A Memory Model based on the Siamese Network for Long-term Tracking

Hankyeol Lee*[0000−0003−1559−5791], Seokeon Choi*[0000−0002−1695−5894], and Changick Kim[0000−0001−9323−8488]

School of Electrical Engineering, KAIST, Republic of Korea
{hankyeol, seokeon, changick}@kaist.ac.kr

Abstract. We propose a novel memory model using deep convolutional features for long-term tracking to handle the challenging issues, including visual deformation or target disappearance. Our memory model is separated into short- and long-term stores inspired by Atkinson-Shiffrin Memory Model (ASMM). In the tracking step, the bounding box of the target is estimated by the Siamese features obtained from both memory stores to accommodate changes in the visual appearance of the target. In the re-detection step, we take features only in the long-term store to alleviate the drift problem. At this time, we adopt a coarse-to-fine strategy to detect the target in the entire image without the dependency of the previous position. In the end, we employ Regional Maximum Activation of Convolutions (R-MAC) as key criteria. Our tracker achieves an F-score of 0.52 on the LTB35 dataset, which is 0.04 higher than the performance of the state-of-the-art algorithm.

Keywords: Long-term tracking, Atkinson-Shiffrin Memory Model, Siamese network, Regional Maximum Activation of Convolutions

1 Introduction

Visual object tracking is one of the most popular tasks in computer vision with many applications such as surveillance, traffic control, and autonomous driving. Given a target with a bounding box in the first frame, the goal of visual tracking is to estimate the bounding boxes of the target in the remaining frames of a video sequence. In particular, short-term object tracking, which assumes that the target is always located in the image, has greatly advanced with various tracking benchmarks [33, 29, 20] and visual object tracking challenges [18, 9] over the last decade.

Meanwhile, long-term tracking has also received increasing attention with the introduction of a new benchmark [24]. Unlike in short-term tracking, the target could disappear over time in long-term tracking due to full occlusion or out-of-view. The long-term tracking task also involves a variety of challenging problems (e.g., viewpoint change, object deformation, and fast motion), because

* These two authors contributed equally.

it deals with a relatively long sequence. Therefore, additional procedures are required to achieve robust performance in long-term tracking.

Convolutional Neural Networks (CNNs) have recently demonstrated remarkable performance in many computer vision tasks [15, 12, 34] and many CNN-based short-term trackers have been proposed [26, 3, 30, 2]. SiamFC [2], one of the successful CNN-based trackers, has especially achieved high performance in real-time on the short-term tracking benchmarks by using a fully-convolutional Siamese network. However, although SiamFC could exploit the expressive power of deep convolutional networks, it is difficult to handle the challenging issues in long-term tracking without additional procedures.

The reason why SiamFC is not suitable for long-term tracking is as follows. Firstly, SiamFC easily fails to track the target if the target appearance changes significantly since tracking works with only the target appearance given in the first frame without an online learning process. Secondly, the target re-detection process is not implemented explicitly, so the re-entered target must be detected in the same way as tracking. However, the tracking process in SiamFC not only exploits the prior knowledge about the target position, but also covers a region of interest which is the neighbor region of the target position estimated in the previous frame. Eventually, this manner causes a strong dependency on the previous position, which may lead to re-detection failure when the target re-enters far from its previous position.

In this paper, we propose a Memory Model via the Siamese network for Long-term Tracking (MMLT) to address the problem for target appearance variation and the dependency on the previous position. Our long-term tracker is a novel approach that applies deep features extracted from the Siamese and VGG networks to Atkinson-Shiffrin Memory Model (ASMM) [1], unlike MUSTer [14] based on the traditional descriptors such as HOG [10] and SIFT [21]. MMLT consists of three parts: tracking, re-detection, and memory management. The structure of the memory for long-term tracking is divided into short- and long-term stores depending on period and manner in which the memory is stored. Tracking and re-detection processes are performed based on this memory model.

In the tracking process, the position of the target is estimated based on the response map of the Siamese network. At this time, we take a weighted sum of the features in both short- and long-term stores to effectively capture appearance variations. In contrast, we employ features only in the long-term store to constrain the drift problem in the re-detection process. Furthermore, we adopt a coarse-to-fine strategy to detect the target in the entire image without the dependency of the previous position. Firstly, we collect several candidates that retain similar semantic meanings with the target to pick out coarse positions in the entire image. Next, we select the final candidate and refine the target position by applying the Siamese network in the long-term store to each candidate. Regional Maximum Activation of Convolutions (R-MAC) [31], which has been proposed for image retrieval, is applied to determine whether the re-detected object is the target or not. We demonstrate that our tracker achieves the state-of-the-art performance on the long-term tracking benchmark [24].

2 Related works

Long-term tracking is characterized by the disappearance of the target object, so re-detection is required as a key process. In addition, since the length of long-term tracking sequence is considerably longer than that of the short-term, the updating scheme capable of accommodating changes in the visual appearance of the target greatly affects performance. Therefore, we briefly introduce how recent long-term trackers [17, 14, 23] are used to overcome the critical issues of long-term tracking.

Kalal *et al.* [17] proposed a new tracking framework, TLD, that decomposes long-term tracking into tracking (T), learning (L), and detection (D). First, the tracker estimates the position of the moving object based on the median-flow tracker [16]. Next, the detector determines the presence of the target in a cascade manner over the entire area of the image and modifies the tracker if necessary. At this time, three processes are performed for detection: patch variance, ensemble classifier, and NN-classifier. Assuming that the tracker and detector may fail, the learning process estimates errors based on P-N learning and learns the tracker and detector more robustly.

Hong *et al.* [14] firstly adopted Atkinson-Shiffrin Memory Model (ASMM) [1], also known as the multi-store model, for long term tracking to cope with appearance changes of the object. The MUlti-Store Tracker (MUSTer) consists of short- and long-term memory stores. An Integrated Correlation Filter (ICF), which is based on Kernelized Correlation Filter (KCF) [13] and Discriminative Scale Space Correlation Filter (DSSCF) [4], is used for short-term tracking. Furthermore, they add a complementary element based on keypoint matching-tracking [27] and MLESAC estimation [32] as a long-term component. In the end, they design a forgetting curve to model remembering-forgetting [7].

Lukežič *et al.* [23] suggested a Fully-Correlated Long-term Tracker (FCLT) that applies discriminative correlation filters to long-term tracking. This is decomposed into the short-term tracker and detector. These two components are modeled by learning DCFs at different time scales. Especially, the problem that the correlation filter could not be applied to detection is solved efficiently.

3 Memory model for long-term tracking (MMLT)

According to Atkinson-Shiffrin Memory Model (ASMM) [1], human memory is divided into three separate components: a sensory register, a short-term store, and a long-term store. The sensory register acts as a buffer for passing information to the short-term store. The short-term store retains the information for a short time, and repeated memories in the short-term store are transferred to the long-term store with semantic information. These memories can be recalled by the retrieval process as needed.

Inspired by this memory model, we propose a memory model for long-term tracking, which is divided into short- and long-term stores. It is not the first time that this multi-store model has been used for long-term tracking. The

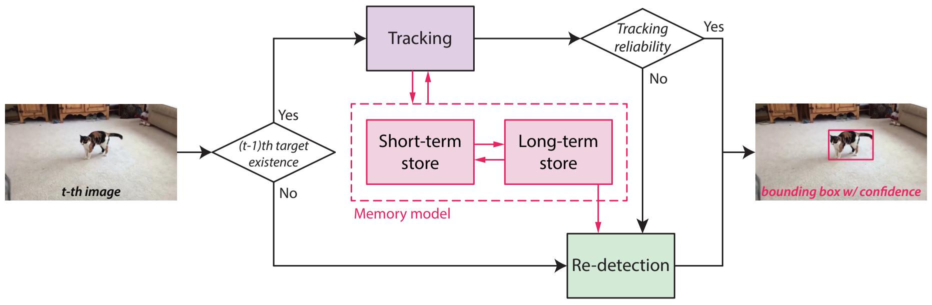

Fig. 1. The overall flowchart of our tracker. The black arrows represent how the tracking and re-detection procedures work depending on two main conditions, and the red arrows indicate how memory is stored and delivered.

MUlti-Store Tracker (MUSTer) [14] is a representative long-term tracker based on ASMM, which is described in detail in the previous section. They employ the traditional descriptors (e.g., HOG [10], color attributes [6], and SIFT [21]) as sensory information, whereas we incorporate rich features extracted from deep neural networks into the memory model to accommodate large changes in the visual appearance of the target.

In this paper, we introduce a novel Memory Model based on the Siamese network for Long-term Tracking (MMLT) to effectively deal with the problems for visual deformation and target disappearance. The short-term store $\mathcal{S}$ retains the Siamese features of the target, and the long-term store $\mathcal{L}$ holds both the Siamese features and semantic features as follows:

$$\mathcal{S} = \{f(z_{S_i})\}_{i=1}^{N_S}, \quad \mathcal{L} = \{f(z_{L_i}), g(z_{L_i})\}_{i=1}^{N_L}, \tag{1}$$

where $f(\cdot)$ and $g(\cdot)$ denote the Siamese feature and the semantic feature extracted from deep neural networks. z is an exemplar image that covers a larger region than the estimated bounding box, which is discussed in SiamFC [2]. We note that the exemplar image z and the estimated target region $\hat{z}$ (e.g., bounding box) are different. S_i and L_i indicate the frame index of the stored features, and i indicates the order entered in each memory store. The memory capacities of the short- and long-term stores are $N_S(= 60)$ and $N_L(= 40)$, respectively.

The entire tracking process including re-detection is executed based on this memory model, which is depicted in Fig. 1. The proposed long-term tracker comprises three parts: tracking, re-detection, and memory management. If the target position in the previous frame is successfully estimated, the tracking process is performed with the Siamese features in the short- and long-term stores (Section 3.1). On the contrary, if this estimation fails in the previous frame or if the current tracking result is unreliable, the re-detection process is operated based on the features in the long-term store (Section 3.2). In the end, Section 3.3 gives details on how memory is stored in the short-term store, how short-term memory is transferred to long-term memory, and memory limits for the short- and long-term stores.

Algorithm 1 Tracking

```
1: Input: Searching region $x$, short-term store $\mathcal{S}$, long-term store $\mathcal{L}$, P-Net $f$
2: Output: Estimated target $\hat{z}$, tracking reliability flag
3: Compute the response using Eq. (3)
4: if (Satisfy Eq. (6)) then
5:     Tracking reliability flag $\leftarrow 1$
6:     Estimated target $\leftarrow \hat{z}$
7:     $N_T \leftarrow N_T + 1$
8: else
9:     Tracking reliability flag $\leftarrow 0$
10:    Estimated target $\leftarrow \emptyset$
11:    $N_T \leftarrow 0$
12: end if
```

3.1 Tracking

Our network for tracking is the same as SiamFC [2]. Thus, we briefly review the process of SiamFC. They denote an exemplar image given in the first frame and a searching region as z and x, which have a dimension of $127 \times 127 \times 3$ and $255 \times 255 \times 3$, respectively. The output feature of the Siamese network, which we call it P-Net, is denoted by $f(\cdot)$. The formulation of the SiamFC to obtain response map can be written as:

$$y = f(z) * f(x), \tag{2}$$

where $*$ denotes the correlation operation. The center position and target size can be estimated from this response map. And, the searching region x of the next frame is formed based on the center position of the estimated target. However, since SiamFC only utilizes the exemplar image given in the first frame, it is difficult to capture target appearance variations during the tracking.

To address this problem, $f(z)$ in Eq. (2) is replaced by combining the output features of P-Net in the short- and long-term stores. Thus, the response map can be calculated by the following equation,

$$y = f_M(\mathcal{S}, \mathcal{L}) * f(x), \tag{3}$$

where $f_M(\mathcal{S}, \mathcal{L})$ denotes the combined features of the short- and long-term stores. $f_M(\mathcal{S}, \mathcal{L})$ is obtained as a weighted sum of short-term features $f_S(\mathcal{S})$ and long-term features $f_L(\mathcal{L})$:

$$f_M(\mathcal{S}, \mathcal{L}) = \lambda \exp(-\frac{N_T}{h_M}) f_S(\mathcal{S}) + (1 - \lambda \exp(-\frac{N_T}{h_M})) f_L(\mathcal{L}), \tag{4}$$

where λ denotes a weight parameter to control the relative importance between short-term memory and long-term memory, h_M is a constant to control the strength of the time, and N_T is the number of successive frames with reliable tracking results. According to this equation, as the number of consecutive frames with reliable tracking results increases, the weight of $f_S(\mathcal{S})$ decreases and that of $f_L(\mathcal{L})$ increases. It means that the tracker initially focuses on short-term memory to accommodate changes in the visual appearance of the target, but switch focus to long-term memory since reliable information is transferred to the long-term store over time. Each feature is calculated as a combination of exemplar images extracted from selected frames as follows:

$$f_S(\mathcal{S}) = \frac{\sum_{i=1}^{N_S} \exp(-\frac{i}{h_S}) f(z_{S_i})}{\sum_{i=1}^{N_S} \exp(-\frac{i}{h_S})}, \quad f_L(\mathcal{L}) = \alpha_L f(z_{L_1}) + (1-\alpha_L) \frac{\sum_{i=2}^{N_L} f(z_{L_i})}{N_L - 1}, \tag{5}$$

where h_S is a constant to control the impact of the short-term store length. $f_S(\mathcal{S})$ is computed by assigning a higher weight to the features according to the order entered early in the short-term store to alleviate the drift problem. On the other hand, we give only the first frame a high weight α_L since the first stored long-term memory ($L_1 = 1$) is always ground truth. We note that the first stored short-term memory might not be ground truth since short-term memory is often reset, which is discussed in the next section.

We need to decide whether or not the re-detection process should be applied, so we set a criterion to decide if the tracking result is reliable. To determine whether the tracking result is reliable, we use the maximum value of the response map $y^* = \max(f_M(\mathcal{S}, \mathcal{L}) * f(x))$. Let $\bar{y}$ be the average maximum value of the response maps calculated by the recent 40 reliable frames. We assume that the tracking result is reliable if the ratio between y^* and $\bar{y}$ is higher than a predefined threshold τ_1 (=0.6) for comparison of the Siamese features, i.e.,

$$y^*/\bar{y} > \tau_1. \tag{6}$$

If this criterion is satisfied, then the center position and target size are estimated from the response map in the same manner as in SiamFC (e.g., upsampling for accurate localization and searching over five scales). Thus, we can get the target region $\hat{z}$ from the above information. However, target estimation is postponed to the re-detection step if it fails to satisfy the criterion, which is introduced in the next section. The overall process of tracking is summarized in Algorithm 1.

3.2 Re-detection

If the criterion of Eq. (6) is not satisfied, the tracking result is considered unreliable and eventually the re-detection step proceeds. We adopt a coarse-to-fine strategy to detect the target in the entire image without the dependency of the position estimated in the previous frame. Only the information in the long-term store is employed to re-detect the target, and the short-term store is reset to handle the drift problem.

Firstly, we use the last layer (*conv*5) of the pre-trained VGGNet to collect the coarse positions of the candidates which have similar semantic meanings with the target in the entire image. This is denoted by $g(\cdot)$, and we call it S-Net. Let I denote the entire image, which has a size of $W \times H \times 3$. The spatial size of the

Algorithm 2 Re-detection

```
1: **Input**: Image $I$, long-term store $\mathcal{L}$, P-Net $f$, S-Net $g$
2: **Output**: Estimated target $\hat{z}$, *target existence flag*
3: Short-term store $\mathcal{S}$ is reset
4: Get candidates using Eq. (7)
5: Select the one candidate using Eq. (8)
6: Estimated target $\leftarrow \hat{z}$
7: **if** (Satisfy Eq. (9) and Eq. (10) and Eq. (11)) **then**
8:     *Target existence flag* $\leftarrow 1$
9: **else**
10:    *Target existence flag* $\leftarrow 0$
11: **end if**
```

output features $g(I)$ is reduced to the size of $\frac{W}{16} \times \frac{H}{16} \times 512$ due to the existence of pooling layers. Then, the semantic response maps are calculated as follows:

$$y_i^s = g(z_{L_i}) * g(I), \quad i = 1, \dots, N_L. \tag{7}$$

Since the spatial size of the features is smaller than that of the entire image, we could not estimate the exact position from the semantic response map y_i^s. So, we only obtain the coarse searching regions x_k of N_D (=3) candidates with high response values. After that, the best candidate is selected by P-Net as follows:

$$x = \underset{x_k}{\operatorname{argmax}}\,(\max(f_L(\mathcal{L}) * f(x_k))), \quad k = 1, \dots, N_D, \tag{8}$$

where x_k is the searching region of each candidate. Unlike searching for the target over five scales, we change it to fifteen scales to effectively deal with scale variations in the re-detection process. Eventually, the searching region x of the best candidate is selected by Eq. (8), and then the target region $\hat{z}$ of this candidate is estimated from the maximum value of $f_L(\mathcal{L}) * f(x)$.

To determine whether the final candidate is the target, three criteria are defined. The first is how the Siamese feature of the final candidate and that of exemplar images in the long-term store are similar, which is expressed as follows:

$$\frac{\max(f_L(\mathcal{L}) * f(x))}{\sum_{i=1}^{N_L} f(z_{L_i}) * f(z_{L_i}) / N_L} > \tau_2, \tag{9}$$

where τ_2 (=0.35) is a certain threshold for comparison of the Siamese features. Short-term memory is not considered because it has been reset.

The second is related to the retrieval process occurring in ASMM [1]. We use the Regional Maximum Activation of Convolutions (R-MAC) vector [31], which has been proposed for image retrieval. The R-MAC feature vector $\mathbf{h}(\cdot)$ is a compact representation of the CNN response map $g(\cdot)$ with semantic meaning. It is robust to scale and translation due to sampling the response map at multiple scales and aggregating the regional vectors. The second criterion based on the semantic meaning regardless of position and size is described as follows:

$$\frac{1}{N_L}\sum_{i=1}^{N_L}\langle \mathbf{h}(g(z_{L_i})), \mathbf{h}(g(z))\rangle > \tau_h, \tag{10}$$

where $\langle\cdot,\cdot\rangle$ denotes the cosine similarity between two vectors. τ_h (=0.6) is a threshold for comparison of the R-MAC vectors, and we call it the R-MAC threshold.

The third is designed to prevent the tracker from being confused by background objects with similar appearance. We extract the exemplar images $\{z_j^n\}_{j=1}^{N_n}$ of N_n(=16) negative samples in the first frame, which are target-like distractors chosen based on P-Net and S-Net. To prevent the target from accidentally being classified as a negative sample, the negative samples are extracted only in the first frame. The third criterion is computed as follows:

$$\max(f_L(\mathcal{L}) * f(x)) > \max(f(z_j^n) * f(x)), \quad j = 1, \ldots, N_n. \tag{11}$$

According to the equation, the final candidate must be more similar to the positive samples in the long-term store than all the negative samples.

If the final candidate satisfies these three criteria, we determine the candidate as the target and proceed with the tracking process from the next frame. The overall of the re-detection process is summarized in Algorithm 2.

3.3 Memory management

According to the memory management in ASMM [1], only a limited amount of the information can be held in the short-term store. Repeated memories among them are moved to the long-term store, and they are reset in the short-term store over time. Unlike the short-term store, memories in the long-term store are maintained for a long time. In particular, the frequently used memory lasts long, but the memory that is not used often is forgotten.

In this section, we model how memory information moves and disappears from memory stores based on the above memory model. The memory management step proceeds only when the tracking result of the current frame is reliable. Firstly, acceptable information from the sensory register is transferred to the short-term store as Siamese features via the following criterion as follows:

$$\frac{\max(f_M(\mathcal{S},\mathcal{L}) * f(x))}{\sum_{i=1}^{N_L} f(z_{L_i}) * f(z_{L_i})/N_L} > \tau_3, \tag{12}$$

where τ_3 (=0.5) is a certain threshold. If the short-term store is full, the short-term store is managed in the first in / first out manner during tracking since the short-term store has a limited capacity of N_S(=60). However, if the tracking result is unreliable, all short-term memories disappear entirely.

On the other hand, moving the short-term information to the long-term store need to be treated more carefully. Therefore, the information is transferred only

Algorithm 3 Memory management

```
1: Input: Estimated target $\hat{z}$, short-term store $\mathcal{S}$, long-term store $\mathcal{L}$
2: Output: Short-term store $\mathcal{S}$, long-term store $\mathcal{L}$
3: if (Satisfy Eq. (12)) then
4:     $z$ is transferred to the short-term store $\mathcal{S}$
5:     if (Satisfy Eq. (10) and Eq. (11)) then
6:         $z$ is transferred to the long-term store $\mathcal{L}$
7:     end if
8: end if
```

when the criterion for Eq. (11) succeeds after Eq. (10) is satisfied consecutively during 10 frames. At this time, both the Siamese and semantic features are transferred to the long-term store. This is summarized in Algorithm 3.

In the end, we set the forgetting curve [7] for each long-term memory. If additional memory is entered after long-term memory capacity is full, the memory with the smallest forgetting curve value disappears. However, we always keep the first memory in the long-term store since it is the ground truth given in the first frame. This forgetting curve is modeled as follows:

$$c_i = \begin{cases} \min(1, k_c c_i), & \text{if } \langle \mathbf{h}(g(z_{L_i})), \mathbf{h}(g(z)) \rangle > \tau_h, \\ p_i \exp(-a_i/h_c), & \text{otherwise,} \end{cases} \quad i = 2, \ldots, N_L, \tag{13}$$

where c_i denotes the forgetting curve of the i-th long-term memory, k_c (>1) represents the reinforcement parameter, and h_c indicates the memory strength parameter. a_i is the age value that is initialized to 0 ($a_i = 0$) if the above condition is satisfied, otherwise it is increased by 1 ($a_i = a_i + 1$). p_i is the baseline of the forgetting curve, which has a value of 1 ($p_i = 1$) when memory is initially stored, but it is re-initialized to a reinforced value $p_i = \min(1, k_c c_i)$ if the above condition is satisfied. In summary, when the newly entered memory is similar to the existing stored memory, it means that the forgetting curve value c_i corresponding to the existing memory is strengthened by k times and the age value a_i is initialized. Otherwise, the age value a_i is increased by 1, and the forgetting curve value c_i of that memory decreases depending on the age.

3.4 Tracking with MMLT

Our three main parts introduced in the previous sections are now integrated as a long-term tracker, which is summarized in Algorithm 4. For the interaction of these three parts, we designate two flags called *tracking reliability* and *target existence flag*. Our tracker is initialized by transferring the ground truth information given in the first frame to the short- and long-term stores. Unlike short-term tracking, which only predicts the bounding box of the target, long-term tracking requires not only the bounding box prediction but also the confidence score assignment to the estimated bounding box. The confidence score should be high if the target is present and vice versa. Thus, we assign the confidence score

Algorithm 4 MMLT

```
1: Input: Images $I$, target in the first frame $\hat{z}_1$
2: Output: Estimated target $\hat{z}$, confidence score $v$
3: Initialization
4: Target existence flag & Tracking reliability flag $\leftarrow 1$
5: for $t = 2, 3, \ldots, N$ do
6:     if (Target existence flag $= 1$) then
7:         Forgetting curve $c_i$ management using Eq. (13)
8:         Tracking (Algorithm 1)
9:         if (Tracking reliability flag $= 1$) then
10:            Memory management (Algorithm 3)
11:        end if
12:    end if
13:    if (Target existence flag $= 0$) or (Tracking reliability flag $= 0$) then
14:        Re-detection (Algorithm 2)
15:    end if
16:    Compute the confidence score on the estimated target using Eq. (14)
17: end for
```

differently depending on whether the target exists or not as follows:

$$v = \begin{cases} \frac{1}{N_L} \sum_{i=1}^{N_L} \langle \mathbf{h}(g(z_{L_i})), \mathbf{h}(g(z)) \rangle, & \text{if } \textit{target existence flag} = 1, \\ \frac{\alpha_v}{N_L} \sum_{i=1}^{N_L} \langle \mathbf{h}(g(z_{L_i})), \mathbf{h}(g(z)) \rangle, & \text{otherwise.} \end{cases} \tag{14}$$

Since $\alpha_v = \frac{\max(f_L(\mathcal{L}) * f(x))}{\sum_{i=1}^{N_L} f(z_{L_i}) * f(z_{L_i}) / N_L}$ is always smaller than 1, the confidence score is assigned a small value when the target does not exist.

4 Experimental results

4.1 Dataset

We experimented with a long-term tracking dataset, called LTB35 [24], to compare the proposed MMLT to other trackers. The LTB35 dataset is officially used in the VOT2018 long-term challenge [19] and has the following characteristics: It contains a total of 146,847 frames with more than 1,000 frames for each sequence of 35 categories. Two sequences (*following* and *liverRun*) are even more than 10,000 frames. The number of frames in each sequence is much longer than the previous short-term datasets [33, 29, 20], which makes the task challenging. In addition, since the target disappears for an average of about 12.4 times, a re-detection process is considered to be particularly important. The resolution of the sequences is between 1280×720 and 290×217, and the size of the target is different for each image and continuously changes over time. The target of each sequence is annotated with an axis-aligned bounding-box, and various objects are categorized such as persons, animals, and vehicles.

Table 1. The maximum F-score for each tracker.

Method	**Type**	**Maximum F-score**
MMLT	LT_1	0.52
FCLT [23]	LT_1	0.48
SiamFC [2]	ST_1	0.40
ECO [3]	ST_1	0.35
ECOhc [3]	ST_1	0.33
CSRDCF [22]	ST_0	0.33
CREST [30]	ST_0	0.33
PTAV [8]	LT_0	0.31
BACF [11]	ST_0	0.31
MUSTER [14]	LT_1	0.29
KCF [13]	ST_0	0.27
TLD [17]	LT_1	0.27
SRDCF [5]	ST_0	0.26
LCT [25]	LT_0	0.25
CMT [28]	LT_1	0.22

4.2 Evaluation protocol

The proposed MMLT was evaluated by the evaluation protocol of the VOT2018 long-term challenge [19], which tracks a target from the first frame to the end of the sequence without re-sets. This evaluation protocol was automatically performed by the VOT toolkit [19], which automatically computed the highest F-score for each sequence based on a detection-like precision-recall plot by using confidence scores assigned to bounding boxes of the target. The experiments were performed using MATLAB R2017a on a system with Intel(R) core(TM) i7-4770 3.40 GHz processor and a single NVIDIA GTX 1080 Ti with 11GB RAM.

4.3 The VOT-LT2018 benchmark

We compared the tracking performance of our MMLT to that of various trackers based on the maximum F-score. Table 1 shows the maximum F-scores of the state-of-the-art trackers which have been reported in the LTB35 dataset paper [24]. Evaluated trackers are separated into four categories as follows: short-term tracker (ST_0), short-term tracker with conservative updating (ST_1), pseudo long-term tracker (LT_0), and re-detecting long-term tracker (LT_1). The following taxonomy has been introduced explicitly in [24].

The proposed MMLT algorithm can be classified as the re-detecting long-term tracker (LT_1) since our tracker judges tracking failure and performs target re-detection. FCLT [23], which is also the re-detecting long-term tracker (LT_1), achieved the highest performance among the existing methods. However, we achieved an F-score of 0.52 on the LTB35 dataset, which is 0.04 higher than the performance of the state-of-the-art algorithm. Other long-term trackers, however, produced lower performance than existing short-term trackers. This in-

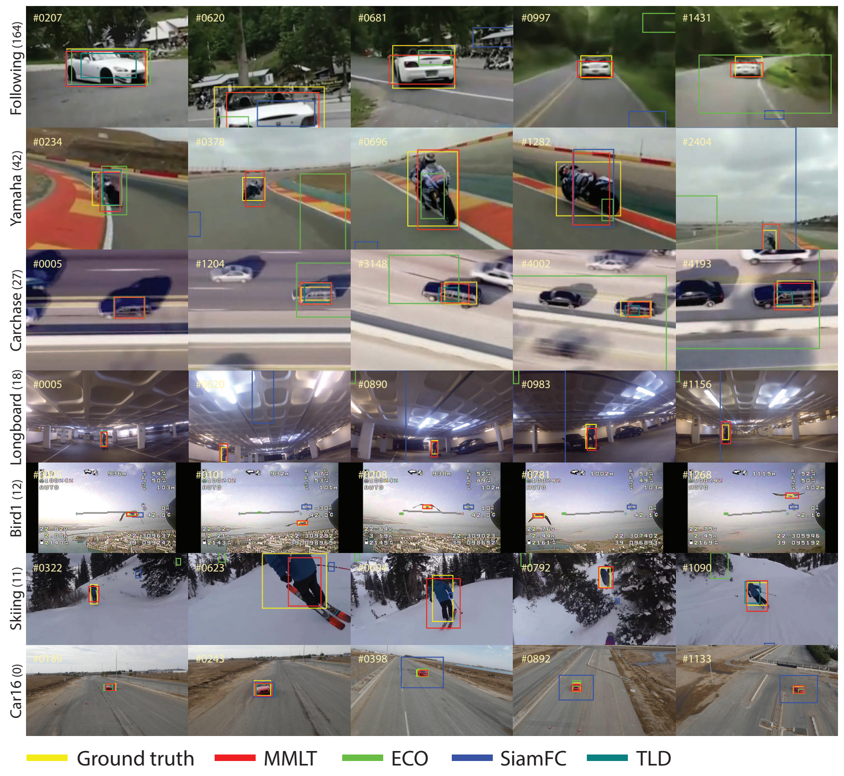

Fig. 2. Qualitative results of the proposed MMLT, ECO [3], SiamFC [2], and TLD [17] in representative frames of challenging seven sequences. Sequences are sorted based on the number of target disappearances, which are indicated by the number in parentheses.

dicates how important and careful the process of re-detection or updating the visual model should be.

We have selected several trackers to analyze the qualitative results with our tracker, which are visualized in Fig. 2. The tracking results for a total of seven sequences are arranged in descending order for the number of target disappearances. The best performer among short-term trackers, ECO [3], tracked the target well even though the viewpoint and scale changed significantly in the *Car* sequence. The correlation filter-based trackers like ECO present a strong advantage in the sequences where the target object does not disappear from the image. On the other hand, SiamFC [2], one of the famous CNN-based short-term trackers, did not adapt as well as ECO to visual appearance variation because an online updating module was not implemented.

In the remaining six sequences in Fig. 2, ECO has easily drifted to other objects or background because it did not deal with the process of overcoming the disappearance of the target. SiamFC often detected objects in the *Yamaha* sequence where the background clutter was relatively small and the target was noticeable compared to the background. TLD has often succeeded in detecting the target in the *Carchase* sequence where the target often disappears but performance was not as good. In short, all of the existing trackers solved the problem only in some attributes and their performance degraded significantly in challenging problems involving complex attributes.

On the other hand, the proposed MMLT achieved robust tracking performance even in challenging sequences where the target frequently disappears and the visual appearance of the object changes significantly. In particular, in the *Bird* sequence, all previous algorithms completely failed to detect and track the target, whereas the proposed MMLT method obtained better performance by the reliable re-detection procedure and online updating manner based on ASMM. We also achieved F-scores of 0.50, 0.44, 0.40, and 0.51 for *Following*, *Yamaha*, *Longboard*, and *Skiing* sequences, respectively. Therefore, the proposed MMLT proved to be a robust long-term tracker compared to the existing methods. The following results demonstrated that MMLT is a more robust long-term tracker than conventional methods. In the end, the average execution speed of MMLT in the long-term experiment provided by the VOT toolkit [19] achieved 6.15 fps.

4.4 Performance analysis of detailed algorithms

To analyze the sub-algorithms of the proposed MMLT in detail, we created several versions. The maximum F-score for each sequence was calculated, which is summarized in Fig. 3. All sequences are ordered based on how much the proposed final version is superior to the other versions, i.e., the final version in the *Car16* sequence was the most superior to the other versions.

Various versions are distinguished based on whether to perform the online updating and re-detection processes. The leftmost version of each sequence is a version that does not perform both the re-detection and update procedures, and it is identical to the existing SiamFC [2] except for the confidence scoring method and specific parameters. We set this version as a baseline and transformed the update process into three ways as follows: always update with the same weights, always update with an exponential forgetting scheme, and update only when confident. The performance of the last version was the best among the three versions, which was notable in the sequences with large appearance changes of the target such as *Nissan*, *Car1*, and *Cat2*.

However, the four versions described above were difficult to detect the target if the tracking object disappeared frequently or for a long period. The main reason for the problem is that the searching range of the Siamese model is limited. Therefore, we applied only the re-detection procedure without updating to verify the performance of re-detection. The performance has greatly improved by the ability to re-detect the target even if the tracker misses the target due to full occlusion and camera motion changes. In the end, the final version further improved performance by adding an online update process and the results were remarkable in *Car16*, *Group2*, *LiverRun*, *Longboard*, *Carchase*, *Person14*, and *skiing* sequences.

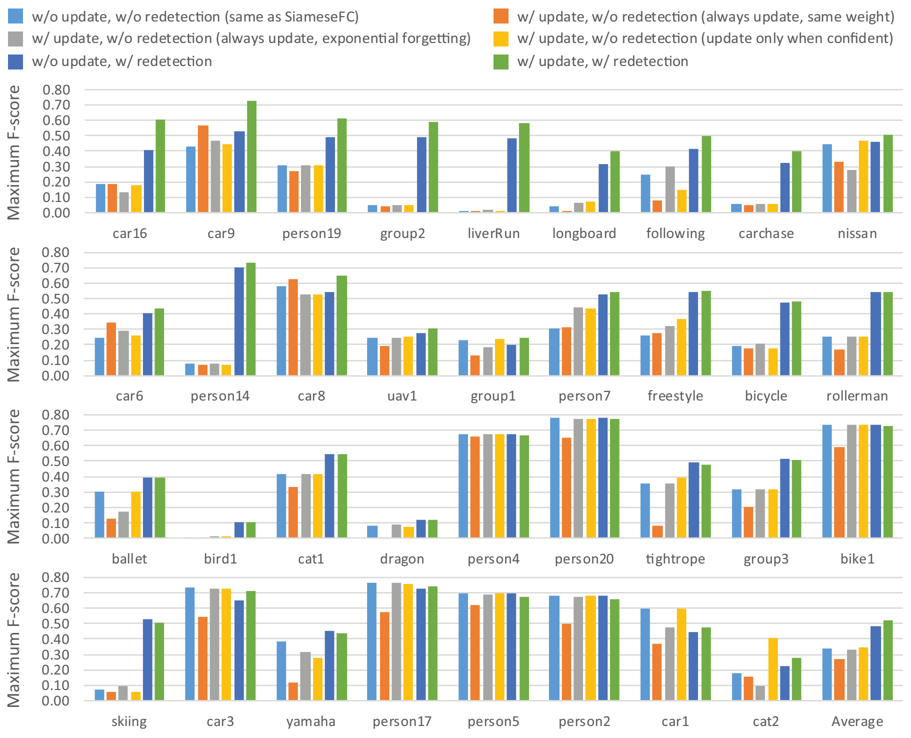

Fig. 3. The maximum F-score for various versions of the proposed MMLT. All sequences are listed based on the order in which the performance of the proposed final version is superior to that of the other versions.

5 Conclusion

In this paper, we have proposed a memory model based on the Siamese network for long-term tracking (MMLT). Memory stores in MMLT are divided into short- and long-term stores depending on each characteristic. The tracking process is operated by taking a weighted sum of the features in both the short- and long-term stores. On the other hand, in the re-detection, only the information in the long-term store is utilized, and the target is detected in the entire image by adopting a coarse-to-fine strategy. As such, memory stores play a crucial role in the tracking and re-detection parts, and the short- and long-term stores are managed differently. The short-term memory is managed in the first in / first out manner, and the forgetting curve is employed for managing the long-term memory. Regional Maximum Activations of Convolutions (R-MAC) is applied to determine the existence of the target and to compute the confidence score. The experimental results on the long-term tracking benchmark show that MMLT achieves state-of-the-art performance.

References

1. Atkinson, R.C., Shiffrin, R.M.: Human memory: A proposed system and its control processes1. In: Psychology of learning and motivation, vol. 2, pp. 89–195. Elsevier (1968)
2. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.: Fully-convolutional siamese networks for object tracking. In: European conference on computer vision. pp. 850–865. Springer (2016)
3. Danelljan, M., Bhat, G., Khan, F.S., Felsberg, M.: Eco: Efficient convolution operators for tracking. In: Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA. pp. 21–26 (2017)
4. Danelljan, M., Häger, G., Khan, F., Felsberg, M.: Accurate scale estimation for robust visual tracking. In: British Machine Vision Conference, Nottingham, September 1-5, 2014. BMVA Press (2014)
5. Danelljan, M., Hager, G., Shahbaz Khan, F., Felsberg, M.: Learning spatially regularized correlation filters for visual tracking. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4310–4318 (2015)
6. Danelljan, M., Shahbaz Khan, F., Felsberg, M., Van de Weijer, J.: Adaptive color attributes for real-time visual tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1090–1097 (2014)
7. Ebbinghaus, H.: Memory: A contribution to experimental psychology. Annals of neurosciences **20**(4), 155 (2013)
8. Fan, H., Ling, H.: Parallel tracking and verifying: A framework for real-time and high accuracy visual tracking. arXiv preprint arXiv:1708.00153 (2017)
9. Felsberg, M., Kristan, M., Matas, J., Leonardis, A., Pflugfelder, R., Häger, G., Berg, A., Eldesokey, A., Ahlberg, J., Čehovin, L., et al.: The thermal infrared visual object tracking vot-tir2016 challenge results. In: European Conference on Computer Vision. pp. 824–849. Springer (2016)
10. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE transactions on pattern analysis and machine intelligence **32**(9), 1627–1645 (2010)
11. Galoogahi, H.K., Fagg, A., Lucey, S.: Learning background-aware correlation filters for visual tracking. In: Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA. pp. 21–26 (2017)
12. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Computer Vision (ICCV), 2017 IEEE International Conference on. pp. 2980–2988. IEEE (2017)
13. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-speed tracking with kernelized correlation filters. IEEE Transactions on Pattern Analysis and Machine Intelligence **37**(3), 583–596 (2015)
14. Hong, Z., Chen, Z., Wang, C., Mei, X., Prokhorov, D., Tao, D.: Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 749–758 (2015)
15. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. arXiv preprint arXiv:1709.01507 (2017)
16. Kalal, Z., Mikolajczyk, K., Matas, J.: Forward-backward error: Automatic detection of tracking failures. In: Pattern recognition (ICPR), 2010 20th international conference on. pp. 2756–2759. IEEE (2010)
17. Kalal, Z., Mikolajczyk, K., Matas, J.: Tracking-learning-detection. IEEE transactions on pattern analysis and machine intelligence **34**(7), 1409–1422 (2012)

18. Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pflugfelder, R., Zajc, L.C., Vojr, T., Hger, G., Lukeic, A., Eldesokey, A., Fernndez, G.: The visual object tracking vot2017 challenge results. In: 2017 IEEE International Conference on Computer Vision Workshops (ICCVW). pp. 1949–1972 (Oct 2017). https://doi.org/10.1109/ICCVW.2017.230
19. Kristan, M., Matas, J., Leonardis, A., Vojir, T., Pflugfelder, R., Fernandez, G., Nebehay, G., Porikli, F., Čehovin, L.: A novel performance evaluation methodology for single-target trackers. IEEE Transactions on Pattern Analysis and Machine Intelligence **38**(11), 2137–2155 (Nov 2016). https://doi.org/10.1109/TPAMI.2016.2516982
20. Liang, P., Blasch, E., Ling, H.: Encoding color information for visual tracking: Algorithms and benchmark. IEEE Transactions on Image Processing **24**(12), 5630–5644 (2015)
21. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International journal of computer vision **60**(2), 91–110 (2004)
22. Lukezic, A., Vojir, T., Zajc, L.C., Matas, J., Kristan, M.: Discriminative correlation filter with channel and spatial reliability. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. vol. 2 (2017)
23. Lukežič, A., Zajc, L.Č., Vojíř, T., Matas, J., Kristan, M.: Fclt-a fully-correlational long-term tracker. arXiv preprint arXiv:1711.09594 (2017)
24. Lukežič, A., Zajc, L.Č., Vojíř, T., Matas, J., Kristan, M.: Now you see me: evaluating performance in long-term visual tracking. arXiv preprint arXiv:1804.07056 (2018)
25. Ma, C., Yang, X., Zhang, C., Yang, M.H.: Long-term correlation tracking. In: Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on. pp. 5388–5396. IEEE (2015)
26. Nam, H., Han, B.: Learning multi-domain convolutional neural networks for visual tracking. In: Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on. pp. 4293–4302. IEEE (2016)
27. Nebehay, G., Pflugfelder, R.: Consensus-based matching and tracking of keypoints for object tracking. In: Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on. pp. 862–869. IEEE (2014)
28. Nebehay, G., Pflugfelder, R.: Clustering of static-adaptive correspondences for deformable object tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2784–2791 (2015)
29. Smeulders, A.W., Chu, D.M., Cucchiara, R., Calderara, S., Dehghan, A., Shah, M.: Visual tracking: An experimental survey. IEEE transactions on pattern analysis and machine intelligence **36**(7), 1442–1468 (2014)
30. Song, Y., Ma, C., Gong, L., Zhang, J., Lau, R.W., Yang, M.H.: Crest: Convolutional residual learning for visual tracking. In: 2017 IEEE International Conference on Computer Vision (ICCV). pp. 2574–2583. IEEE (2017)
31. Tolias, G., Sicre, R., Jégou, H.: Particular object retrieval with integral max-pooling of cnn activations. arXiv preprint arXiv:1511.05879 (2015)
32. Torr, P.H., Zisserman, A.: Mlesac: A new robust estimator with application to estimating image geometry. Computer vision and image understanding **78**(1), 138–156 (2000)
33. Wu, Y., Lim, J., Yang, M.H.: Object tracking benchmark. IEEE Transactions on Pattern Analysis and Machine Intelligence **37**(9), 1834–1848 (2015)
34. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). pp. 2881–2890 (2017)