MART: Motion-Aware Recurrent Neural Network for Robust Visual Tracking

Heng Fan Haibin Ling
Department Computer Science, Stony Brook University, Stony Brook, NY, USA
{hefan,hling}@cs.stonybrook.edu

Abstract

We introduce **MART**, **M***otion-***A***ware* **R***ecurrent neural network (MA-RNN) for* **T***racking, by modeling robust long-term spatial-temporal representation. In particular, we propose a simple, yet effective context-aware displacement attention (CADA) module to capture target motion in videos. By seamlessly integrating CADA into RNN, the proposed MA-RNN can spatially align and aggregate temporal information guided by motion from frame to frame, leading to more effective representation that benefits a tracker from motion when handling occlusion, deformation, viewpoint change etc. Moreover, to deal with scale change, we present a monotonic bounding box regression (mBBR) approach that iteratively predicts regression offsets for target object under the guidance of intersection-over-union (IoU) score, guaranteeing non-decreasing accuracy. In extensive experiments on five benchmarks, including GOT-10k, LaSOT, TC-128, OTB-15 and VOT-19, our tracker* **MART** *consistently achieves state-of-the-art results and runs in real-time.*

1. Introduction

Visual tracking has been one of the most important components in computer vision with many applications such as robotics, surveillance and so on. For a robust tracker, one of the core problems is how to design an effective feature representation for target appearance modeling [56] so that the tracker can well deal with various challenges in videos such as occlusion, deformation, view changes, etc.

Early approaches usually leverage various hand-crafted representation (*e.g.*, pixel value [4], HoG [24]) for tracking. Recently, inspired by powerful deep networks [23, 31], researchers have resorted to deep representation for tracking and achieved significant improvement [2, 8, 17, 18, 32, 33, 41, 42, 55, 57, 62]. Despite considerable advancement, most existing trackers focus on spatial feature representation of current frame for tracking, while leaving rich temporal information under explored. Consequently, their performances may degrade when target feature is corrupted by challenges such as occlusion, deformation, etc.

A natural remedy for this problem is to use both *spatial* and *temporal* representation for appearance modeling. This way, current frame can be effectively enhanced for tracking with extra support from historical frames, even when difficult challenges occur. A recent representative effort in [64] develops such a spatial-temporal representation for tracking. Since target features are usually not spatially aligned between frames due to motion, this method estimates optical flow using a sub-network (FlowNet [14]) to capture motion dynamics of the target for spatial feature alignment. Despite upgrading performance, this method can be improved in two aspects: (1) *Efficiency*. To obtain optical flow, a large extra optical network is integrated into feature representation network, resulting in more computation and inefficient tracking inference. (2) *Long-term representation*. In tracker [64], spatial-temporal representation is achieved by warping and aggregating features on a short fixed temporal window, making it difficult to obtain a long-term representation, which is desired for tracking.

To handle the above issues and obtain an efficient long-term spatial-temporal representation, we propose a novel **M**otion-**A**ware **R**ecurrent neural network (MA-RNN) for **T**racking. Specifically, an MA-RNN consists of two parts, a context-aware displacement attention (CADA) module and an RNN. The RNN component, implemented based on ConvGRU [1], aims at long-term representation by learning to aggregate temporal features. Due to motion dynamics, however, targets are usually spatially *misaligned* across frames (see Fig. 1 for an example). In such situation, direct aggregation of features into RNN may even be detrimental to the representation. Attacking this problem, we propose a *simple*, yet *effective* CADA module for efficient motion capture and apply it to align feature aggregation in RNN for robust representation. In particular, the motion dynamics of a target are captured by modeling *displacement* of each unit on feature maps across frames. To achieve this, we match each unit in current frame to a local region in last frame, and compute a soft attention score map to represent such displacement. For robustness, context of each unit is taken into consideration for matching. After obtaining the displacement attention score map, we propagate it to guide spatial

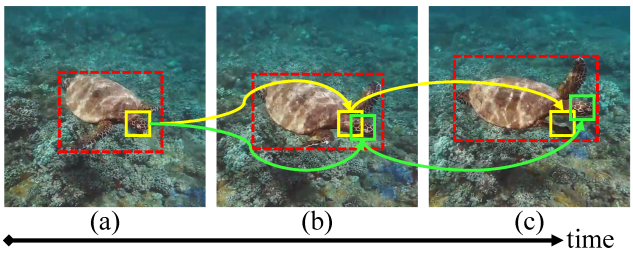


Figure 1. Misalignment caused by target motion. The yellow patches in (b) and (c) have the same spatial positions with yellow patch in (a) while they do not *semantically* align with it. Instead, the green patches in (b) and (c) with slight displacements are *semantically* better aligned with the yellow patch in (a).

alignment of cross-frame features in RNN. By seamlessly integrating CADA into RNN, MA-RNN can model long-term spatial-temporal appearance in feature representation.

Based on the spatial-temporal representation, we apply a classifier to localize target. To handle scale change, we propose a separate scale estimation component using multi-step bounding box regression (BBR). The core is a target-aware BBR-IoU network that, given a reference target, predicts the offsets of its bounding box and IoU score for regression result. The IoU score is used to guide iterative bounding box regression. Since the IoU score of the candidate box is guaranteed to be non-decreasing, we call our method monotonic bounding box regression (mBBR). As a single step of regression usually works well (as observed in recent studies as well [32, 33]), mBBR works efficiently most of time, with very few iterations by using IoU score as an early-stop condition.

To evaluate our tracker, we conduct experiments on five popular benchmarks including GOT-10k [25], LaSOT [16], TC-128 [38], OTB-15 [59] and VOT-19 [30]. The proposed MART consistently achieves state-of-the-art results[1].

In summary, the contributions in this paper include: **(i)** a novel motion-aware recurrent neural network (MA-RNN) for spatial-temporal representation for tracking; **(ii)** a novel context-aware displacement attention (CADA) module for target motion dynamic capture and spatial feature alignment; **(iii)** a simple yet effective monotonic bounding box regression (mBBR) for accurate target scale estimation; and **(iv)** state-of-the-art performances on five benchmarks.

2. Related Work

Visual tracking has been extensively studied in recent years. In the following we discuss the most related work and refer readers to [36, 37, 47] for recent tracking surveys.

Deep Tracking. Inspired by success in image classification [23, 31], deep feature has been used for tracking and demonstrated state-of-the-art performance [42, 55, 57]. One of recent trends is to integrate correlation filter tracking with deep feature to learn a discriminative classification model [9, 41, 44, 48, 49, 64]. Especially, the work of [64] leverages temporal information for tracking. Despite robustness, these trackers cannot well deal with heavy scale variation. To handle this issue, the work of [8] decomposes tracking into localization and estimation subtasks and adopts IoU prediction [27] for scale estimation, achieving significant improvement. Another trend is to learn a similarity measurement for tracking using Siamese networks [2, 50]. Owing to balanced accuracy and efficiency, the work of [2] has been improved with many extensions [21, 22, 33, 58]. Notably, by integrating with region proposal network (RPN) [45], the work of [33] greatly improves Siamese tracker in dealing with scale change, which is further enhanced by incorporating distractor detection [63], multi-stage mechanism [18, 54] and deeper architecture [32, 62].

Our work is related to but different from [64] that uses a large FlowNet [14] for motion capture. In contrast, we propose an efficient CADA module for motion dynamics. Besides, we model a long-term representation using RNNs, which differs from [64] with a short-term representation. Our work is relevant to [8] by sharing the similar spirit of decomposing tracking into two sub-problems. However, our solutions to the two sub-problems are very different than those in [8]. In addition, IoU prediction [27] is employed in both [12] and our work, but in different ways. The work of [8] directly utilizes IoU score for scale estimation, while our approach uses it to guide bounding box regression. Different from the multi-step BBR in [18, 54], ours is more adaptive by using IoU score to guide BBR.

An interesting observation is that, our CADA module can be connected to the recent Siamese tracker [2] that learns a similarity measurement for tracking. The difference is, the displacement, obtained by matching, is directly used at *box-level* for target localization in [2], while we use it at *pixel-level* to capture motion dynamics and guide spatial feature alignment. Besides, our CADA module leverages contextual information for robust matching.

RNN for Spatial-temporal Representation. RNN has been extensively explored in computer vision owing to its capacity in modeling long-term feature representation including video object segmentation [51], video action classification/recognition [1, 15], video object detection [40, 60].

Our approach is closely related to [1] that proposes convolutional gated recurrent unit networks (ConvGRU) and then applies it for action recognition. Implemented based on ConvGRU, our work MA-RNN is different than ConvGRU in two main aspects. First, MA-RNN aims at distinguishing target/background in videos instead of classifying a video clip. Second, MA-RNN uses a motion-aware mod-

[1]The code will be available at `https://hengfan2010.github.io/projects/MART/MART.htm`

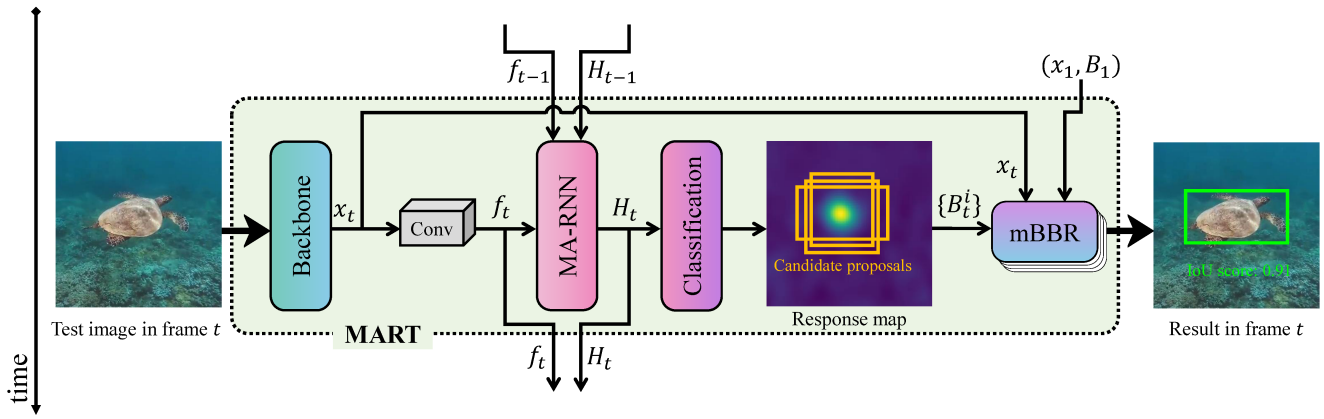


Figure 2. The MART tracking pipeline, including *target localization* and *scale estimation*. The localization branch, based on our MA-RNNs, provides target position to generate candidate proposals, which are sent to mBBR for scale estimation.

ule to capture dynamics for robust representation in RNN, which is important for visual tracking.

Motion Dynamic Modeling. As a crucial component in video understanding, how to capture and model motion dynamics attracts increasing attention. A common solution is to compute optical flow between frames. Inspired by deep learning, the accuracy of optical flow has been greatly improved (*e.g.*, FlowNet/FlowNet 2.0 [14, 26]). Despite this, it is time-consuming to obtain such motion information with optical flow when considering efficiency requirement in tracking. An alternative solution for motion modeling is to explicitly compute displacement of pixels by performing correlation [19]. Unlike [19] that aims at detecting object movements for video object detection, we utilize CADA to align features in RNNs for tracking. Besides, we take into consideration context information of each unit to better capture motion dynamics, which significantly differs from [19].

3. Motion-Aware RNN for Tracking (MART)

3.1. Overview

Following paradigm in [8], we decompose tracking into two exclusive sub-tasks, including target localization and target scale estimation. Fig. 2 illustrates the overview of our tracking algorithm.

In target localization, for a test image, we employ a backbone network (*i.e.*, ResNet [23] pre-trained on ImageNet [13]) to extract initial feature representation x_t. Considering the gap between classification and tracking tasks, we apply an extra conv layer to transform the initial feature x_t to f_t. Together with spatial feature f_{t-1} and spatial-temporal representation H_{t-1} from last frame, f_t is fed to MA-RNN for spatial-temporal representation H_t in current frame. Classification is performed on H_t to obtain the target position.

Target scale estimation is performed using mBBR. With the target position by localization component, we sample a set of candidate proposals $\{B_c^i\}$ around target position using previous scale information. These proposals $\{B_c^i\}$ and the test image feature x_t, together with initial bounding box B_1 and reference image feature x_1, are sent to mBBR for fianl target scale estimation.

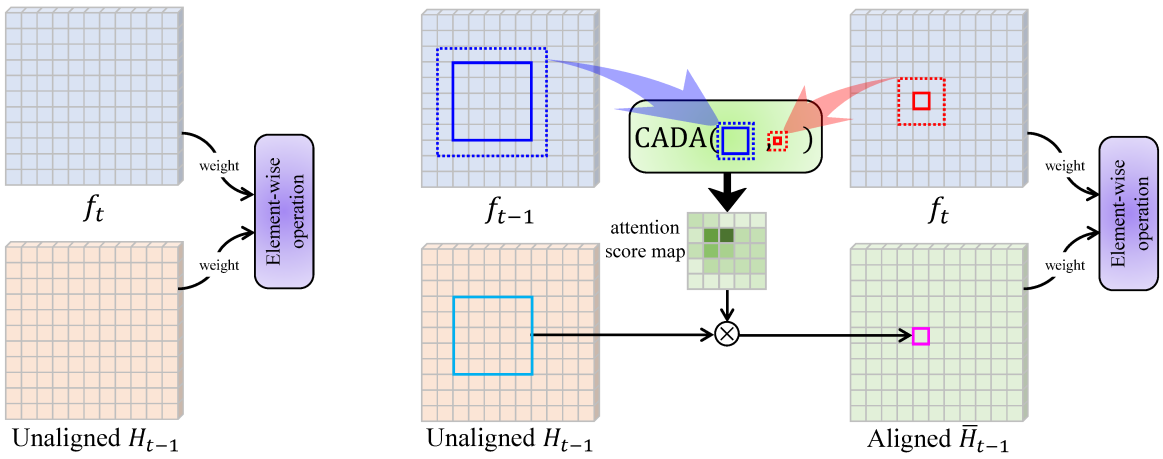
3.2. Motion-Aware RNN (MA-RNN)

In this work, we aim at learning a robust long-term spatial-temporal representation for object tracking. For this purpose, a natural choice is RNN that captures long-term representation by aggregating temporal information from frame to frame. Considering the importance of spatial information in 2D images/videos, the work of [1] replaces linear product operation in GRU-RNN [6] with convolution and proposes ConvGRU for action recognition, achieving superior performance. Mathematically, ConvGRU can be formulated as follows,

$$\begin{aligned} z_t &= \phi(W_{fz} * f_t + U_{fz} * H_{t-1} + b_z) \\ r_t &= \phi(W_{fr} * f_t + U_{fr} * H_{t-1} + b_r) \\ \widetilde{H}_t &= \varphi(W_{f\widetilde{H}} * f_t + U_{f\widetilde{H}} * (r_t \circ H_{t-1}) + b_{\widetilde{H}}) \\ H_t &= z_t \circ \widetilde{H}_t + (1 - z_t) \circ H_{t-1} \end{aligned} \tag{1}$$

where f_t and H_{t-1} are feature input to ConvGRU and spatial-temporal representation in last frame, W_{fz}, U_{fz}, W_{fr}, U_{fr}, $W_{f\widetilde{H}}$, $U_{f\widetilde{H}}$ represent convolutional kernels and are end-to-end learned, z_t and r_t are update and reset gates, and H_t denotes spatial-temporal representation in current frame. '$*$' and '$\circ$' are convolution operation and Hadamard product, respectively. The bias terms in Eq. (1) are omitted for simplicity.

From Eq. (1), we observe that the temporal information is derived through direct aggregation between features f_t and H_{t-1} (see Fig. 3 (a)), which ignores the misalignment problem caused by motion dynamics in videos (see Fig. 2



(a) Feature aggregation in existing RNN. (b) Feature aggregation in MA-RNN.

Figure 3. Comparison between feature aggregation in existing RNN (*e.g.*, ConvGRU [1]) and our MA-RNN. Instead of directly aggregating features, we leverage CADA module to align and guide feature aggregation for better spatial-temporal representation.

again). To deal with this issue, we propose the MA-RNN with a key innovative CADA module, which efficiently captures motion flow to guide temporal feature aggregation over frames. The idea is to compute the displacement of a unit (on feature map) to capture motion flow. Since target object usually moves smoothly or slowly in videos, we compute the displacement of a unit within a local region. In specific, for a feature unit $f_t(p,q)$ at position (p,q) on feature map f_t, the CADA module matches $f_t(p,q)$ and each feature unit within a small region centered at (p,q) on feature map f_{t-1}. The attentional matching score map, denoted by S, is then utilized to transform the spatial-temporal representation H_{t-1} to align with feature f_t. Mathematically, the matching process by CADA can be formulated as follows,

$$S_{p,q}(i,j) = \frac{\mathrm{sim}\big(f_t(p,q), f_{t-1}(p+i,q+j)\big)}{\sum_{i,j\in\{-d,\cdots,d\}} \mathrm{sim}\big(f_t(p,q), f_{t-1}(p+i,q+j)\big)} \tag{2}$$

where $S_{p,q}(i,j)$ denotes the normalized similarity score for feature unit $f_{t-1}(p+i,q+j)$ on f_{t-1}, d controls the size of local region, and $\mathrm{sim}(\cdot,\cdot)$ computes the similarity between two feature units, which can be defined by dot product for its simplicity, *i.e.*, $\mathrm{sim}\big(f_t(p,q), f_{t-1}(p+i,q+j)\big) = f_t(p,q) \bullet f_{t-1}(p+i,q+j)$.

For the same units of target between different frames, their contexts are similar to each other. Therefore, we introduce contextual information of each unit into similarity computation. For a unit $f_t(p,q)$, its context region feature $C_t(p,q)$ is defined as a set of unit features for a local region centered at (p,q) with size k (excluding itself) as follows (also see the dashed red rectangle in Fig. 3 (b)),

$$C_t(p,q) = \{f_t(p+i,q+j)\ \forall\, i,j \in \{-k,\cdots,k\}\}/\{f_t(p,q)\} \tag{3}$$

Considering deformation of target, we apply max pooling operation to $C_t(p,q)$, and obtain the contextual feature unit $f_t^C(p,q)$ as

$$f_t^C(p,q) = \mathrm{maxpool}(C_t(p,q)) \tag{4}$$

Therefore, we re-write $\mathrm{sim}(\cdot,\cdot)$ as containing two weighted terms,

$$\begin{aligned}\mathrm{sim}\big(f_t(p,q), f_{t-1}(p+i,q+j)\big) &= \alpha f_t(p,q) \bullet f_{t-1}(p+i,q+j) \\ &+ (1-\alpha) f_t^C(p,q) \bullet f_{t-1}^C(p+i,q+i)\end{aligned} \tag{5}$$

where the α adjusts the importance of contextual information.

Note that, our CADA module shares the similar spirit with recent Siamese tracker [2] that computes a soft attentional matching score between a template and a search region for tracking. Different from [2], however, we utilize CADA module to capture motion flow and align spatial features in RNN. Besides, we consider contextual information in CADA for more robust matching.

With Eq. (2), MA-RNN is obtained using attention score $S_{p,q}(i,j)$ to align spatial features in RNN, and mathematically expressed as followed as,

$$\begin{aligned}\bar{H}_{t-1}(p,q) &= \sum_{i,j\in\{-d,\cdots,d\}} \big(H_{t-1}(p+i,q+j) S_{p,q}(i,j)\big) \\ z_t &= \phi(W_{fz} * f_t + U_{fz} * \bar{H}_{t-1}) \\ r_t &= \phi(W_{fr} * f_t + U_{fr} * \bar{H}_{t-1}) \\ \widetilde{H}_t &= \varphi(W_{f\widetilde{H}} * f_t + U_{f\widetilde{H}} * (r_t \circ \bar{H}_{t-1})) \\ H_t &= z_t \circ \widetilde{H}_t + (1-z_t) \circ \bar{H}_{t-1}\end{aligned} \tag{6}$$

This way, MA-RNN is able to learn a robust long-term spatial-temporal representation. Fig. 3 (b) illustrates the spatial feature alignment in RNN by CADA module. It is worth noticing that, our CADA module is computed within 3ms, which is much more efficient than optical flow.

3.3. Target Localization with MA-RNN

With long-term spatial-temporal representation by MA-RNN, we perform classification for target localization. To adapt to target appearance change, the classifier requires online update. Inspired by the simplicity and efficiency of classifier in [8], we adopt the same classification network that consists of two convolutional layers and is defined as follows,

$$\psi(H;\{w_2,w_1\}) = \sigma(w_2 * \delta(w_1 * H)) \tag{7}$$

where H is the spatial-temporal feature representation, $\{w_2, w_1\}$ denote convolutional kernels, $\sigma(\cdot)$ and $\delta(\cdot)$ are parametric exponential linear unit (PELU) [52] and identity activation functions.

Inspired by discriminative correlation filter trackers [9, 10,41,48], the classification model is learned by minimizing the square loss, similar to [8], as follows,

$$\mathcal{L}_{\text{cls}} = \sum_{i=1}^{n} \gamma_i \|\psi(H_i;\{w_2,w_1\}) - y_i\|^2 + \sum_j \lambda_j \|w_j\|^2 \tag{8}$$

where y_i denotes a 2D Gaussian label, γ_i represents importance of each training sample, and λ_j the regularization parameter.

It is worth noting that, different from [8] that adopts feature representation (*i.e.*, x_t in Fig. 2) extracted from pre-trained ResNet [23], our classification model is built on long-term spatial-temporal representation (*i.e.*, H_t in Fig. 2) by MA-RNN. Considering the gap between classification and tracking tasks, we apply an extra conv layer, followed by a ResNet backbone, for feature transformation before feeding it to MA-RNN (see Fig. 2). During training, the additional conv layer, MA-RNN and classification network are jointly end-to-end trained using the ADAM optimizer [28]. In inference phase, only the classification network is updated. For efficiency, we utilize strategy in [8] for online learning of classifier. Please refer to [8] for more details.

3.4. Target Scale Estimation with m-BBR

While providing target position, the localization component cannot well estimate target scale. Inspired by detection community (*e.g.*, [27,45]), recent trackers use either one or multi-stage BBR (*e.g.*, [18, 32, 33, 54, 62, 63]) or IoUNet (*e.g.*, [3, 8]) to estimate target scale. The former method is efficient by predicting offsets within one forward pass while lack of localization confidence reasoning, which may results in non-monotonic regression problem [27]. The latter one is reliable, however, requires both forward pass w.r.t. computing IoU score and backward pass w.r.t. computing gradients for box update. In addition, the IoU increment is relatively small compared to regression. Thus, it often needs multiple iterations for final scale estimation.

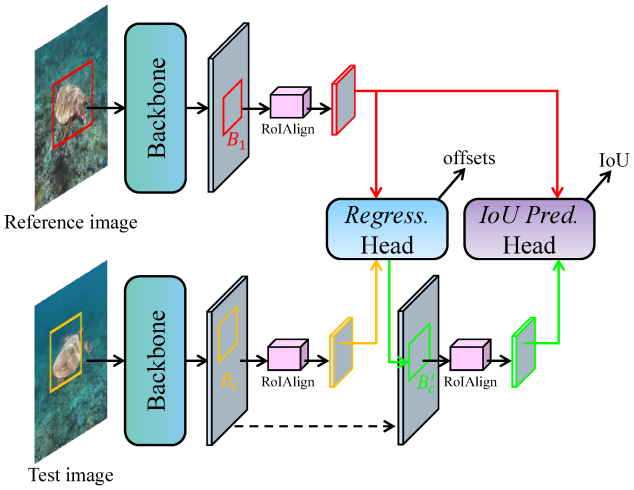


Figure 4. Illustration of BBR-IOU network that regresses target bounding box and performs IoU prediction.

Taking advantages of both the aforementioned approaches, we introduce a multi-step iterative method, *i.e.*, monotonic bounding box regression (mBBR), for target scale estimation. The core of mBBR is a BBR-IoU network that first predicts regression offsets for a box and then perform IoU prediction [27] for the regression result. To implement BBR-IoU network, we employ the similar feature modulation strategy in [8]. Fig. 4 shows the pipeline of BBR-IoU network, and due to space limited, we refer readers to supplementary material for its detailed architecture.

As demonstrated in Fig. 4, BBR-IoU takes as inputs the reference image I_1, the initial bounding box B_1, the test image I_T and a candidate bounding box B_c, and outputs a 4d regression offset vector (r_1, r_2, r_3, r_4) and IoU score O_c for regressed box $B_c^{'}$ ($B_c^{'}$ is obtained by applying regression offsets to B_c). To train the BBR-IoU network, we use the follow loss function,

$$\mathcal{L}_{\text{est}} = \sum_{i=\{1,2,3,4\}} \mathcal{L}_{\text{smooth}}(r_i, r_i^*) + \mathcal{L}_{\text{mse}}(O_c, O_c^*) \tag{9}$$

where $(r_1^*, r_2^*, r_3^*, r_4^*)$ are the regression labels of box B_c and O_c^* denotes the IoU between $B_c^{'}$ and groundtruth target bounding box in the test image. $\mathcal{L}_{\text{smooth}}$ is smooth-L_1 loss [20] and $\mathcal{L}_{\text{mse}}$ the mean square loss.

Once training completed, we utilize BBR-IoU network to iteratively estimate target scale under the guidance of IoU. If IoU score is above an acceptable threshold θ_{IoU}, we directly output regression for scale estimation; if it is decreased compared to last iteration, we stop regression and output offsets of last iteration for scale estimation. Otherwise, regression iterates over new box until reaching to the maximum number N_{reg} of regressions. Owing to the guidance by IoU score, mBBR guarantees that the regression accuracy is monotonically increasing. Algorithm 1 summa-

Algorithm 1: Monotonic Bounding Box Regression

Input: I_1, B_1, I_T, B_c, θ_{IoU}, N_{reg}, trained model BBR-IoU;
Output: Final refined target bounding box B;
$B_c^1 = B_c$, $B = B_c$;
for $i = 1$ to N_{reg} **do**
  /*regression and IoU prediction*/
  $(r_1^i, r_2^i, r_3^i, r_4^i), O_c^i \leftarrow$ BBR-IOU(I_1, B_1, I_T, B_c^i);
  $B_c^{i+1} \leftarrow$ Regressing B_c^i using $(r_1^i, r_2^i, r_3^i, r_4^i)$;
  $B \leftarrow B_c^{i+1}$;
  if $O_c^i \geq \theta_{\text{IoU}}$ **then**
    break; /*IoU score as early-stop condition*/
  else if $O_c^i < O_c^{i-1}$ **then**
    /*recover B using last regression result*/
    $B \leftarrow B_c^i$;
    break;
  end
end

rizes the working pipeline of mBBR.

Note that, our mBBR significantly differs from [18, 54] with multi-step BBR and [8] with IoU prediction. The methods of [18, 54] perform a fix number of steps of regression for scale estimation. In addition, localization confidence is ignored in each step, which may result in non-monotonic regression issue [27]. By contrast, we leverage IoU score to guide each step of regression and use it as an early-stop condition. By doing so, we can achieve adaptive regression with monotonically increasing accuracy. In [8], a candidate box is refined within two stages: computing IoU score in forward pass and adjusting candidate box using gradient descent method in backward pass. These two stages are iteratively repeated for a fixed time. Unlike [8], our approach directly predicts offsets for scale estimation. Since a single step of regression works well in most cases, our mBBR stops within very few iterations most of time, which is much more efficient.

3.5. Training and Tracking

Training. We train the target localization and estimation parts separately. For the target localization component, the transformation conv layer, MA-RNNs and classification are end-to-end trained using ADAM method [28] on videos. To reduce redundancy, we sparsely sample a frame with a random interval $\in [5, 15]$ to form a new sequence for training. For each frame, we sample a square patch with an area of about 5×5 times the target and randomly shift it. These image patches are resized to a fixed size and then sent for training. For scale estimation, BBR-IoU network is trained on image pairs with each consisting of a reference image patch and a test image patch. These image pairs are sampled from the same video with a maximum gap of 50 frames. For reference image, we sample a square patch with an area of about 5×5 times centered at the target. We sample a similar image patch for test image, with perturbations in position and scale as in [8]. The reference and test image patches are resized to the same resolution before training. For each image pair, we randomly generate 16 candidate boxes in the test image and ensure each one with a minimum 0.6 IoU with the groundtruth bounding box. Data augmentation strategies, such as image flipping, rotation and color jittering, are adopted.

Visual Tracking. We split tracking into target localization and target scale estimation. For each sequence, we precompute the feature for reference image in target scale estimation. When a new frame arrives, we extract a region of interest based on tracking result of last frame and calculate its spatial-temporal feature representation, which is fed to the classification network for predicting target position. For robustness, we sample a set of M initial target proposals around the target position, and apply mBBR for scale estimation. The final target scale is determined by the refined box with the maximum IoU score. To adapt to target appearance changes, we collect, every V frames, historical spatial-temporal representations from up to U frames for online updating of the classification model. In order to facilitate update, we utilize the strategy in [8] for online learning during tracking. Note that, MA-RNN does not need online update as they have learned to model generic spatial-temporal representation for our task.

4. Experiments

Implementation. We implement MART in python using PyTorch [43] on an Nvidia GTX-1080 GPU. We employ pre-trained ResNet-50 [23] as our backbone network and freeze the parameters in both training and tracking. Our MART runs at a speed of around 31 *fps*. We train the target localization branch using training splits of LaSOT [16], GOT-10k [25] and VID [46]. We train for 50 epochs using ADAM [28]. The learning rate starts from 10^{-3} with a decay of 0.1 every 10 epochs. The channel of spatial-temporal representation H_t is 256. The time step for training MA-RNNs is empirically set to 10. The d, k and α are set to 5, 1 and 0.8, respectively. For BBR-IoU network, we use training splits of LaSOT [16], GOT-10k [25] and VID [46] and COCO [39]. We train for 50 epochs with ADAM [28] using learning rate of 10^{-3} with a decay of 0.1 every 10 epochs. The IoU threshold θ_{IoU} is set to 0.85. The maximum number N_{reg} of iterations is 3. The V and U are set to 20 and 50, and the number M of initial proposals for scale estimation is set to 5.

Table 1. Comparisons on GOT-10k [25]. The best two scores are highlighted in **red** and **blue** colors, respectively.

Tracker	AO	$SR_{0.50}$	$SR_{0.75}$
ECO-HC [9]	0.286	0.276	0.096
CFNet [53]	0.293	0.265	0.087
MDNet [42]	0.299	0.303	0.099
HCF [41]	0.315	0.297	0.088
ECO [9]	0.316	0.309	0.111
SiamFC [2]	0.348	0.353	0.098
SPM [54]	0.513	0.593	0.359
ATOM [8]	0.556	0.634	0.402
DiMP-50 [3]	**0.611**	**0.717**	**0.492**
MART	**0.628**	**0.732**	**0.504**

4.1. Experiment on GOT-10k

GOT-10k [25] is proposed to assess short-term tracking performance. We evaluate MART on the server provided by the organizers using the testing split with 180 sequences. The performance is measured using average overlap (AO) and success rate (SR) with different thresholds 0.5 and 0.75. Tab. 1 demonstrates the comparisons to other trackers, showing that MART achieves the best performance under all metrics. Specifically, MART obtains AO of 0.628, $SR_{0.50}$ of 0.732 and $SR_{0.75}$ of 50.4, outperforming the second best tracker DiMP [3] with AO of 0.611, $SR_{0.50}$ of 0.717 and $SR_{0.75}$ of 0.492 by 1.7%, 1.5% and 1.2%, respectively. In comparison with ATOM [8] with 0.556 AO, 0.634 $SR_{0.50}$ and 0.403 $SR_{0.75}$, we obtain significant gains by 7.2%, 9.8% and 10.2%, showing the advance of spatial-temporal representation by our MA-RNN.

4.2. Experiment on LaSOT

LaSOT [16] is a large-scale dataset consisting of 1,400 sequences. Following the protocol, we utilize 1,120 sequences for training and the rest 280 for testing. We compare MART with 12 state-of-the-art tracking algorithms (DiMP [3], ATOM [8], SiamRPN++ [32], C-RPN [18], SiamDW [62], MDNet [42], SiamFC [2], StructSiam [61], DSiam [21], ECO [9], STRCF [34] and TRACA [7]).

The results, evaluated using *success* plot, are demonstrated in Fig 5 (a). We observe that, our MART achieves the best performance with 0.571 success score, outperforming the second best DiMP [3] by 0.3%. In comparison to ATOM [8] with 0.523 success score, we obtain considerable gains by 4.8%. Our MART outperforms SiamRPN++ [32] by 7.5% in success plot. In addition, compared to multi-step regression approach for tracking in C-RPN [18] with 0.455 success score, our tracker reveals clear improvements.

4.3. Experiment on TC-128

TC-128 [38] consists of 128 fully annotation colorful videos. Following [38], we employ *success* plot in *one-pass evaluation* (OPE) for evaluation. We compare our MART to 9 state-of-the-art trackers (DiMP [3], SiamRPN++ [32], ATOM [8], SiamFC [2], ECO [9], C-COT [12], PTAV [17], DeepSRDCF [11] and HCF [41]), and the results are shown in Fig. 5 (b). From Fig. 5 (b), we observe that, our MART achieves the best result with success score of 0.621, outperforming the second best DiMP [3] with 0.609 success score by 1.2%. In comparison with ATOM that applies only spatial feature for tracking and achieves 0.590 success score, our approach obtains significant gains of 3.1%, showing the advantages of spatial-temporal representation in robust localization.

4.4. Experiment on OTB-2015

OTB-2015 [59] contains 100 fully annotated sequences. We employ *success* plot metric in OPE to assess different algorithms. We compare our proposed MART to 12 state-of-the-art trackers (DiMP [3], SiamRPN++ [32], ATOM [8], C-RPN [18], DaSiamRPN [63], SiamRPN [33], GradNet [35], SA-Siam [22], ACT [5], SiamFC [2], ECO-HC [9] and TRACA [7]), and the results are shown in Fig. 5 (c).

We can see from Fig. 5 (c), our approach achieves competitive result with success score of 0.678 compared to SiamRPN++ [32] and DiMP [3]. In comparison with ATOM that applies only spatial feature for tracking and achieves 0.655 success score, our approach obtains significant gains of 2.3%, showing the advantages of spatial-temporal representation in robust localization. C-RPN [18] proposes a cascade architecture that employs multi-step regressions for scale estimation, and obtains 0.663 success score. Different from [18], our multi-step regressions are guided by the IoU score, and outperform C-RPN [18] by 2.5% in terms of precision and success plots.

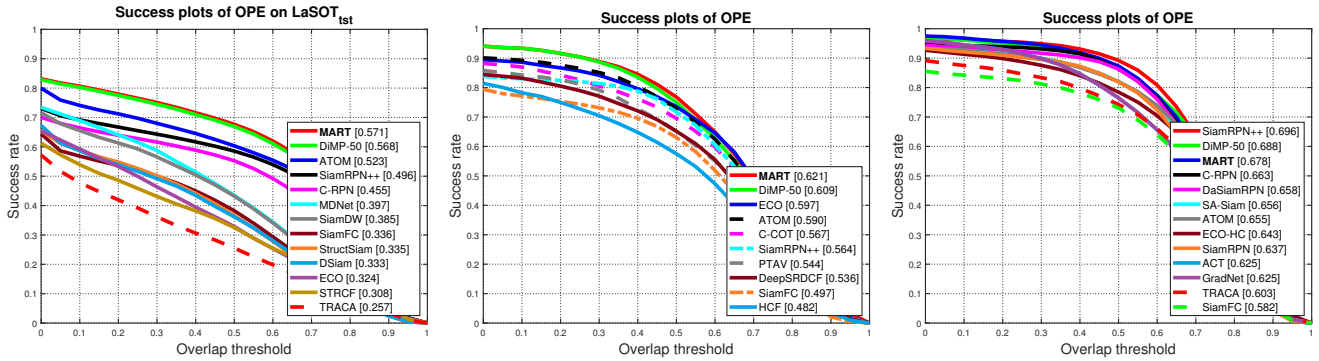
4.5. Experiment on VOT-2019

VOT-2019 [30] contains 60 sequences which are developed by replacing 12 less representative videos in VOT-2018 [29] with more challenging ones. Similar to VOT-2018, each tracker is evaluated with EAO, accuracy and robustness. We compare our MART with several recent top-performance trackers from VOT-2019, and Tab. 2 demonstrates the results. DiMP [3] achieves the best EAO of 0.379. Our MART obtains promising result with EAO of 0.356, which significantly outperforms ATOM [8] with EAO of 0.292 and SiamRPN++ [32] with EAO of 0.285.

4.6. Ablation Experiment

To validate the effect of different components, we conduct ablation experiments on LaSOT_{tst} [16] regarding the target localization and target estimation.

Tab. 3 shows the ablation experiments on target localization. Without any temporal information aggregation, the baseline tracker achieves 0.531 success (SUC) score. When



(a) Comparisons on LaSOT (b) Comparisons on TC-128 (c) Comparisons on OTB-2015

Figure 5. Comparisons of our MART and other state-of-the-art trackers on LaSOT [16], TC-128 [38] and OTB-2015 [59]. Our method achieves the best results on LaSOT and TC-128 and performs favorably against many trackers on OTB-2015.

Table 2. Comparisons on VOT-2019 [30]. The best two scores are highlighted in **red** and **blue** colors, respectively.

Tracker	EAO	Accuracy	Robustness
DCFST [30]	0.361	0.589	0.321
SiamCRF [30]	0.330	**0.625**	**0.296**
SPM [54]	0.275	0.577	0.507
SiamRPN++ [32]	0.285	0.599	0.482
SiamDW [62]	0.299	0.600	0.467
ATOM [8]	0.292	0.603	0.411
DiMP-50 [3]	**0.379**	0.594	**0.278**
MART	**0.356**	**0.607**	0.362

adding RNN (*i.e.*, ConvGRU [1]) for temporal representation, the performance is improved to 0.539, showing the advantage of leveraging temporal cue for tracking. To effectively enhance spatial-temporal representation, we propose to apply a matching based displacement attention module to capture motion for feature alignment in RNN, and push the SUC score to 0.565 with significant gain of 2.5%, which suggests the importance of feature alignment for robust representation. By incorporating contextual information, the SUC score is further improved by 0.6% from 0.565 to 0.571. The ablative experiments in Tab. 3 clearly evidence the effectiveness of our MA-RNN.

Tab. 4 demonstrates the results of our algorithm with different strategies for target scale estimation. As shown in Tab. 4, with one step of bounding box regression, we achieve 0.546 SUC score. When directly adding more regressions, the SUC score is improved to 0.567 while the speed is decreased from 42 *fps* to 26 *fps*. Unlike direct use of multi-step regression, we develop an adaptive regression approach under the guidance of IoU prediction, and achieve better performance with 0.571 SCU score and faster speed with 31 *fps*. Besides, we compare our mBBR with IoU prediction in [8] that estimates scale via iterative forward and backward passes. When replacing mBBR with IoU prediction network, we observe slight gain of 0.3% in SUC score

Table 3. Ablation study on target localization with spatial-temporal representation.

Component	MART			
RNN		✓	✓	✓
Displacement Attention			✓	✓
Contextual Information				✓
SUC (%)	53.1	53.9	56.5	57.1

Table 4. Ablation study on target scale estimation with different strategies.

	IoU network [8]	mBBR		
		One step	wo / IoU	w / IoU
SUC (%)	57.4	54.6	56.7	57.1
Speed (*fps*)	23	42	26	31

from 0.571 to 0.574 while significant speed decrease from 31 *fps* to 23 *fps*, showing more balanced performance of our method.

5. Conclusion

In this paper, we explore spatial-temporal representation for visual tracking. In specific, we introduce novel motion-aware recurrent neural networks to simultaneously capture motion dynamics of target and align spatial temporal features, leading to more effective representation. Based on spatial-temporal representation, we develop an online classification model for target localization. In addition, for target scale estimation, we introduce a monotonic multi-step regression approach that utilizes the IoU prediction score to guide bounding box regression in each step. Integrating the target localization and scale estimation, our tracker MART achieves state-of-the-art results on five benchmark and runs in real-time.

Acknowledgment. This work is supported in part by NSF Grants IIS-2006665 and IIS-1814745.

References

[1] Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. Delving deeper into convolutional networks for learning video representations. In *ICLR*, 2016. 1, 2, 3, 4, 8

[2] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *ECCVW*, 2016. 1, 2, 4, 7

[3] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *ICCV*, 2019. 5, 7, 8

[4] David S Bolme, J Ross Beveridge, Bruce A Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *CVPR*, 2010. 1

[5] Boyu Chen, Dong Wang, Peixia Li, Shuang Wang, and Huchuan Lu. Real-time'actor-critic'tracking. In *ECCV*, 2018. 7

[6] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *EMNLP*, 2014. 3

[7] Jongwon Choi, Hyung Jin Chang, Tobias Fischer, Sangdoo Yun, Kyuewang Lee, Jiyeoup Jeong, Yiannis Demiris, and Jin Young Choi. Context-aware deep feature compression for high-speed visual tracking. In *CVPR*, 2018. 7

[8] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Atom: Accurate tracking by overlap maximization. In *CVPR*, 2019. 1, 2, 3, 5, 6, 7, 8

[9] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Eco: Efficient convolution operators for tracking. In *CVPR*, 2017. 2, 5, 7

[10] Martin Danelljan, Gustav Häger, Fahad Khan, and Michael Felsberg. Accurate scale estimation for robust visual tracking. In *BMVC*, 2014. 5

[11] Martin Danelljan, Gustav Hager, Fahad Shahbaz Khan, and Michael Felsberg. Learning spatially regularized correlation filters for visual tracking. In *ICCV*, 2015. 7

[12] Martin Danelljan, Andreas Robinson, Fahad Shahbaz Khan, and Michael Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *ECCV*, 2016. 2, 7

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 3

[14] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *ICCV*, 2015. 1, 2, 3

[15] Yong Du, Wei Wang, and Liang Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *CVPR*, 2015. 2

[16] Heng Fan, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Hexin Bai, Yong Xu, Chunyuan Liao, and Haibin Ling. Lasot: A high-quality benchmark for large-scale single object tracking. In *CVPR*, 2019. 2, 6, 7, 8

[17] Heng Fan and Haibin Ling. Parallel tracking and verifying: A framework for real-time and high accuracy visual tracking. In *ICCV*, 2017. 1, 7

[18] Heng Fan and Haibin Ling. Siamese cascaded region proposal networks for real-time visual tracking. In *CVPR*, 2019. 1, 2, 5, 6, 7

[19] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to track and track to detect. In *ICCV*, 2017. 3

[20] Ross Girshick. Fast r-cnn. In *ICCV*, 2015. 5

[21] Qing Guo, Wei Feng, Ce Zhou, Rui Huang, Liang Wan, and Song Wang. Learning dynamic siamese network for visual object tracking. In *ICCV*, 2017. 2, 7

[22] Anfeng He, Chong Luo, Xinmei Tian, and Wenjun Zeng. A twofold siamese network for real-time object tracking. In *CVPR*, 2018. 2, 7

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 2, 3, 5, 6

[24] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *TPAMI*, 37(3):583–596, 2014. 1

[25] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *TPAMI*, 2019. 2, 6, 7

[26] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, 2017. 3

[27] Borui Jiang, Ruixuan Luo, Jiayuan Mao, Tete Xiao, and Yuning Jiang. Acquisition of localization confidence for accurate object detection. In *ECCV*, 2018. 2, 5, 6

[28] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2014. 5, 6

[29] Matej Kristan and et al. The visual object tracking vot2018 challenge results. In *ECCVW*, 2018. 7

[30] Matej Kristan and et al. The visual object tracking vot2019 challenge results. In *ICCVW*, 2019. 2, 7, 8

[31] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1, 2

[32] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *CVPR*, 2019. 1, 2, 5, 7, 8

[33] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In *CVPR*, 2018. 1, 2, 5, 7

[34] Feng Li, Cheng Tian, Wangmeng Zuo, Lei Zhang, and Ming-Hsuan Yang. Learning spatial-temporal regularized correlation filters for visual tracking. In *CVPR*, 2018. 7

[35] Peixia Li, Boyu Chen, Wanli Ouyang, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Gradnet: Gradient-guided network for visual object tracking. In *ICCV*, 2019. 7

[36] Peixia Li, Dong Wang, Lijun Wang, and Huchuan Lu. Deep visual tracking: Review and experimental comparison. *PR*, 76:323–338, 2018. 2

[37] Xi Li, Weiming Hu, Chunhua Shen, Zhongfei Zhang, Anthony Dick, and Anton Van Den Hengel. A survey of appearance models in visual object tracking. *ACM TIST*, 4(4):1–48, 2013. 2

[38] Pengpeng Liang, Erik Blasch, and Haibin Ling. Encoding color information for visual tracking: Algorithms and benchmark. *TIP*, 24(12):5630–5644, 2015. 2, 7, 8

[39] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 6

[40] Mason Liu and Menglong Zhu. Mobile video object detection with temporally-aware feature maps. In *CVPR*, 2018. 2

[41] Chao Ma, Jia-Bin Huang, Xiaokang Yang, and Ming-Hsuan Yang. Hierarchical convolutional features for visual tracking. In *ICCV*, 2015. 1, 2, 5, 7

[42] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*, 2016. 1, 2, 7

[43] Adam Paszke et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 6

[44] Yuankai Qi, Shengping Zhang, Lei Qin, Hongxun Yao, Qingming Huang, Jongwoo Lim, and Ming-Hsuan Yang. Hedged deep tracking. In *CVPR*, 2016. 2

[45] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 2, 5

[46] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015. 6

[47] Arnold WM Smeulders, Dung M Chu, Rita Cucchiara, Simone Calderara, Afshin Dehghan, and Mubarak Shah. Visual tracking: An experimental survey. *TPAMI*, 36(7):1442–1468, 2013. 2

[48] Chong Sun, Dong Wang, Huchuan Lu, and Ming-Hsuan Yang. Learning spatial-aware regressions for visual tracking. In *CVPR*, 2018. 2, 5

[49] Yuxuan Sun, Chong Sun, Dong Wang, You He, and Huchuan Lu. Roi pooled correlation filters for visual tracking. In *CVPR*, 2019. 2

[50] Ran Tao, Efstratios Gavves, and Arnold WM Smeulders. Siamese instance search for tracking. In *CVPR*, 2016. 2

[51] Pavel Tokmakov, Karteek Alahari, and Cordelia Schmid. Learning video object segmentation with visual memory. In *ICCV*, 2017. 2

[52] Ludovic Trottier, Philippe Gigu, Brahim Chaib-draa, et al. Parametric exponential linear unit for deep convolutional neural networks. In *ICMLA*, 2017. 5

[53] Jack Valmadre, Luca Bertinetto, Joao Henriques, Andrea Vedaldi, and Philip HS Torr. End-to-end representation learning for correlation filter based tracking. In *CVPR*, 2017. 7

[54] Guangting Wang, Chong Luo, Zhiwei Xiong, and Wenjun Zeng. Spm-tracker: Series-parallel matching for real-time visual object tracking. In *CVPR*, 2019. 2, 5, 6, 7, 8

[55] Lijun Wang, Wanli Ouyang, Xiaogang Wang, and Huchuan Lu. Visual tracking with fully convolutional networks. In *ICCV*, 2015. 1, 2

[56] Naiyan Wang, Jianping Shi, Dit-Yan Yeung, and Jiaya Jia. Understanding and diagnosing visual tracking systems. In *ICCV*, 2015. 1

[57] Naiyan Wang and Dit-Yan Yeung. Learning a deep compact image representation for visual tracking. In *NIPS*, 2013. 1, 2

[58] Qiang Wang, Zhu Teng, Junliang Xing, Jin Gao, Weiming Hu, and Stephen Maybank. Learning attentions: residual attentional siamese network for high performance online visual tracking. In *CVPR*, 2018. 2

[59] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. *TPAMI*, 37(9):1834–1848, 2015. 2, 7, 8

[60] Fanyi Xiao and Jae Yong Lee. Video object detection with an aligned spatial-temporal memory. In *ECCV*, 2018. 2

[61] Yunhua Zhang, Lijun Wang, Jinqing Qi, Dong Wang, Mengyang Feng, and Huchuan Lu. Structured siamese network for real-time visual tracking. In *ECCV*, 2018. 7

[62] Zhipeng Zhang and Houwen Peng. Deeper and wider siamese networks for real-time visual tracking. In *CVPR*, 2019. 1, 2, 5, 7, 8

[63] Zheng Zhu, Qiang Wang, Bo Li, Wei Wu, Junjie Yan, and Weiming Hu. Distractor-aware siamese networks for visual object tracking. In *ECCV*, 2018. 2, 5, 7

[64] Zheng Zhu, Wei Wu, Wei Zou, and Junjie Yan. End-to-end flow correlation tracking with spatial-temporal attention. In *CVPR*, 2018. 1, 2