the recent long-term challenge by VOT several new trackers were proposed [16], grouped into four families: short-trackers that do not implement re-detection nor model occlusion, short-term trackers with conservative updating, pseudo long-term trackers that do not return a box when the target is not visible, and re-detecting long-term trackers.

Recently, siamese paradigm-based trackers have been proposed which are shown to deliver state-of-art perfor-mance on long-term videos. Some such trackers include MBMD [16], DaSiamLT [16], SiamRPN++ [18] *etc.*. On top of the Siamese framework, these trackers use a SSD-MobielNet architecture, a region proposal framework and deep ResNet50 architectures, respectively, which help them to generate very accurate bounding box predictions.

As until recently few works only focused on long-term tracking, only few long-term benchmark datasets were available also. The most popular dataset to this year was UAV20L with 20 long videos captured from low-altitude unmanned aerial vehicles. Interestingly, in last few months several new benchmarks [33, 23, 24, 16, 22, 25] were introduced, several of whom focus on long-term tracking.

In this work we analytically analyze the impact of model decay, among others its model decay which leads to drift [30]. We propose a tracker that relies on Siamese instance search with rare and self-aware, and hence robust, model updates. Further, we revisit the importance of full image search, originally proposed in [14], for long-term tracking.

Short-term tracking. Short-term tracking has been one of the oldest and most popular fields in computer vision. Short-term trackers can be grouped in three categories: those that perform tracking by detection, tracking by correlation filters and tracking by similarity comparisons.

MDNet [27] is a recent successful tracking-by-detection tracker using deep learning. It employs a deep classification network as the detector, while the last layer is specialized for each video. MDNet employs a risky update strategy and a local search scheme, thus being sensitive tracking and sampling drift. EBT [37] goes beyond local search by generating instance-specific object proposals over the whole frame, using an SVM learned online. The online learning of EBT runs also the risk of model decay even in the stage

of proposal generation.

Very good performance is delivered by trackers based on discriminative correlation filters (DCF). Since the MOSSE tracker [3], many variants have been proposed. [9] uses multi-dimensional features. [12] proposes kernelized correlation filters. [5] incorporate robust scale estimation, while [7, 15] address the boundary effects caused by the circular shift. [8] learns the filters in the continuous spatial domain and [4] further improves [8] by improving on the model over-fitting. And, [20, 28, 8, 4] integrate convolutional features with the DCF framework. DCF trackers have shown great performance on tracking benchmarks of short videos [17, 36]. However, by relying on frequent, risky update schemes and local search, these trackers are susceptible to model decay in long videos.

Recently, a new and promising family is Siamese trackers. The first Siamese trackers [32, 2] relied on a tracking by similarity comparison strategy. They simply search for the candidate most similar to the original image patch of the target given in the starting frame, using a run-time fixed but learned *a priori* deep Siamese similarity function. Due to their no-updating nature, Siamese trackers are robust against several sources of error that cause model decay. Improved versions of siamese trackers include regional proposal frameworks (*e.g.* SiamRPN [19], DaSiamRPN [38]) and deeper backbones such as ResNet50 (SiamRPN++ [18]) which provide more accurate predictions and improve the tracking performance.

6. Experiments

6.1. Implementation Details

For LT-SINT we adopt most implementation details from SINT. We rely on an Imagenet pretrained VGG-16 network [29] up to the `relu4_3` layer, consisting of 10 convolution layers, 3 2-by-2 max pooling layers, and ReLU [26] nonlinearities. The first stage of global search identifies 10 locations and in the second stage, the search is performed over 11 scales: $2^{\{-0.4:0.08:0.4\}}$. In these two stages, the query patch of the initial target is resized to 32×32 pixels. For the third stage as well as the local search strategy, search is performed over 5 scales close to the previously estimated one: $\{0.9509, 0.9751, 1, 1.0255, 1.0517\}$ [2]. The input resolution for these cases is set to 64×64 pixels.

We train the LSTM decay recognition network offline on tracks returned by SINT with global search on ALOV [30]. In the LSTM network, the similarity maps are first encoded by a small convnet and the whole sequence is summarized by a two-layer LSTM network following by two fully-connected layers. The convnets comprise two 3×3 convolutional layers with strides of 8 and 16 output channels, respectively, and a 5×5 max-pooling layer. Model updates are performed with SGD with momentum for 10 iterations

	UAV20L	YouTubeLong
Local Search	39.8	23.3
Global Search	49.4	37.6

Table 1: Comparison between global search and local search on UAV20L and YouTubeLong, measured in AUC (%). Global search is advantageous when tracking for long duration where the target might disappear and reappear.

only, using a learning rate of 0.01 and a high momentum of 0.9 to not deviate too much from the previous model parameters. The classifier with parameters θ is trained using a binary cross entropy loss.

Datasets. We performed experiments on four long-term datasets, namely *UAV20L* [24], *YouTubeLong*, *OxUvA* [33] and VOT2018 long-term [16]. Here, *YouTubeLong* is a set of very long videos (around 25-30 minutes length each) that we gathered from YouTube, sparesely annotated every 25 frames. Note that the videos are annotated sparsely every 100 frames, as this suffices in practice [33]. It also contains "absent" annotations. In addition OTB50 dataset has been used for a shot-term tracking study. Note that building a tracker for short-term scenarios is beyond the scope of this paper, however, it is important that any tracker that works well for long-term should as well be acceptable for short-term scenarios.

Evaluation metrics. We use the AUC metric as in OTB [36] and UAV20L [24] adapted to also incorporate absent frames: any predicted box on a frame with an "absent" annotation is penalized aggressively with the *worst possible IoU*, namely 0. A detection is successful if the IoU is larger than a threshold, and the percentage of successfully tracked frames is calculated. We plot the curve for the AUC by varying the IoU threshold. As recommended in [33] for OxUva we use the true positive ration (TPR). For VOT2018 long-term dataset, Precision, Recall and F-score, as defined in [16], are used.

6.2. Evaluating Full Image Search

We first evaluate what is the effect of global and hybrid search in long-term tracking, while not performing any model updates. We show the results in Table 1. As expected, global search returns considerably better results, as it eliminates model decay caused by sampling drift. In fact, the difference is even more noticeable on the YoutubeLong, where videos are wilder and longer. Also, the performance of the method was studied for several different intervals T for the hybrid search, and it was found that the hybrid search with $T = 15$ permitted real time tracking at 25 fps without any negative impact on tracking accuracy.

	UAV20L	YouTubeLong
LT-SINT with		
no model updates	49.4	37.6
updates at every frame	50.4	25.3
siamese similarity updates	51.4	40.4
LSTM decay recognition network	53.9	41.9

Table 2: Comparison between the proposed self-aware model update and three baselines, measured in AUC (%). "no-upd" does not update the model. "blind-upd" updates every time. "sim-upd" updates the model when the similarity score of the predicted box is above the threshold (0.5). The proposed "selfaware-upd" is effective in handling model drift, achieving the best performance.

6.3. Evaluating the Decay Recognition Network

Next, we evaluate the decay recognition network and its effect on model decay in long-term tracking. We compare four variants: global search tracking with LT-SINT with *(i)* no model updates at all, *(ii)* with model updates in every frame, *(iii)* using a siamese 0.5 similarity threshold as a decay recognition network and *(iv)* using the LSTM decay recognition network. We show results in Table 2.

We observe that the LSTM decay recognition network is able to minimize model decay in both UAV20L and YoutubeLong. An interesting observation is that on the hard and long YouTubeLong videos updating at every frame causes a large drop in accuracy because of the large model decay. We conclude that the LSTM decay recognition network can predict well when an update is likely to cause contribute bias to the model, while allowing for adapting to the changing appearance of the target objects.

6.4. State-of-the-Art Benchmarking

Long videos. We compare the proposed Long-term Siamese Tracker with other state-of-the-art methods on the UAV20L, on YouTubeLong, OxUva and VOT2018 long-term datasets.

UAV20L, YouTubeLong, OxUva. Results related to these datasets are presented in Tables 3. It is observed that ECO-DEEP, although being a short-term tracker, works well on UAV20L even for videos up to 100 seconds long. The reason is that UAV20L videos undergo mild appearance changes of the targets. On the half-an-hour hard videos in YouTubeLong, however, ECO-DEEP is less successful, as several risky updates lead to model decay. And, on the diverse and very long OxUva videos LT-SINT has 10% margin from the second best tracker. To evaluate if the various trackers can locate the target till the end, we examine the last 20 seconds of the YoutubeLong videos, where LT-SINT maintains an excellent accuracy. We conclude that LT-SINT achieves top performance in long-term tracking for these datasets.

	UAV20L	YouTube Long	YouTube Long *(20 sec)*	OxUvA
ECO-DEEP [4]	42.7	7.1	1.4	39.5
TLD [14]	22.8	22.4	20.2	20.8
LTCT [21]	25.5	2.2	0.2	29.2
SPL [31]	35.6	—	—	—
SRDCF [7]	34.3	—	—	—
MUSTer [13]	32.9	—	—	—
SiamFC+R [33]	—	—	—	42.7
SiamFC [34]	—	—	—	39.1
MDNet [27]	—	—	—	47.2
SINT [32]	49.4	37.6	—	42.6
EBT [37]	—	—	—	32.1
BACF [10]	—	—	—	31.6
Staple [1]	—	—	—	27.3
LT-SINT	**52.4**	**42.1**	**39.5**	**57.9**

Table 3: State-of-the-art comparison, measured in AUC (%). LT-SINT outperforms other trackers on all four datasets.

VOT2018 long-term. On this dataset, LT-SINT provides an F-score of 0.536, which is a bit behind some of the state-of-art trackers. However, this should not be counted as a limitation of LT-SINT. Among the state-of art trackers, SiamRPN++ uses a deep network architecture, namely ResNet50, DaSiamLT uses a region proposal framework and MBMD employs a SSD-MobileNet architecture. Clearly, the use of these frameworks and larger training datasets on top of a plain Siamese network improves the predictions and leads to higher values of F-score. Since the goal of this paper is to present a theoretical analysis of model bias and pave way for research to reduce it, we limited ourselves to trackers based on basic Siamese matching. In this context, a fair comparison would be with SiamFC, and we see that adding the decay recognition network already helps in improving the performance of our tracker.

Short videos. While model decay is visible in long videos, for completeness we evaluate LT-SINT also on the short videos of OTB [36], using the standard AUC metric. Without exhaustive hyperparameter optimizations, LT-SINT scores 59.8 AUC, which is in the same ballpark as other popular short-term trackers that rely on aggressive model updating, like ECO [4] (69.1), CFNet [34] (58.6) or SiamFC [2] (58.2). The slightly lower performance on short term videos can be expected by long-term trackers as in effect, long-term suited trackers have an additional constraint.

7. Conclusion

This paper highlights the negative impact that adverse bounding box predictions have to the tracker models, introducing an unvoidable bias term to the learning dynamics of the tracker and leading to model decay. We examine empirically the effect the various tracking challenges have on model decay. Based on the insights we propose two simple

modifications to siamese trackers, such that model updates are possible without leading to significant model decay. Experiments on four long-term and one short-term tracking benchmarks show that the proposed tracker is accurate and does not suffers from model decay even in wild, 30-minute long videos.

References

[1] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. S. Torr. Staple: Complementary learners for real-time tracking. In *CVPR*, June 2016. 8

[2] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr. Fully-convolutional siamese networks for object tracking. In *ECCV VOT workshop*, 2016. 4, 7, 8

[3] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui. Visual object tracking using adaptive correlation filters. In *CVPR*, 2010. 7

[4] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg. Eco: Efficient convolution operators for tracking. In *CVPR*, 2017. 4, 7, 8

[5] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg. Accurate scale estimation for robust visual tracking. In *BMVC*, 2014. 4, 7

[6] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg. Adaptive decontamination of the training set: A unified formulation for discriminative visual tracking. In *CVPR*, June 2016. 1, 3, 4

[7] M. Danelljan, G. Häger, F. Khan, and M. Felsberg. Learning spatially regularized correlation filters for visual tracking. In *ICCV*, 2015. 7, 8

[8] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg. Beyond correlation filters: learning continuous convolution operators for visual tracking. In *ECCV*, 2016. 7

[9] M. Danelljan, F. Shahbaz Khan, M. Felsberg, and J. van de Weijer. Adaptive color attributes for real-time visual tracking. In *CVPR*, 2014. 7

[10] H. K. Galoogahi, A. Fagg, and S. Lucey. Learning background-aware correlation filters for visual tracking. In *ICCV*, 2017. 8

[11] S. Hare, A. Saffari, and P. H. Torr. Struck: Structured output tracking with kernels. In *ICCV*, 2011. 4

[12] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *TPAMI*, 2015. 4, 7

[13] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao. Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking. In *CVPR*, 2015. 8

[14] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *TPAMI*, 34(7):1409–1422, 2010. 6, 8

[15] H. Kiani Galoogahi, T. Sim, and S. Lucey. Correlation filters with limited boundaries. In *CVPR*, 2015. 7

[16] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pfugfelder, L. C. Zajc, T. Vojir, G. Bhat, A. Lukezic, A. Eldesokey, G. Fernandez, and et al. The sixth visual object tracking vot2018 challenge results, 2018. 6, 7

[17] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, L. Cehovin, G. Fernandez, T. Vojir, G. Hager, G. Nebehay, and R. Pflugfelder. The visual object tracking vot2015 challenge results. In *ICCV VOT workshop*, 2015. 1, 7

[18] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan. SiamRPN++: Evolution of Siamese Visual Tracking with Very Deep Networks. In *CVPR*, 2019. 6, 7

[19] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High performance visual tracking with siamese region proposal network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 7

[20] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang. Hierarchical convolutional features for visual tracking. In *ICCV*, 2015. 7

[21] C. Ma, X. Yang, C. Zhang, and M.-H. Yang. Long-term correlation tracking. In *CVPR*, 2015. 5, 6, 8

[22] A. Milan, L. Leal-Taixé, I. D. Reid, S. Roth, and K. Schindler. MOT16: A benchmark for multi-object tracking. *CoRR*, 2016. 6

[23] A. Moudgil and V. Gandhi. Long-term visual object tracking benchmark. *arXiv preprint arXiv:1712.01358*, 2017. 6

[24] M. Mueller, N. Smith, and B. Ghanem. A benchmark and simulator for uav tracking. In *ECCV*, 2016. 6, 7

[25] M. Müller, A. Bibi, S. Giancola, S. Al-Subaihi, and B. Ghanem. Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In *ECCV*, 2018. 6

[26] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010. 7

[27] H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*, 2016. 4, 6, 8

[28] Y. Qi, S. Zhang, L. Qin, H. Yao, Q. Huang, J. Lim, and M.-H. Yang. Hedged deep tracking. In *CVPR*, 2016. 7

[29] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 7

[30] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: an experimental survey. *TPAMI*, 36(7):1442–1468, 2014. 1, 4, 6, 7

[31] J. S. Supancic and D. Ramanan. Self-paced learning for long-term tracking. In *CVPR*, 2013. 6, 8

[32] R. Tao, E. Gavves, and A. W. M. Smeulders. Siamese instance search for tracking. In *CVPR*, 2016. 3, 4, 5, 7, 8

[33] J. Valmadre, L. Bertinetto, J. F. Henriques, R. Tao, A. Vedaldi, A. W. M. Smeulders, P. H. S. Torr, and E. Gavves. Long-term tracking in the wild: A benchmark. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part III*, 2018. 3, 6, 7, 8

[34] J. Valmadre, L. Bertinetto, J. F. Henriques, A. Vedaldi, and P. H. Torr. End-to-end representation learning for correlation filter based tracking. 2017. 8

[35] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *CVPR*, 2013. 4

[36] Y. Wu, J. Lim, and M.-H. Yang. Object tracking benchmark. *TPAMI*, 37(9):1834–1848, 2015. 1, 7, 8

[37] G. Zhu, F. Porikli, and H. Li. Beyond local search: Tracking objects everywhere with instance-specific proposals. In *CVPR*, 2016. 6, 8

[38] Z. Zhu, Q. Wang, B. Li, W. Wu, and W. Hu. Distractor-aware Siamese Networks for Visual Object Tracking. In *CVPR*, 2019. 7