IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 28, 2021, accepted January 10, 2022, date of publication January 14, 2022, date of current version January 26, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3143809

Visual Tracking by Adaptive Continual Meta-Learning

JANGHOON CHOI[1], (Member, IEEE), SUNGYONG BAIK[2], (Member, IEEE), MYUNGSUB CHOI[3], (Member, IEEE), JUNSEOK KWON[4], (Member, IEEE), AND KYOUNG MU LEE[2], (Fellow, IEEE)

[1]College of Computer Science, Kookmin University, Seoul 02707, South Korea
[2]Department of ECE, ASRI, Seoul National University, Seoul 08826, South Korea
[3]Google Research, Seoul 06236, South Korea
[4]School of Computer Science and Engineering, Chung-Ang University, Seoul 06974, South Korea

Corresponding author: Junseok Kwon (jskwon@cau.ac.kr)

This work was supported by the Republic of Korea Government [Ministry of Science and ICT (MSIT)] through the Research Fund of the National Research Foundation of Korea (NRF) under Grant NRF-2021R1C1C2095450.

ABSTRACT We formulate the visual tracking problem as a semi-supervised continual learning problem, where only an initial frame is labeled. In contrast to conventional meta-learning based approaches that regard visual tracking as an instance detection problem with a focus on finding good weights for model initialization, we consider both initialization and online update processes simultaneously under our adaptive continual meta-learning framework. The proposed adaptive meta-learning strategy dynamically generates the hyperparameters needed for fast initialization and online update to achieve more robustness via adaptively regulating the learning process. In addition, our continual meta-learning approach based on knowledge distillation scheme helps the tracker adapt to new examples while retaining its knowledge on previously seen examples. We apply our proposed framework to deep learning-based tracking algorithm to obtain noticeable performance gains and competitive results against recent state-of-the-art tracking algorithms while performing at real-time speeds.

INDEX TERMS Continual learning, meta learning, object tracking, visual tracking.

I. INTRODUCTION

Visual tracking, which is one of the fundamental computer vision problems, has seen practical applications in robotics, automated surveillance, and autonomous driving. Given the initial video frame with a bounding box label of the target object, the goal of the visual tracking problem is to track the target object throughout the subsequent video frames without losing the target object. However, conventional tracking algorithms face several challenges in various circumstances such as scale change, occlusion, illumination change, deformation, background clutter, and motion blur.

Recently, with the advances in the application of deep convolutional neural networks (CNN) to image classification and object detection tasks [1]–[4], visual tracking algorithms have also achieved large improvements in performance, owing to the representation power of their deep backbone networks [5], [6] and the object detection framework [7], [8]. However, there is a misalignment between goals of object detection and visual tracking problem, where object detection aims to locate all objects of same semantic class whereas visual tracking aims to locate a specific object instance. To overcome this gap, visual tracking algorithms employ some form of domain adaptation process to the object detection framework, such as online network finetuning using stochastic gradient descent (SGD)-based methods [5], [9]–[11] or Siamese network structure [6], [7], [12]–[14] which generates a target-specific convolutional kernel from the initial frame.

While recent tracking algorithms were successful in achieving high performance metrics on several visual tracking benchmarks [15]–[17], the importance of the online adaptation process was often overlooked despite their crucial role in visual tracking. In particular, the tracker may need to update its model since the appearance of the target object constantly changes and similar distractor objects can appear in a given scene. Moreover, these aspects are further emphasized

The associate editor coordinating the review of this manuscript and approving it for publication was Inês Domingues.

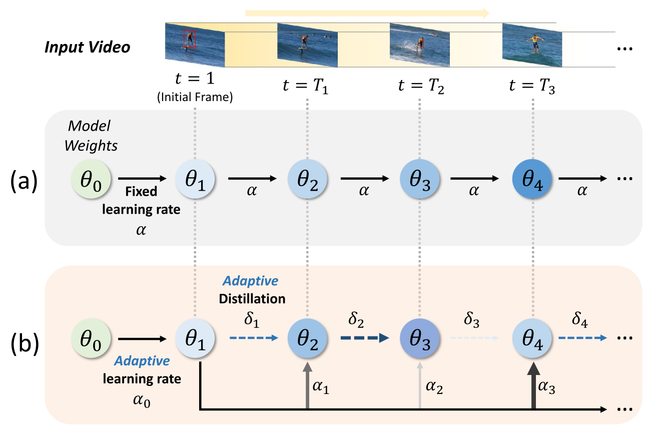


FIGURE 1. Motivation for the proposed visual tracking framework. Given an input video, (a) conventional tracking algorithm initializes and updates its model weights using fixed and predefined learning rates. (b) Our proposed tracking framework incorporates an adaptive learning scheme for both model initialization and online update using adaptive learning rate and adaptive knowledge distillation, which increases the flexibility of the tracker to learn new examples, while aiming to achieve robustness through retaining the memory on previously seen examples.

in long-term tracking scenarios [18]–[20] where the target can be absent for a prolonged time interval. Online update was often achieved by incorporating hand-crafted regularization schemes and meticulous hyperparameter selection, due to the lack of training samples and label uncertainty. In cases of most Siamese network-based trackers, online adaptation was completely ignored to achieve faster and real-time speeds. To address these aforementioned issues, several recent trackers employed meta-learning-based adaptation schemes [21]–[28] in order to learn the adaptation process itself. However, a majority of them either focused only on finding a good initialization for the tracker [21], [23] or only regulating the online update process [22], [25]–[27].

In this paper, we introduce a more generalized visual tracking framework, in which we model both adaptation and continual processes under our adaptive continual meta-learning framework. We adopt an adaptive scheme where hyperparameters can dynamically change to deal with various tracking scenarios. In addition, our continual meta-learning scheme employs adaptive knowledge distillation-based update strategy to help the tracker adapt to newly obtained examples while retaining the necessary knowledge on previously seen examples. During offline training, our integrated framework trains (1) network weights that are good for both initialization and future online updates; (2) hyperparameter generator network which adaptively generates the learning rates, instance-wise weights, and loss hyperparmeter for controlling the adaptation process; and (3) knowledge distiller network for regulating the balance between learning new examples and retaining the knowledge from the previous step.

To validate the effectiveness of our proposed framework, we apply our method to Siamese network-based tracking algorithm TACT [29], which is a two-stage detector-based tracker. We compare our method to other state-of-the-art trackers on test splits of large-scale visual tracking datasets, including LaSOT [18], OxUvA [19], TLP [20], TrackingNet [16], and GOT-10k [17]. We further demonstrate the effectiveness of our framework by component-wise ablation analyses on the LaSOT dataset. Our framework requires minimal computational overhead, achieving real-time speeds. Motivation for our tracking framework is shown in Figure 1.

II. RELATED WORK

A. DEEP NEURAL NETWORK-BASED TRACKING ALGORITHMS

Contemporary visual tracking algorithms solve visual tracking via tracking-by-detection, where they attempt to locate the target by finding the position where the classifier produces the highest classification score for the target class. With the powerful representation capacity of deep nerual networks, recent trackers employ deep neural networks for feature extraction and classification. While features from denoising autoencoder are used in [30], MDNet [5] used VGG [3] features with multi-task learning, which RT-MDNet has accelerated to real-time speed by using ROIAlign [31]. Correlation filter-based trackers [32], [33] have also been widely used on top of pretrained deep features, such as C-COT [34] and ECO [35], in which they use continuous convolutional operator for the fusion of multi-resolution CNN feature maps. Other approaches include spatially regularized filters [36], the fusion of multilevel features [37], group feature selection [38], and spatial transformers [39].

Recently, Siamese network-based trackers have gained traction due to their simplicity and high performance [6]–[8], [14], [40]–[47]. SiamFC [6] introduced a fully convolutional end-to-end approach with increased speed and accuracy. SiamRPN [7] added a region proposal network for more accurate localization and size estimation while DaSiamRPN [40] enhanced its discriminability by introducing negative pairs during training to suppress distractors. Both [41] and [42] utilized deeper and wider feature extractors based on [1] and [48] for further performance gains. Other works include general transformation learning model [43], local pattern detection for structure-based prediction [44], cascaded region proposal for sequential refinement [45], and recurrent optimization based model [46]. Recent approaches employ inverted residual networks [49], attentional cascade keypoints [13], sailency information [50], and relation networks [51] for Siamese networks. There also have been approaches to automatically find lightweight networks for efficient matching such as [52] and [53], inspired by network architecture search (NAS) methods.

Introduction of transformers [54] and self-attention [55] to computer vision applications also enabled utilization of additional temporal information for visual tracking. Recent approaches include end-to-end fully-convolutional networks [56], incorporating rich scene information [57], use of space-time memory networks [58], and feature fusion with transformers [59].

B. META-LEARNING FOR VISUAL TRACKING

To overcome the issues of conventional model adaptation in visual tracking, noteworthy methods have been proposed to improve the tracking performance by employing meta-learning based adaptation at test-time. Among meta-learning algorithms, model-agnostic meta-learning (MAML) [60] based approaches [61]–[65] recently gained attention owing to its simplicity and versatility. MAML aims to find good model weights that can be trained to generalize well with a small number of SGD steps and a small amount of training data. Meta-Tracker [21] was one of the first to apply MAML on MDNet [5] for fast adaptation, reducing the number of SGD iterations. MetaRTT [24] extended the idea by simultaneously finding the learning rates for initialization and online update. In addition, [23] used MAML to convert a modern detection network into a tracker. However, all above methods used fixed learning rates for all tracking scenarios and thus lack the adaptiveness to deal with diverse individual scenarios, which are accompanied by various training examples with varying degree of label uncertainty. Additionally, they do not address the erroneous updates performed with uncertain and mislabeled examples. On the other hand, our proposed method is able to adaptively change the hyperparameters to deal with these scenarios.

Other meta-learning-based tracking algorithms introduce a separate meta-learner network to regulate the adaptation process. [27] and [22] used loss gradient information obtained during tracking to update the target feature representation. Moreover, [25] used a separate update module to acquire the updated accumulated template. An optimization-based architecture with a model predictor was used in [11] to predict the filter weights while a similar approach using recurrent neural optimizer is proposed in [46]. However, a majority of the aforementioned methods are mainly focused on short-term tracking scenarios, and are not designed for long-term tracking scenarios. With the exception of [26], where they used a meta-updater network that takes multiple cues as input to make the binary decision whether to update or not to update the baseline tracker.

C. INCREMENTAL OBJECT CLASSIFICATION AND DETECTION

Conventional training setting for the classification problem assumes that abundant labeled training examples are always available for all classes at any point in training time. By contrast, incremental/continual setting assumes new examples or new classes are given in a sequential manner, thus the model has to be trained incrementally to prevent the model from catastrophic forgetting, which is a phenomenon where the performance of the model on previously seen examples significantly degrades over time. Recent approaches for deep neural networks include iCaRL [66] which learns the classifier and representation simultaneously based on replay memory; EWC [67] which selectively slows down learning of weights based on their importance; and LwF [68] where task-specific parameters from previous tasks are utilized with knowledge distillation loss to prevent the network from forgetting, while improving the performance on a new task. Related to LwF, incremental learning of object classification and detection models based on the knowledge distillation [69] scheme to prevent catastrophic forgetting have recently emerged [70]–[73].

Inspired by aforementioned approaches, we employ a knowledge distillation-based continual meta-learning scheme for our visual tracking framework. Different from conventional continual learning settings where new examples are given in a sequential manner along with their corresponding ground truth labels, these labels are not available under standard visual tracking setting. Since labels for new examples have to be obtained in a self-supervised manner, chance of adapting the model based on mislabeled examples persists. To alleviate this issue, we introduce two solutions. (1) When performing an online update at a certain time step, we always start from initially adapted weights where previous weights are used for knowledge distillation. This reduces error accumulation and overfitting to small number of training examples, while increasing the flexibility of the tracker. (2) We introduce an adaptive knowledge distiller network that predicts the importance weights for each previous frame where the magnitude of weights determine the degree of knowledge distilled from a certain frame. By controlling these weights, the tracker can choose between learning new examples and retaining the previous knowledge.

III. PROPOSED METHOD

Our proposed framework consists of two large components, which are the baseline tracking algorithm and the adaptive continual meta-learner module. In the following subsections, training procedure for our proposed adaptive continual meta-learner module is delineated.

Assuming a baseline tracker f_{θ_0} with its default weights θ_0, the meta-learner network g controls the learning process through adaptively generating the hyperparameters to modify the direction and magnitude of the loss gradients. The meta-learner network g contains four sub-networks, g_α, g_β, g_γ, and g_δ, where each sub-network generates the learning rate α, instance weight β, focal loss hyperparameter γ, and knowledge distillation hyperparameter δ, respectively. Our objective is to train the default weights θ_0 and network weights for g. To train both parameters, we construct a simulated tracking episode to perform the initial and online adaptation processes and assess how well these adaptations are conducted by evaluating the loss on the future frames. Our training scheme extends on the basic meta-learning formulation of dividing the training set into support and query sets, then performing inner-loop and outer-loop optimizations for meta-training analogous to such as in [60].

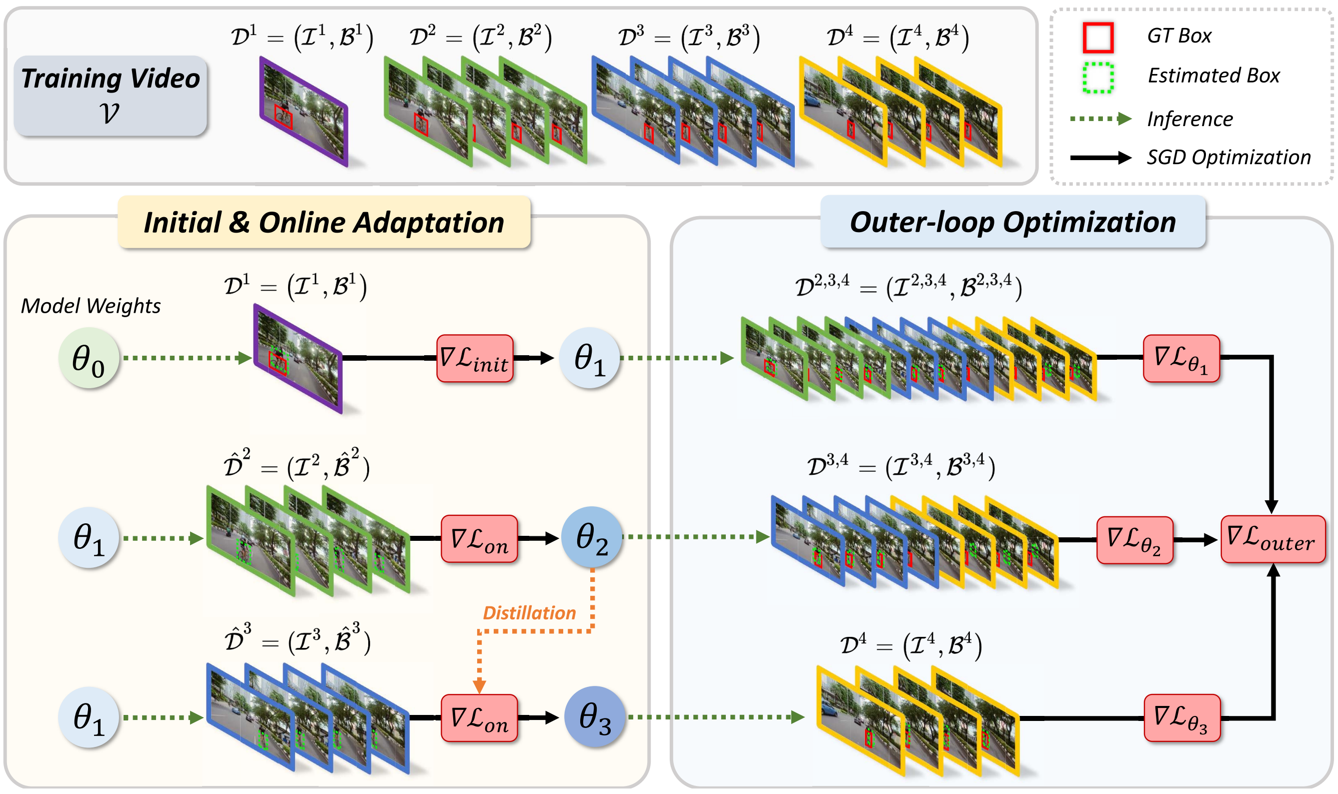

Training Video $\mathcal{V}$
$\mathcal{D}^1 = (\mathcal{I}^1, \mathcal{B}^1)$
$\mathcal{D}^2 = (\mathcal{I}^2, \mathcal{B}^2)$
$\mathcal{D}^3 = (\mathcal{I}^3, \mathcal{B}^3)$
$\mathcal{D}^4 = (\mathcal{I}^4, \mathcal{B}^4)$
GT Box
Estimated Box
Inference
SGD Optimization
Initial & Online Adaptation
Outer-loop Optimization
Model Weights
θ_0
θ_1
θ_2
θ_3
$\nabla\mathcal{L}_{init}$
$\hat{\mathcal{D}}^2 = (\mathcal{I}^2, \hat{\mathcal{B}}^2)$
$\hat{\mathcal{D}}^3 = (\mathcal{I}^3, \hat{\mathcal{B}}^3)$
$\nabla\mathcal{L}_{on}$
Distillation
$\mathcal{D}^{2,3,4} = (\mathcal{I}^{2,3,4}, \mathcal{B}^{2,3,4})$
$\mathcal{D}^{3,4} = (\mathcal{I}^{3,4}, \mathcal{B}^{3,4})$
$\nabla\mathcal{L}_{\theta_1}$
$\nabla\mathcal{L}_{\theta_2}$
$\nabla\mathcal{L}_{\theta_3}$
$\nabla\mathcal{L}_{outer}$

FIGURE 2. Overview for the training process of proposed framework. The training video $\mathcal{V}$ is divided into four datasets, $\{\mathcal{D}^1, \mathcal{D}^2, \mathcal{D}^3, \mathcal{D}^4\}$, where each dataset $\mathcal{D}^i$ contains frame images $\mathcal{I}^i$ and GT box labels $\mathcal{B}^i$. Initial adaptation is performed using the initial frame and label in $\mathcal{D}^1$, and online adaptations are conducted using self-supervised labels $\hat{\mathcal{B}}^2$, $\hat{\mathcal{B}}^3$ in $\hat{\mathcal{D}}^2$, $\hat{\mathcal{D}}^3$. Afterwards, outer loop optimization is performed to evaluate all adapted weights θ_i on $\mathcal{D}^{i+1,\cdots}$ for meta-training.

A. META-TRAINING WITH SIMULATED EPISODES

1) TRACKER SETTING

The baseline tracking algorithm f_θ with weights θ takes a video frame image I^t as an input and outputs K candidate bounding boxes $b_\theta^t \in \mathbb{R}^{K\times 4}$ with corresponding confidence values $c_\theta^t \in \mathbb{R}^K$, which is denoted as,

$$b_\theta^t, c_\theta^t = f_\theta\left(I^t\right). \tag{1}$$

Tracking is performed by choosing the bounding box with the highest confidence value as an output. Online updates are conducted by training the tracker using this chosen output box, in which other boxes are labeled as positive if they have high overlap with the output box (IoU$> \tau_p$) and negative otherwise (IoU$< \tau_n$). We chose overlap threshold values $\tau_p = 0.5$ and $\tau_n = 0.3$ for training and testing.

2) EPISODE SETTING

Given a training video sequence $\mathcal{V}$ of length T with its frame images $\{I^1, I^2, \ldots, I^T\}$ and ground truth target bounding box annotations $\{b^1, b^2, \ldots, b^T\}$, the video sequence is divided into four time-ordered video segments $\mathcal{I}^1, \mathcal{I}^2, \mathcal{I}^3$, and $\mathcal{I}^4$ with corresponding bounding box label sets $\mathcal{B}^1$, $\mathcal{B}^2$, $\mathcal{B}^3$, and $\mathcal{B}^4$. Then, each video segment and label set are paired to form four datasets $\mathcal{D}^1 = (\mathcal{I}^1, \mathcal{B}^1)$, $\mathcal{D}^2 = (\mathcal{I}^2, \mathcal{B}^2)$, $\mathcal{D}^3 = (\mathcal{I}^3, \mathcal{B}^3)$, and $\mathcal{D}^4 = (\mathcal{I}^4, \mathcal{B}^4)$, respectively. Each video segment contains frame images with sizes of $|\mathcal{I}^1| = 1$, $|\mathcal{I}^2| = |\mathcal{I}^3| = N$, and $|\mathcal{I}^4| = T - 2N - 1$, where N is the number of frames used for online adaptation.

Given a baseline tracker f_{θ_0} with the default weights θ_0, the initial adaptation is first performed using the dataset $\mathcal{D}^1 = (\mathcal{I}^1, \mathcal{B}^1)$ to obtain adapted weights θ_1. Then, using the initialized tracker f_{θ_1} on images in $\mathcal{I}^2$, we can obtain estimated labels $\hat{\mathcal{B}}^2$ to form the dataset for self-supervised online update $\hat{\mathcal{D}}^2 = (\mathcal{I}^2, \hat{\mathcal{B}}^2)$ where the tracker is updated from θ_1 to θ_2. Lastly, using the adapted tracker f_{θ_2}, online update is performed again with dataset $\hat{\mathcal{D}}^3 = (\mathcal{I}^3, \hat{\mathcal{B}}^3)$ to obtain θ_3. After simulating tracking episodes, we obtain intermediate weights θ_0, θ_1, θ_2, and θ_3 for the tracker. To train our overall framework, we evaluate the tracker on different combination of datasets based on each intermediate weight, then perform outer-loop optimization on the loss to train the default weights θ_0 and network weights for meta-learner g. The overview for the training process of our proposed framework is depicted in Figure 2.

3) INITIAL ADAPTATION

Our tracker first performs the initial adaptation process using the initial frame and label, $\mathcal{D}^1$. Starting from θ_0, the adapted

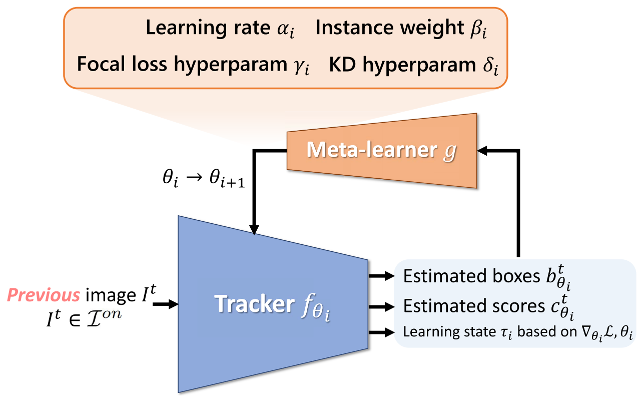

Learning rate α_i
Instance weight β_i
Focal loss hyperparam γ_i
KD hyperparam δ_i
Meta-learner g
$\theta_i \rightarrow \theta_{i+1}$
Previous image I^t
$I^t \in \mathcal{I}^{on}$
Tracker f_{θ_i}
Estimated boxes $b^t_{\theta_i}$
Estimated scores $c^t_{\theta_i}$
Learning state τ_i based on $\nabla_{\theta_i}\mathcal{L}, \theta_i$

FIGURE 3. Diagram of our proposed online adaptation process. Given an input frame image I^t, baseline tracking algorithm f_{θ_i} with weights θ_i returns estimated boxes and scores $b^t_{\theta_i}, c^t_{\theta_i}$ with its learning state τ_i based on $[\bar{\nabla}_{\theta_i}\mathcal{L}, \bar{\theta_i}]$. The meta-learner g receives these information as input and generates the learning rate α_i, instance weight β_i, focal loss hyperparameter γ_i, and knowledge distillation hyperparameter δ_i, where these hyperparameters control the adaptation process of $\theta_i \rightarrow \theta_{i+1}$.

weights θ_1 are obtained by,

$$\theta_1 = \theta_0 - \alpha_0 \odot \nabla_{\theta_0}\mathcal{L}_{init}\left(\mathcal{D}^1\right), \tag{2}$$

where $\alpha_0 = g_\alpha(\tau_0)$ is the predicted per-parameter learning rate and the input τ_0 is the learning state based on the layer-wise mean of gradients and kernels $[\bar{\nabla}_{\theta_0}\mathcal{L}, \bar{\theta_0}]$, as defined in [65]. The loss function for the initial adaptation $\mathcal{L}_{init}$ is defined as,

$$\mathcal{L}_{init}\left(\mathcal{D}^1\right) = \beta_0 \mathrm{FL}\left(c^1_{\theta_0}; \gamma_0\right), \tag{3}$$

where FL denotes the focal loss [74] evaluated using the initially given bounding box label $\mathcal{B}^1$, $\beta_0 = g_\beta\left(c^1_{\theta_0}\right)$ is the instance weight for the initial frame, and $\gamma_0 = g_\gamma\left(c^1_{\theta_0}\right)$ is the focusing hyperparameter used in focal loss to control the balance between losses of easy and hard samples. β_0 and γ_0 are both scalar values.

4) ONLINE ADAPTATIONS

Using the initially adapted tracker f_{θ_1} and frames in $\mathcal{I}^2$, online adaptation is performed using $\hat{\mathcal{D}}^2$ to obtain updated parameters θ_2 as in,

$$\theta_2 = \theta_1 - \alpha_1 \odot \nabla_{\theta_1}\mathcal{L}_{on}\left(\hat{\mathcal{D}}^2\right), \tag{4}$$

where $\alpha_1 = g_\alpha(\tau_1)$. Loss function for the online update is defined as,

$$\mathcal{L}_{on}\left(\hat{\mathcal{D}}^2\right) = \frac{1}{|\mathcal{I}^2|}\sum_{I^i \in \mathcal{I}^2}\left\{\beta^i_1 \mathrm{FL}\left(c^i_{\theta_1}; \gamma^i_1\right) + \delta^i_1 \mathrm{KD}\left(c^i_{\theta_1}, c^i_{\theta_0}\right)\right\}, \tag{5}$$

where FL is evaluated using self-labeled bounding boxes in $\hat{\mathcal{B}}^2$ as labels, $\beta^i_1 = g_\beta\left(c^i_{\theta_1}\right)$, and $\gamma^i_1 = g_\gamma\left(c^i_{\theta_1}\right)$. KD is the knowledge distillation loss equivalent to the standard binary cross entropy loss and is used to measure discrepancy between predictions made by the model with parameters θ_0 and θ_1. To control the degree of knowledge distilled from a certain example, the knowledge distillation hyperparameter $\delta^i_1 = g_\delta\left(c^i_{\theta_1}, c^i_{\theta_0}\right)$ is predicted, where δ^i_1 is a scalar value.

Afterwards, further online adaptation is performed using $\hat{\mathcal{D}}^3$, where $\hat{\mathcal{B}}^3$ is obtained by evaluating the tracker f_{θ_2} on frames in $\mathcal{I}^3$. Updated parameters θ_3 can be acquired by evaluating the equations analogous to the previous step as in Eq. (4) and Eq. (5) where,

$$\theta_3 = \theta_1 - \alpha_2 \odot \nabla_{\theta_1}\mathcal{L}_{on}\left(\hat{\mathcal{D}}^3\right), \tag{6}$$

$$\mathcal{L}_{on}\left(\hat{\mathcal{D}}^3\right) = \frac{1}{|\mathcal{I}^3|}\sum_{I^i \in \mathcal{I}^3}\left\{\beta^i_2 \mathrm{FL}\left(c^i_{\theta_1}; \gamma^i_2\right) + \delta^i_2 \mathrm{KD}\left(c^i_{\theta_1}, c^i_{\theta_2}\right)\right\}, \tag{7}$$

where online adaptation is performed from the initially adapted parameters θ_1 rather than previous-step parameters θ_2 to reduce the effect of erroneous updates. For our proposed meta-learning framework, knowledge distillation is used in the temporal domain to enforce long-term memory on the tracker. We use the tracker after k-th online update f_{θ_k} as the teacher network and utilize this network to generate soft labels for frames in $\hat{\mathcal{D}}^k$. When updating the tracker to obtain $f_{\theta_{k+1}}$, knowledge from f_{θ_k} can be transferred to $f_{\theta_{k+1}}$, where the amount of knowledge distilled can be controlled by scaling the KD loss term in equation (5) and (7) using

Algorithm 1 Visual Tracking With Meta-Learner

```
Input: Tracking algorithm $f_\theta$ with default weights $\theta_0$
       Trained meta-learner network $g$
       Tracking sequence of length $L$, $\{I^1, I^2, \ldots, I^L\}$
       Initial target bounding box coordinates $b^1$
Output: Target bounding box coordinates for each frame

# Initialization at $t = 1$
Form dataset $\mathcal{D}^1 = \left(I^1, b^1\right)$ for initial adaptation
Model initialization from $\theta_0$ using $\mathcal{D}^1$ as in Eq. (2) and (3),
updating $\theta \leftarrow \theta_1$

# For later frames in tracking sequence
for $t = 2$ to $L$ do
    Obtain candidate boxes $b^t_{\theta_i}$ and confidence scores
    $c^t_{\theta_i}$ from input frame $I^t$ as in Eq. (1)
    Choose box with the highest confidence score as
    output $\hat{b}^t$
    If an output is confident (PSR < $\tau_{on}$), store corre-
    sponding frame and output box $(I^t, \hat{b}^t)$ to dataset
    for online update $\hat{\mathcal{D}}^{on} = (\mathcal{I}^{on}, \hat{\mathcal{B}}^{on})$
    if t mod $U = 0$ and $|\mathcal{I}^{on}| \geq N$ then
        Online update from $\theta_1$ using $\theta_i$ and $N$ training
        samples from $\hat{\mathcal{D}}^{on}$ as in Eq. (6) and (7), updating
        $\theta \leftarrow \theta_{i+1}$ and $i \leftarrow i + 1$
        Clear buffer for dataset $\hat{\mathcal{D}}^{on}$
    end if
end for
```

TABLE 1. Comparison on the LaSOT test set.

	TACT-18 $\rightarrow$ **ConTACT-18**	TACT-50 $\rightarrow$ **ConTACT-50**	GlobalTrack [12]	ATOM [10]	DiMP-50 [11]	SiamRPN++ [41]	LTMU [26]	SPLT [75]	FCOS-MAML [23]	Ocean [8]	SiamFC [6]	CLNet [76]	SiamFC++ [77]	PrDiMP50 [78]
AUC	0.556 $\rightarrow$ 0.568	0.575 $\rightarrow$ 0.589	0.521	0.518	0.569	0.496	0.572	0.426	0.523	0.560	0.336	0.499	0.544	0.598
Precision	0.583 $\rightarrow$ 0.597	0.607 $\rightarrow$ 0.627	0.529	0.506	-	0.491	0.572	0.396	-	0.566	0.339	0.494	-	-
Norm. Precision	0.638 $\rightarrow$ 0.653	0.660 $\rightarrow$ 0.681	0.599	0.576	0.650	0.569	-	0.494	-	-	0.420	0.574	-	0.688
FPS	57 $\rightarrow$ 52	42 $\rightarrow$ 38	6	30	43	35	13	25.7	42	25	58	45.6	90	30

TABLE 2. Comparison on the OxUvA test set.

(%)	TACT-50 $\rightarrow$ **ConTACT-50**	GlobalTrack [12]	SPLT [75]	MBMD [79]	LTMU [26]	EBT [80]	SiamFC+R [6]	SINT [81]	LCT [82]	TLD [83]	MDNet [5]	ECO-HC [35]
MaxGM	70.9 $\rightarrow$ 76.3	60.3	62.2	54.4	75.1	28.3	45.4	32.6	39.6	43.1	34.3	31.4
TPR	80.9 $\rightarrow$ 79.1	57.4	49.8	60.9	74.9	32.1	42.7	42.6	29.2	20.8	47.2	39.5
TNR	62.2 $\rightarrow$ 73.6	63.3	77.6	48.5	75.4	0.0	48.1	0.0	53.7	89.5	0.0	0.0

TABLE 3. Comparison on the TLP dataset.

(%)	TACT-50 $\rightarrow$ **ConTACT-50**	LTMU [26]	GlobalTrack [12]	SPLT [75]	MDNet [5]	SiamFC [6]	ECO [35]	GOTURN [84]	TLD [83]
AUC	0.508 $\rightarrow$ 0.522	0.571	0.520	0.416	0.372	0.237	0.205	0.200	0.122

δ_k, which is generated by the meta-learner. Also, updating from θ_1 reduces error accumulation and overfitting to small number of training examples, while increasing the flexibility of the tracker. By controlling δ_k, the tracker can choose between learning new examples and retaining the previous long-term knowledge.

5) OUTER-LOOP OPTIMIZATION

After completing the simulated tracking episode on input video sequence $\mathcal{V}$, we obtain intermediate tracker weights θ_0, θ_1, θ_2, and θ_3. We perform meta-training on our overall tracking framework by evaluating and performing outer-loop optimization for the tracker with each intermediate weight, using different combinations of held-out datasets with ground truth annotations to encourage reduced overfitting and better generalization performance of each adaptation process. Overall outer-loop loss function $\mathcal{L}_{outer}$ for outer-loop optimization is given as following,

$$\mathcal{L}_{outer}(\mathcal{V}) = \lambda_0 \mathcal{L}_{\theta_0}\left(\mathcal{D}^{2,3,4}\right) + \lambda_1 \mathcal{L}_{\theta_1}\left(\mathcal{D}^{2,3,4}\right) + \lambda_2 \mathcal{L}_{\theta_2}\left(\mathcal{D}^{3,4}\right) + \lambda_3 \mathcal{L}_{\theta_3}\left(\mathcal{D}^{4}\right), \quad (8)$$

where λ_0, λ_1, λ_2, and λ_3 are stage-wise weighting hyper-parameters with sum of 1 and superscript on $\mathcal{D}$ indicates a combination of respective datasets (*i.e.*, $\mathcal{D}^{i,j} = \mathcal{D}^i \cup \mathcal{D}^j$). Each individual loss term $\mathcal{L}_{\theta_i}$ with respect to each weight θ_i is defined as,

$$\mathcal{L}_{\theta_i}(\mathcal{D}) = \sum_{(I^j, b^j) \in \mathcal{D}} \mathrm{FL}\left(c^j_{\theta_i}; \gamma_{outer}\right), \quad (9)$$

where focal loss FL is evaluated for binary class predictions $c^j_{\theta_i}$ obtained from the tracker with weights θ_i, using the ground truth bounding box b^j and γ_{outer} is fixed to 0.5. Each loss $\mathcal{L}_{\theta_i}$, except for $\mathcal{L}_{\theta_0}$, is evaluated with dataset $\mathcal{D}^{i+1,\cdots}$ to measure the generalization performance on unseen future frames, assessing the quality of the adaptation process conducted from the previous weights θ_{i-1} using the meta-learner network g. It also serves to facilitate the tracker with weights θ_i to make more accurate predictions for subsequent frames in $\mathcal{I}^{i+1}$ for better future self-supervised update using the estimated labels, $\hat{\mathcal{D}}^{i+1} = (\mathcal{I}^{i+1}, \hat{\mathcal{B}}^{i+1})$. Note that for all aforementioned focal loss terms FL, additional IoU loss term evaluated on b^t_θ for bounding box regression is omitted for simplicity.

The process of outer-loop optimization is identical to the gradient-based bilevel optimization process of MAML [60] and its variants [61]–[65], where MAML aims to find good model weights that can be trained to generalize well on unseen tasks with small amount of training data. Key difference between the original MAML and our proposed method is that in addition to finding the good model weights θ_0 for generalization, our method incorporates the meta-learner network g, where g is also trained at the outer-loop optimization stage since it participated in generating the intermediate weights $\theta_1, \theta_2, \theta_3$. By obtaining gradients $\nabla_{\theta_0}\mathcal{L}_{outer}, \nabla_{\theta_1}\mathcal{L}_{outer}, \nabla_{\theta_2}\mathcal{L}_{outer}, \nabla_{\theta_3}\mathcal{L}_{outer}, \nabla_{\phi}\mathcal{L}_{outer}$ where ϕ represents the weights for the meta-learner network g, we can train our framework using an off-the-shelf optimizer.

TABLE 4. Comparison on the TrackingNet test set.

(%)	TACT-18 → **ConTACT-18**	TACT-50 → **ConTACT-50**	GlobalTrack [12]	ATOM [10]	DiMP-50 [11]	SiamRPN++ [41]	DASiam [40]	UPDT [37]	FCOS-MAML [23]	SiamFC [6]	ROAM++ [46]	ECO [35]	SiamFC++ [77]	PrDiMP50 [78]
Precision	70.1 → 71.6	70.8 → 72.7	65.6	64.8	68.7	69.4	59.1	55.7	-	53.3	62.3	49.2	70.5	70.4
Norm. Precision	78.4 → 79.9	78.8 → 80.7	75.4	77.1	80.1	80.0	73.3	70.2	82.2	66.6	75.4	61.8	80.0	81.6
Success	73.4 → 75.0	74.0 → 75.8	70.4	70.3	74.0	73.3	63.8	61.1	75.7	57.1	67.0	55.4	75.4	75.8

TABLE 5. Comparison on the GOT-10k test set.

(%)	TACT-18 → **ConTACT-18**	TACT-50 → **ConTACT-50**	ATOM [10]	DiMP-50 [11]	SiamMask [85]	Ocean [8]	CFNet [86]	SiamFC [6]	GOTURN [84]	ROAM++ [46]	ECO [35]	CF2 [87]	MDNet [5]	SiamFC++ [77]	PrDiMP50 [78]
$\mathbf{SR}_{0.50}$	64.8 → 66.9	66.5 → 68.7	63.4	71.7	58.7	72.1	40.4	35.3	37.5	53.2	30.9	29.7	30.3	69.5	73.8
$\mathbf{SR}_{0.75}$	44.7 → 47.1	47.7 → 48.7	40.2	49.2	36.6	-	14.4	9.8	12.4	23.6	11.1	8.8	9.9	47.9	54.3
AO	55.9 → 57.7	57.8 → 59.4	55.6	61.1	51.4	61.1	37.4	34.8	34.7	46.5	31.6	31.5	29.9	59.5	63.4

B. VISUAL TRACKING WITH META-LEARNER

Herein, we propose the visual tracking with a novel adaptive continual meta-learner. The tracking process is purposely made simple to retain the speed of the original backbone tracking algorithm while requiring as small memory overhead as possible. Given an input tracking sequence of length L, the proposed initial adaptation process is performed using the initial frame I^1 and bounding box b^1 to obtain initial weights θ_1 for update. During the tracking process, frames that yield output confidence values with peak-to-sidelobe ratios (PSR) smaller than $\tau_{on} = 0.7$ are considered as confident frames and stored to the dataset for online update $\hat{\mathcal{D}}^{on}$. Online update is performed every $U = 100$ frames by employing N most confident frames from the dataset $\hat{\mathcal{D}}^{on}$ and initial weights θ_1, updating the weights θ_{i-1} to θ_i, and the buffer for $\hat{\mathcal{D}}^{on}$ is cleared after every update. The overall tracking procedure is organized and described in Algorithm 1, and Figure 3 shows the diagram for our proposed online adaptation process.

IV. EXPERIMENTS

In this section, we elaborate on the implementation details for the backbone tracker and the proposed meta-learning framework, followed by the experimental results to validate the performance gains obtained by our framework on five large-scale visual tracking benchmark datasets. We also demonstrate the results for attribute-wise and module-wise ablation experiments to further analyze the effectiveness of our proposed method.

A. IMPLEMENTATION DETAILS

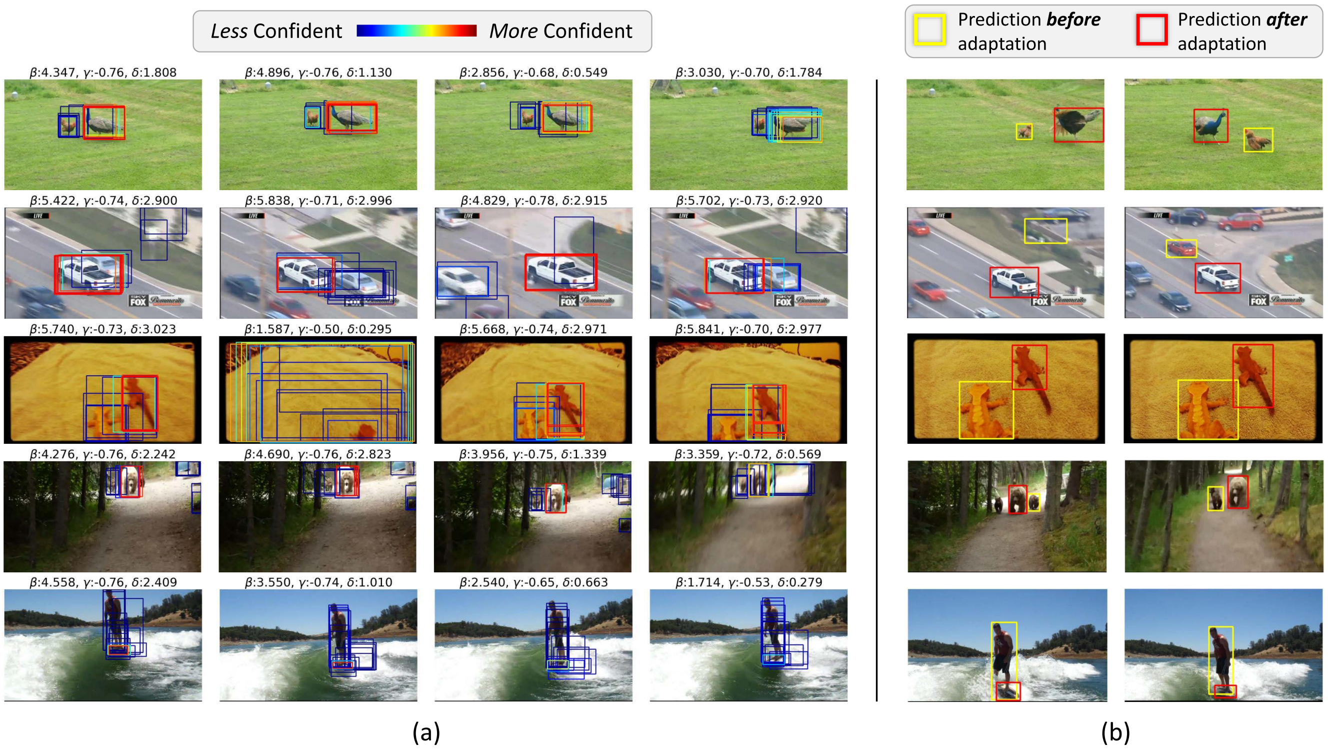
1) BACKBONE TRACKER

We employ Siamese network-based backbone tracking algorithm based on TACT [29], which is a variant of a two-stage detection network. TACT is based on GlobalTrack [12] and (1) is a long-term oriented tracker which fits our purpose, (2) is a full-frame search-based tracker where we can consider all potential distractors in a scene for update, and (3) has no hand-crafted motion smoothness constraints. Considering the aforementioned aspects, we can validate that the performance changes come solely from our proposed meta-learning framework without any influence from other potential variables. While freezing the weights of the feature extractor layers, the region proposal layers, and context embedding layers, we perform meta-training on the last ROI classification and refinement layers, starting from the original weights of TACT. We refer the modified trackers as **ConTACT-18** and **ConTACT-50** which are extensions of TACT-18 and TACT-50, respectively, with adaptive continual meta-learners.

2) META-LEARNER ARCHITECTURE

For the meta-learner g, its sub-networks $g_\alpha, g_\beta, g_\gamma$, and g_δ are all 3-layer MLPs with group normalization [88] and ReLU activation between the linear layers. The number of intermediate hidden units for each sub-network are 128, 256, 256, and 512, respectively. Assuming L is the number of layers that are involved in the adaptation process, the adaptive learning rate generator g_α takes $2L$-dimensional learning state as an input and returns L-dimensional layer-wise multipliers which are then multiplied to the per-parameter base learning rate α_{base}, similar to [61], and can be applied to each layer for SGD. Elaborating on the $2L$-dimensional learning state τ, its dimension is determined by the number of layers L of the backbone tracker network f_θ. The backbone tracker f_θ TACT [29] is based on a two-stage object detection framework, and we chose to use the final ROI classification and refinement layers of TACT for the adaptation process, which consist of 5-layer CNNs, resulting in $L = 5$ for our implementation of ConTACT. The learning state τ is constructed in a similar manner as in [65], where we concatenate the L-dimensional layer-wise mean of kernel weights and L-dimensional layer-wise mean of kernel gradients. L is fixed throughout the process of adaptations.

The adaptive instance weight generator g_β and adaptive focusing hyperparameter generator g_γ both take confidence values $c^t_{\theta_i} \in \mathbb{R}^K$ obtained from a given frame as input and returns scalar values β^t_i and γ^t_i. The adaptive knowledge



Less Confident
More Confident
Prediction *before* adaptation
Prediction *after* adaptation
β:4.347, γ:-0.76, δ:1.808
β:4.896, γ:-0.76, δ:1.130
β:2.856, γ:-0.68, δ:0.549
β:3.030, γ:-0.70, δ:1.784
β:5.422, γ:-0.74, δ:2.900
β:5.838, γ:-0.71, δ:2.996
β:4.829, γ:-0.78, δ:2.915
β:5.702, γ:-0.73, δ:2.920
β:5.740, γ:-0.73, δ:3.023
β:1.587, γ:-0.50, δ:0.295
β:5.668, γ:-0.74, δ:2.971
β:5.841, γ:-0.70, δ:2.977
β:4.276, γ:-0.76, δ:2.242
β:4.690, γ:-0.76, δ:2.823
β:3.956, γ:-0.75, δ:1.339
β:3.359, γ:-0.72, δ:0.569
β:4.558, γ:-0.76, δ:2.409
β:3.550, γ:-0.74, δ:1.010
β:2.540, γ:-0.65, δ:0.663
β:1.714, γ:-0.53, δ:0.279
(a)
(b)

FIGURE 4. Visualization for the effectiveness of the meta-learner. (a) Self-labeled frames for online adaptation, with candidate boxes color-coded according to their confidence values, along with β, γ and δ values generated by the meta-learner. (b) Future frames with predicted output boxes before and after the online adaptation with corresponding frames in (a).

TABLE 6. Attribute-wise ablation on LaSOT test set.

	TACT-18	**ConTACT-18**	TACT-50	**ConTACT-50**
ARC	0.551	0.565	0.570	0.584
BC	0.477	0.494	0.486	0.509
DEF	0.586	0.599	0.609	0.629
FOC	0.456	0.469	0.478	0.491
IV	0.579	0.595	0.613	0.627
ROT	0.558	0.573	0.582	0.596

TABLE 7. Component-wise ablation on LaSOT test set.

	Variant	**ConTACT-18**	**ConTACT-50**
(1)	Full Model (ConTACT)	0.5688	0.5889
(2)	**(1)** w/o adaptive weighting	0.5649	0.5826
(3)	**(2)** w/o adaptive focal loss	0.5639	0.5808
(4)	**(3)** w/o adaptive KD	0.5619	0.5790
(5)	**(4)** w/o adaptive learning rate	0.5608	0.5773
(6)	**(5)** w/o per-param. learning rate	0.5532	0.5721
(7)	**(1)** w/o online adaptation from θ_1	0.5583	0.5762
(8)	**(1)** w/o online adaptation	0.5620	0.5804
(9)	w/o any adaptation (TACT)	0.5562	0.5755

distiller g_δ takes two confidence vectors obtained from two different models as an input, $[c^t_{\theta_i}, c^t_{\theta_{i-1}}] \in \mathbb{R}^{2K}$, where $[\cdot, \cdot]$ denotes concatenation, and outputs a single scalar value δ^t_i.

3) TRAINING DETAILS

Dimensions of input images are the same as in [29] and the overall framework is trained with training splits of ImageNetVID [89], GOT-10k [17], LaSOT [18], and TrackingNet [16], from which a video sequence $\mathcal{V}$ is randomly chosen. $T = 13$ frames, in turn, are uniformly sampled inside a time window of 500 frames inside $\mathcal{V}$, along with their bounding box annotations. Among sampled frames, the first frame is used as $\mathcal{D}^1$. As for the remaining 12 frames, $N = 4$ frames are assigned to each of $\mathcal{D}^2$, $\mathcal{D}^3$ and $\mathcal{D}^4$ in a sequential order. For all frames and annotations in $\mathcal{V}$, random image augmentations, such as Gaussian noise, blur, horizontal flips, and bounding box jittering are applied. For online adaptation, we choose a box with highest confidence from $K = 64$ candidate boxes for a given frame and use this box as self-supervision.

Self-supervision is performed based on the estimated bounding box, where a best candidate box is chosen for a single image and can be considered as the pseudo ground-truth box. Based on this pseudo GT box, $K = 64$ candidate boxes estimated in an image can be labeled for classification (positive or negative) by calculating the IoU scores between the pseudo GT box. Candidate boxes having scores larger than $\tau_p = 0.5$ are labeled positive, and boxes with scores less than $\tau_n = 0.4$ are labeled negative, where classification

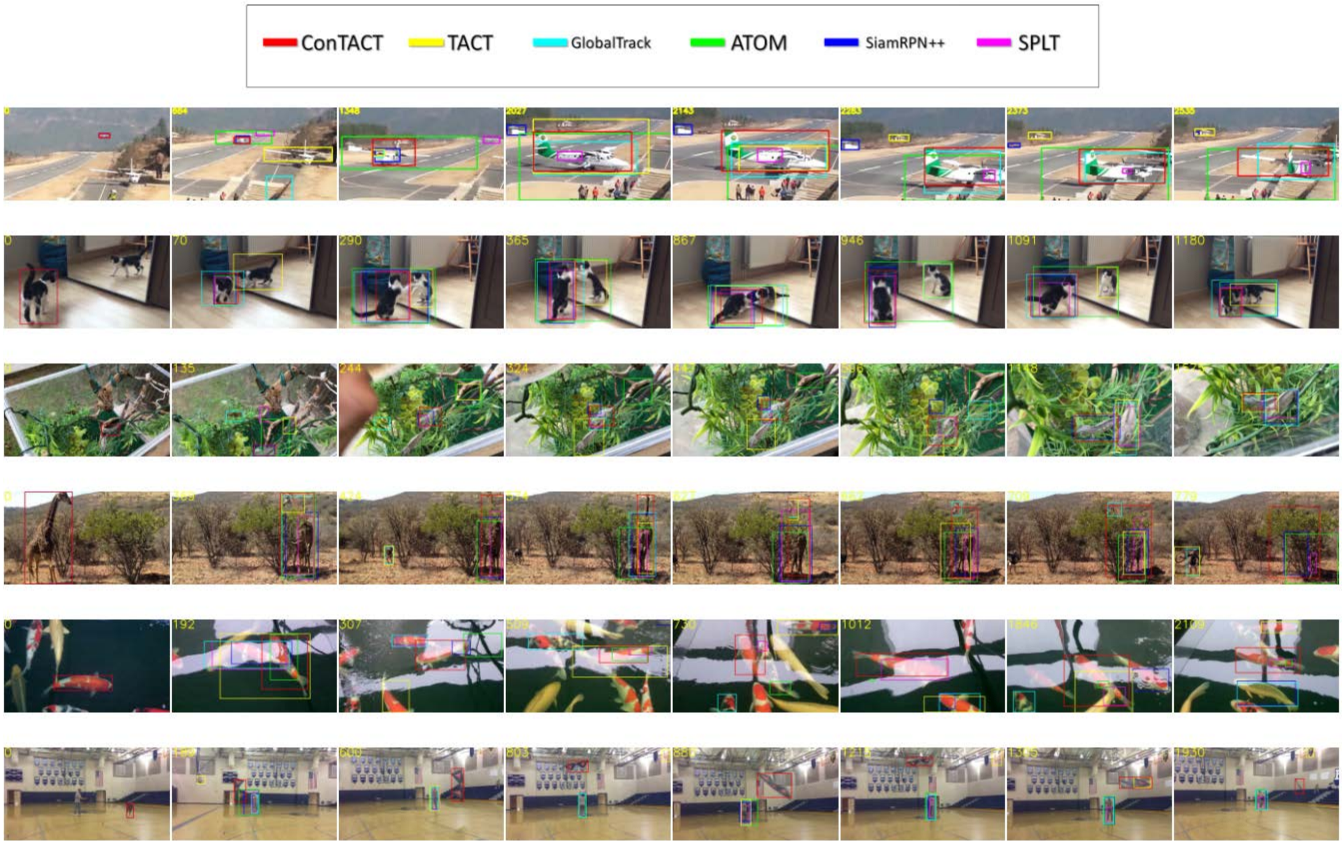


FIGURE 5. **Qualitative comparison with other trackers. Results are shown for sequences *airplane-1, cat-20, chameleon-20, giraffe-15, goldfish-8, and kite-6*. Best viewed zoomed in on a high resolution display.**

losses are enforced. For positive candidate boxes, bounding box regression loss (IoU loss) is calculated and used for adaptation. Our online training scheme is identical to the training scheme of the original tracker [29] and Faster R-CNN [4]. For a single dataset $\hat{\mathcal{D}}^i$ with $N = 4$ images in $\mathcal{I}^i$, corresponding $N = 4$ best candidate boxes $\hat{\mathcal{B}}^i$ are first estimated using the tracker, where these boxes are used as the pseudo ground-truth boxes for each frame. Using $\hat{\mathcal{D}}^i = (\mathcal{I}^i, \hat{\mathcal{B}}^i)$, we can train our tracker with self-supervision, where classification loss and regression loss can be enforced on $N \times K = 256$ estimated candidate boxes following the aforementioned procedure.

For both initial and online adaptations, per-parameter base learning rate α_{base} are initialized to 10^{-3} and single-step SGD update is performed for faster speed. For the outer-loop optimization, Adam [90] optimizer with learning rate of 10^{-5} is used with weight decay of 10^{-5} and trained for 5×10^5 iterations with batch size of 4.

B. QUALITATIVE AND QUANTITATIVE EVALUATION

1) EVALUATION DATASETS AND METRICS

We conducted evaluations for our trackers on test splits of five large-scale visual tracking benchmark datasets: LaSOT [18], OxUvA [19], TLP [20], TrackingNet [16], and GOT-10k [17]. LaSOT, OxUvA, and TLP are long-term visual tracking benchmarks with average sequence lengths longer than 1 min., whereas TrackingNet and GOT-10k have shorter sequence lengths with larger number of sequences with more various semantic classes of objects. **LaSOT** [18] dataset is a large-scale and long-term tracking dataset with 1, 400 video sequences for training and testing, with an average of 2, 512 frames ($\approx$ 83 secs) in length, and are annotated with target bounding boxes. We evaluated our trackers on the test split (Protocol II) of 280 video sequences, and report the performance metrics of area-under-curve (AUC) of the success plot, location precision, and normalized precision for comparison. **OxUvA** [19] dataset is focused on long-term tracking performance of a tracker where its dev and test splits have 200 and 166 sequences, respectively, with an average length of 4,260 frames ($\approx$ 142 secs). Since target can leave and reappear in a frame under the long-term tracking scenario, trackers must report the target bounding boxes as well as whether the target is present or absent in a given frame. The performance metrics are the maximum geometric mean (MaxGM) over the true positive rate (TPR) and the true negative rate (TNR), with IoU thresholds of 0.5. **TLP** [20] dataset also evaluates the long-term tracking

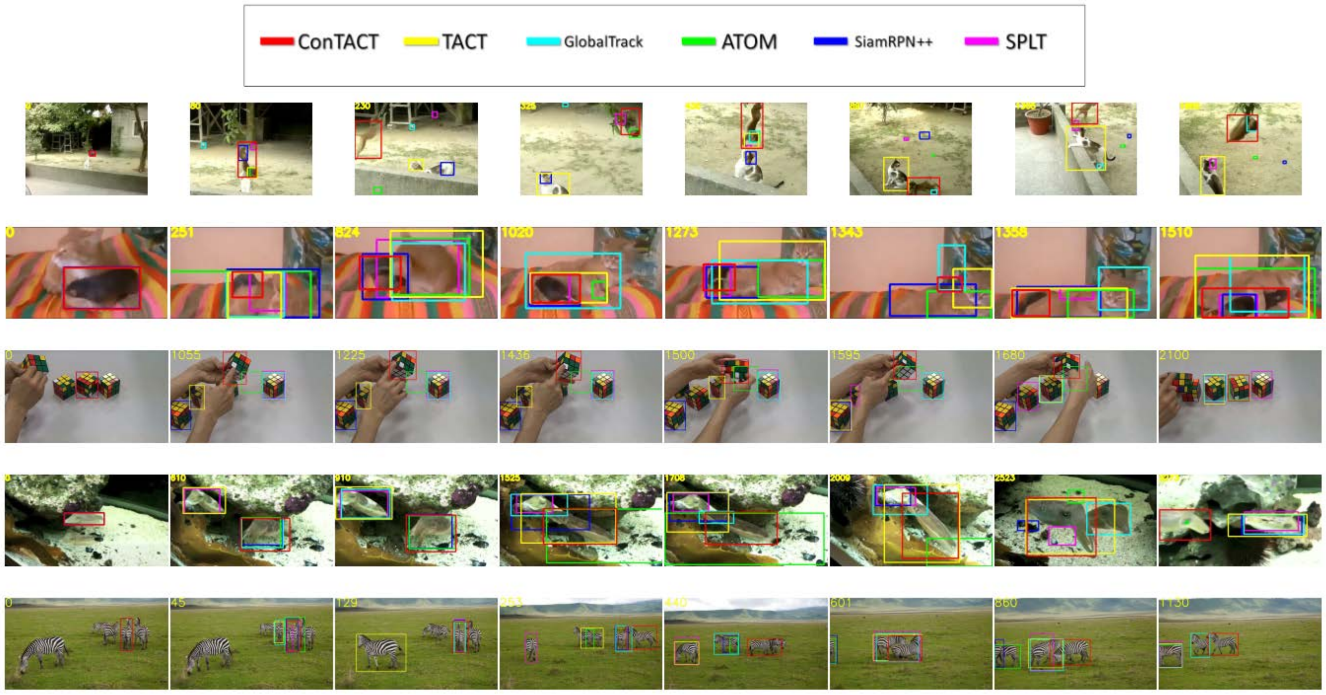


FIGURE 6. **Additional qualitative comparison with other trackers tested on real-world videos. Best viewed zoomed in on a high resolution display.**

performance where it contains 50 HD real-world videos, with average sequence length of 13,500 frames (≈ 450 secs). AUC of the success plot is used as the performance metric. **TrackingNet** [16] is a large-scale tracking dataset with more than 30, 000 videos gathered from YouTube, of which 511 sequences assigned as the test split. Similar to the other tracking benchmarks, location precision, normalized precision, and AUC of the success plot are used as performance metrics. **GOT-10k** [17] is a dataset composed under the one-shot experiment setting where the training and test splits have disjoint set of object classes. It contains 10, 000 video sequences of which 420 are used in the test split. Performance metrics are calculated by the success rate (SR, with thresholds 0.5 and 0.75) and average overlap (AO).

2) COMPARISON TO OTHER TRACKERS

Results for evaluation of our trackers on the LaSOT test set are provided in Table 1. Applying the proposed adaptive continual meta-learner on both variants of TACT, denoted as **ConTACT-18** and **ConTACT-50**, show consistent and noticeable gains on all performance metrics on both variants, while retaining real-time speeds of 52 fps and 38 fps. Both variants outperform many recent ResNet-based tracking algorithms, GlobalTrack [12], ATOM [10], DiMP [11], SiamRPN++ [41], SPLT [75], and Ocean [8]. For further evaluation of the long-term tracking capabilities, we evaluated our tracker on the OxUvA test set and presented the results in Table 2. To detect the absence of the target, we simply used confidence threshold value of 0.97 to label target as absent if confidence is below this threshold. The proposed method shows substantial performance gains in MaxGM and TNR metrics compared to TACT, where the performance gains are more pronounced under long-term sequences. Evaluation on relatively short-term, large-scale tracking benchmarks TrackingNet and GOT-10k are shown in Table 4 and 5.

Both of our trackers show consistent performance gains on all metrics for both datasets, validating the effectiveness of our proposed meta-learner on both long-term and short-term tracking applications where performance improvements are more pronounced in the long-term tracking applications. The baseline tracker of our algorithm is TACT [29], which is based on GlobalTrack [12] where GlobalTrack is a full-frame search-based tracker with no hand-crafted motion smoothness constraints (local search, cosine window penalty, linear interpolation between bounding boxes, etc.) commonly used in other tracking algorithms, and GlobalTrack requires minimal hyperparmeter tuning. Due to the aforementioned characteristics, TACT and GlobalTrack perform better on long-term tracking benchmarks such as LaSOT and OxUvA and the performance gains made by our proposed algorithm are less pronounced on short-term tracking benchmarks such as TrackingNet and GOT-10k. Despite these differences in characteristics, our proposed ConTACT-18 and ConTACT-50 successfully improves the baseline tracking algorithm TACT by noticeable margins, with competitive performance even compared with other recently published tracking algorithms. Qualitative comparison between other trackers, TACT [29], GlobalTrack [12], ATOM [10], SiamRPN++ [41], and SPLT [75], are shown in Figure 6.

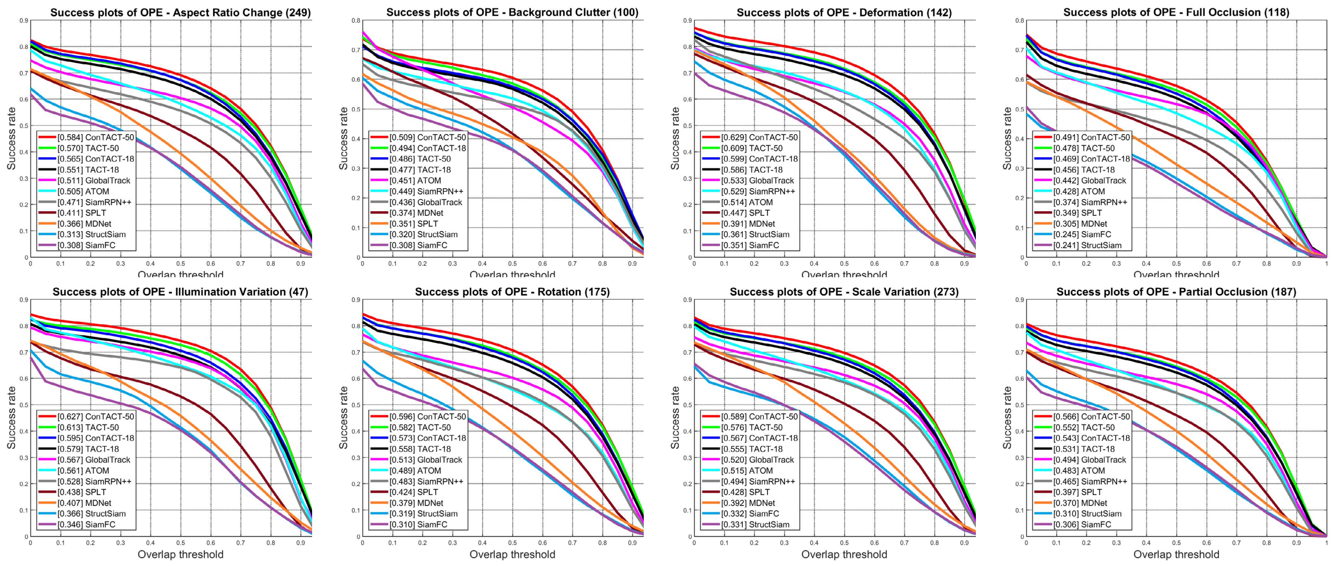


FIGURE 7. **Attribute-wise success plots for the test split of the LaSOT dataset.**

C. ANALYSIS

1) ABLATION STUDY

a: ATTRIBUTE-WISE ABLATION

To further analyze the effectiveness of proposed continual meta-learner, we show attribute-wise AUC performance on the LaSOT test set in Table 6, with comparison on six different challenge attributes of LaSOT. Displaying performance gains in all attributes, largest improvement comes from BC (background clutter) attribute, which validates the effectiveness of our initial and online adaptation strategy on eliminating the hard negatives while tracking.

Additional attribute-wise success plots with comparison between other tracking algorithms are also shown in Figure 7. Both variants of the proposed algorithm show competitive performance on multiple challenge attributes compared to other state-of-the-art trackers.

b: COMPONENT-WISE ABLATION

To verify the contribution of each component in our meta-learning framework, component-wise ablation results are shown in Table 7, where we sequentially remove each adaptive learning component in **(2)**-**(5)**. The results suggest that every component contributes to performance gain, where adaptive instance weighting contributes the most. Results in **(8)** show that our adaptive learning approach is effective even without any online adaptation, where only initial adaptation is adaptively performed. Regarding the online adaptation, results in **(6)**, which are obtained with naïve online fine-tuning with learning rate of 10^{-3} on TACT, show reduced performance possibly due to erroneous updates and over-fitting. Also, results in **(7)** suggest that online adaptation from the initial weights θ_1 instead of previous weights θ_{i-1} contributes to a large performance gain, owing to reduced error accumulation.

2) VISUALIZING THE ADAPTIVE LEARNING

In Figure 4, we show five video examples of online adaptation with self-labeled training samples, where erroneous predictions in the future frames are corrected after the adaptation. During the adaptation process, β, γ and δ values predicted by the meta-learner for each training sample dynamically change. The meta-learner assigns relatively lower β and δ values to examples with less confident, uncertain predictions while the negative γ value consistently directs to focus more on maximizing the class margin for confident examples, giving less attention to ambiguous examples that may lead the tracker to fail in the future.

V. CONCLUSION

In this paper, we proposed a novel adaptive continual meta-learning framework for visual tracking that dynamically generates the hyperparameters needed for initialization and online update with self-labeled examples. Also, our continual meta-learning approach based on knowledge distillation scheme helps the tracker adapt to new examples while retaining its knowledge on previously seen examples. We apply our proposed framework to deep learning-based tracking algorithm, where our **ConTACT-18** and **ConTACT-50** achieve noticeable performance gains and competitive results against recent state-of-the-art tracking algorithms on **all** five large-scale visual tracking benchmarks, while running at real-time speeds.

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105.

[3] K. Simonyan and A. Zisserman, ''Very deep convolutional networks for large-scale image recognition,'' 2014, *arXiv:1409.1556*.

[4] S. Ren, K. He, R. Girshick, and J. Sun, ''Faster R-CNN: Towards real-time object detection with region proposal networks,'' in *Proc. NIPS*, 2015, pp. 91–99.

[5] H. Nam and B. Han, ''Learning multi-domain convolutional neural networks for visual tracking,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 4293–4302.

[6] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, ''Fully-convolutional Siamese networks for object tracking,'' 2016, *arXiv:1606.09549*.

[7] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, ''High performance visual tracking with Siamese region proposal network,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8971–8980.

[8] Z. Zhang, H. Peng, J. Fu, B. Li, and W. Hu, ''Ocean: Object-aware anchor-free tracking,'' in *Proc. ECCV*, 2020, pp. 771–787.

[9] I. Jung, J. Son, M. Baek, and B. Han, ''Real-time MDNet,'' in *Proc. ECCV*, 2018, pp. 83–98.

[10] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, ''ATOM: Accurate tracking by overlap maximization,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4660–4669.

[11] G. Bhat, M. Danelljan, L. Van Gool, and R. Timofte, ''Learning discriminative model prediction for tracking,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6182–6191.

[12] L. Huang, X. Zhao, and K. Huang, ''GlobalTrack: A simple and strong baseline for long-term tracking,'' in *Proc. AAAI*, 2019, pp. 11037–11044.

[13] E. Wang, D. Wang, Y. Huang, G. Tong, S. Xu, and T. Pang, ''Siamese attentional cascade keypoints network for visual object tracking,'' *IEEE Access*, vol. 9, pp. 7243–7254, 2021.

[14] M. Ondrasovic and P. Tarabek, ''Siamese visual object tracking: A survey,'' *IEEE Access*, vol. 9, pp. 110149–110172, 2021.

[15] Y. Wu, J. Lim, and M. H. Yang, ''Object tracking benchmark,'' *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1834–1848, Sep. 2015.

[16] M. Ḿuller, A. Bibi, S. Giancola, S. Alsubaihi, and B. Ghanem, ''TrackingNet: A large-scale dataset and benchmark for object tracking in the wild,'' in *Proc. ECCV*, 2018, pp. 300–317.

[17] L. Huang, X. Zhao, and K. Huang, ''GOT-10k: A large high-diversity benchmark for generic object tracking in the wild,'' *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 5, pp. 1562–1577, May 2021.

[18] H. Fan, H. Ling, L. Lin, F. Yang, P. Chu, G. Deng, S. Yu, H. Bai, Y. Xu, and C. Liao, ''LaSOT: A high-quality benchmark for large-scale single object tracking,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5374–5383.

[19] J. Valmadre, L. Bertinetto, J. F. Henriques, R. Tao, A. Vedaldi, A. W. Smeulders, P. H. Torr, and E. Gavves, ''Long-term tracking in the wild: A benchmark,'' in *Proc. ECCV*, 2018, pp. 670–685.

[20] A. Moudgil and V. Gandhi, ''Long-term visual object tracking benchmark,'' in *Proc. ACCV*, 2018, pp. 629–645.

[21] E. Park and A. C. Berg, ''Meta-tracker: Fast and robust online adaptation for visual object trackers,'' in *Proc. ECCV*, 2018, pp. 569–585.

[22] J. Choi, J. Kwon, and K. M. Lee, ''Deep meta learning for real-time target-aware visual tracking,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 911–920.

[23] G. Wang, C. Luo, X. Sun, Z. Xiong, and W. Zeng, ''Tracking by instance detection: A meta-learning approach,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6288–6297.

[24] I. Jung, K. You, H. Noh, M. Cho, and B. Han, ''Real-time object tracking via meta-learning: Efficient model adaptation and one-shot channel pruning,'' in *Proc. AAAI*, 2020, pp. 11205–11212.

[25] L. Zhang, A. Gonzalez-Garcia, J. V. D. Weijer, M. Danelljan, and F. S. Khan, ''Learning the model update for Siamese trackers,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 4010–4019.

[26] K. Dai, Y. Zhang, D. Wang, J. Li, H. Lu, and X. Yang, ''High-performance long-term tracking with meta-updater,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6298–6307.

[27] P. Li, B. Chen, W. Ouyang, D. Wang, X. Yang, and H. Lu, ''GradNet: Gradient-guided network for visual object tracking,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6162–6171.

[28] L. Huang, X. Zhao, and K. Huang, ''Bridging the gap between detection and tracking: A unified approach,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3999–4009.

[29] J. Choi, J. Kwon, and K. M. Lee, ''Visual tracking by TridentAlign and context embedding,'' in *Proc. ACCV*, 2020, pp. 1–17.

[30] N. Wang and D.-Y. Yeung, ''Learning a deep compact image representation for visual tracking,'' in *Proc. NIPS*, 2013, pp. 1–10.

[31] K. He, G. Gkioxari, P. Dollár, and R. Girshick, ''Mask R-CNN,'' in *Proc. ICCV*, 2017, pp. 2961–2969.

[32] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, ''High-speed tracking with kernelized correlation filters,'' *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 583–596, Mar. 2015.

[33] D. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, ''Visual object tracking using adaptive correlation filters,'' in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 2544–2550.

[34] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg, ''Beyond correlation filters: Learning continuous convolution operators for visual tracking,'' in *Proc. ECCV*, 2016, pp. 472–488.

[35] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, ''ECO: Efficient convolution operators for tracking,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6638–6646.

[36] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg, ''Convolutional features for correlation filter based visual tracking,'' in *Proc. IEEE Int. Conf. Comput. Vis. Workshop (ICCVW)*, Dec. 2015, pp. 58–66.

[37] G. Bhat, J. Johnander, M. Danelljan, F. S. Khan, and M. Felsberg, ''Unveiling the power of deep tracking,'' in *Proc. ECCV*, 2018, pp. 483–498.

[38] T. Xu, Z.-H. Feng, X.-J. Wu, and J. Kittler, ''Joint group feature selection and discriminative filter learning for robust visual object tracking,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 7950–7960.

[39] W. Zheng, H. Yu, and Z. Lu, ''Two-step affine transformation prediction for visual object tracking,'' *IEEE Access*, vol. 9, pp. 36512–36521, 2021.

[40] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu, ''Distractor-aware Siamese networks for visual object tracking,'' in *Proc. ECCV*, 2018, pp. 101–117.

[41] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, ''SiamRPN++: Evolution of Siamese visual tracking with very deep networks,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4282–4291.

[42] Z. Zhang and H. Peng, ''Deeper and wider Siamese networks for real-time visual tracking,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4591–4600.

[43] Q. Guo, W. Feng, C. Zhou, R. Huang, L. Wan, and S. Wang, ''Learning dynamic Siamese network for visual object tracking,'' in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1763–1771.

[44] Y. Zhang, L. Wang, J. Qi, D. Wang, M. Feng, and H. Lu, ''Structured Siamese network for real-time visual tracking,'' in *Proc. ECCV*, 2018, pp. 351–366.

[45] H. Fan and H. Ling, ''Siamese cascaded region proposal networks for real-time visual tracking,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7952–7961.

[46] T. Yang, P. Xu, R. Hu, H. Chai, and A. B. Chan, ''ROAM: Recurrently optimizing tracking model,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6718–6727.

[47] Z. Chen, B. Zhong, G. Li, S. Zhang, and R. Ji, ''Siamese box adaptive network for visual tracking,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6668–6677.

[48] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, ''Going deeper with convolutions,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[49] F. Zhang, X. Qian, L. Han, and Y. Shen, ''Inverted residual Siamese visual tracking with feature crossing network,'' *IEEE Access*, vol. 9, pp. 27158–27166, 2021.

[50] Z. Zhou, W. Pei, X. Li, H. Wang, F. Zheng, and Z. He, ''Saliency-associated object tracking,'' in *Proc. ICCV*, Oct. 2021, pp. 9866–9875.

[51] S. Cheng, B. Zhong, G. Li, X. Liu, Z. Tang, X. Li, and J. Wang, ''Learning to filter: Siamese relation network for robust tracking,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 4421–4431.

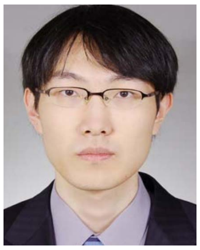
[52] Z. Zhang, Y. Liu, X. Wang, B. Li, and W. Hu, ''Learn to match: Automatic matching network design for visual tracking,'' in *Proc. ICCV*, Oct. 2021, pp. 13339–13348.

[53] B. Yan, H. Peng, K. Wu, D. Wang, J. Fu, and H. Lu, "LightTrack: Finding lightweight neural networks for object tracking via one-shot architecture search," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 15180–15189.
[54] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. NIPS*, 2017, pp. 5998–6008.
[55] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7794–7803.
[56] B. Yan, H. Peng, J. Fu, D. Wang, and H. Lu, "Learning spatio-temporal transformer for visual tracking," in *Proc. ICCV*, Oct. 2021, pp. 10448–10457.
[57] B. Yu, M. Tang, L. Zheng, G. Zhu, J. Wang, H. Feng, X. Feng, and H. Lu, "High-performance discriminative tracking with transformers," in *Proc. ICCV*, Oct. 2021, pp. 9856–9865.
[58] Z. Fu, Q. Liu, Z. Fu, and Y. Wang, "STMTrack: Template-free visual tracking with space-time memory networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 13774–13783.
[59] X. Chen, B. Yan, J. Zhu, D. Wang, X. Yang, and H. Lu, "Transformer tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 8126–8135.
[60] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. ICML*, 2017, pp. 1126–1135.
[61] Z. Li, F. Zhou, F. Chen, and H. Li, "Meta-SGD: Learning to learn quickly for few-shot learning," 2017, *arXiv:1707.09835*.
[62] A. Antoniou, H. Edwards, and A. Storkey, "How to train your MAML," in *Proc. ICLR*, 2019, pp. 1–11.
[63] S. Baik, S. Hong, and K. M. Lee, "Learning to forget for meta-learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 2379–2387.
[64] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," 2018, *arXiv:1803.02999*.
[65] S. Baik, M. Choi, J. Choi, H. Kim, and K. M. Lee, "Meta-learning with adaptive hyperparameters," in *Proc. NeurIPS*, 2020, pp. 1–19.
[66] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "iCaRL: Incremental classifier and representation learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2001–2010.
[67] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks," *Proc. Nat. Acad. Sci. USA*, vol. 114, no. 13, pp. 3521–3526, Mar. 2017.
[68] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 2935–2947, Dec. 2018.
[69] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *Proc. NIPS Workshop*, 2014, pp. 1–9.
[70] K. Shmelkov, C. Schmid, and K. Alahari, "Incremental learning of object detectors without catastrophic forgetting," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3400–3409.
[71] X. Liu, H. Yang, A. Ravichandran, R. Bhotika, and S. Soatto, "Multi-task incremental learning for object detection," 2020, *arXiv:2002.05347*.
[72] S. Hou, X. Pan, C. C. Loy, Z. Wang, and D. Lin, "Lifelong learning via progressive distillation and retrospection," in *Proc. ECCV*, 2018, pp. 437–452.
[73] W. Zhou, S. Chang, N. Sosa, H. Hamann, and D. Cox, "Lifelong object detection," 2020, *arXiv:2009.01129*.
[74] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.
[75] B. Yan, H. Zhao, D. Wang, H. Lu, and X. Yang, "'Skimming-Perusal' tracking: A framework for real-time and robust long-term tracking," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 2385–2393.
[76] X. Dong, J. Shen, L. Shao, and F. Porikli, "CLNet: A compact latent network for fast adjusting Siamese trackers," in *Proc. ECCV*, 2020, pp. 378–395.
[77] Y. Xu, Z. Wang, Z. Li, Y. Yuan, and G. Yu, "SiamFC++: Towards robust and accurate visual tracking with target estimation guidelines," in *Proc. AAAI*, 2020, pp. 12549–12556.
[78] M. Danelljan, L. Van Gool, and R. Timofte, "Probabilistic regression for visual tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 7183–7192.
[79] Y. Zhang, D. Wang, L. Wang, J. Qi, and H. Lu, "Learning regression and verification networks for long-term visual tracking," 2018, *arXiv:1809.04320*.
[80] G. Zhu, F. Porikli, and H. Li, "Beyond local search: Tracking objects everywhere with instance-specific proposals," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 943–951.
[81] R. Tao, E. Gavves, and A. W. M. Smeulders, "Siamese instance search for tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1420–1429.
[82] C. Ma, X. Yang, C. Zhang, and M.-H. Yang, "Long-term correlation tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 5388–5396.
[83] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1409–1422, Dec. 2012.
[84] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 FPS with deep regression networks," in *Proc. ECCV*, 2016, pp. 749–765.
[85] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. S. Torr, "Fast online object tracking and segmentation: A unifying approach," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1328–1338.
[86] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. S. Torr, "End-to-end representation learning for correlation filter based tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2805–2813.
[87] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 3074–3082.
[88] Y. Wu and K. He, "Group normalization," in *Proc. ECCV*, 2018, pp. 3–19.
[89] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and Li Fei-Fei, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[90] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015, pp. 1–15.

JANGHOON CHOI (Member, IEEE) received the B.S. degree in electrical and computer engineering and the Ph.D. degree in electrical engineering and computer science (supervised by Prof. Kyoung Mu Lee) from Seoul National University (SNU), Seoul, South Korea, in 2013 and 2021, respectively. He was a Postdoctoral Researcher at the Automation and Systems Research Institute (ASRI), Seoul National University, from March 2021 to September 2021, under Prof. Kyoung Mu Lee. He is currently an Assistant Professor at the College of Computer Science, Kookmin University, Seoul. His research interests include computer vision problems, including visual tracking and meta-learning applications.

SUNGYONG BAIK (Member, IEEE) received the B.A.Sc. degree in engineering science with a major in electrical and computer engineering from the University of Toronto, Toronto, Canada, in 2015. He is currently pursuing the Ph.D. degree in electrical and computer engineering with Seoul National University. He was a research intern at Meta Reality Labs, in 2019. His research interests include few-shot learning, meta-learning, and its applications.

MYUNGSUB CHOI (Member, IEEE) received the B.S. degree in electrical and computer engineering and the Ph.D. degree in electrical engineering and computer science from Seoul National University (SNU), Seoul, South Korea, in 2013 and 2021, respectively. He was a Research Intern at Snap Inc., in 2018. His research interests include input-adaptive methodologies, such as attention models and meta-learning that are applicable for various computer vision problems including video frame interpolation and super-resolution. He has won the Runner-Up Award at the AIM 2019 Challenge on Video Temporal Super-Resolution.



JUNSEOK KWON (Member, IEEE) received the B.Sc. degree, the M.Sc. degree in object tracking (supervised by Prof. Kyoung Mu Lee), and the Ph.D. degree in electrical engineering and computer science from Seoul National University, Seoul, South Korea, in 2006, 2008, and 2013, respectively. He was a Postdoctoral Researcher, under Prof. Luc Van Gool, with the Computer Vision Laboratory, ETH Zürich, from 2013 to 2014. He is currently an Associate Professor with the School of Computer Science and Engineering, Chung-Ang University, Seoul. He is also working in the field of object tracking to capture the dynamics of cities. His research interests include visual tracking, visual surveillance, and Monte Carlo sampling method and its variants.

KYOUNG MU LEE (Fellow, IEEE) received the B.S. and M.S. degrees in control and instrumentation engineering from Seoul National University (SNU), Seoul, South Korea, in 1984 and 1986, respectively, and the Ph.D. degree in electrical engineering from the University of Southern California, in 1993. He is currently with the Department of ECE, SNU, as a Professor. He is an Advisory Board Member of the Computer Vision Foundation (CVF). He was a Distinguished Lecturer of the Asia-Pacific Signal and Information Processing Association (APSIPA), from 2012 to 2013. He has received several awards, in particular, the Medal of Merit and the Scientist of Engineers of the Month Award from the Korean Government, in 2018 and 2020, respectively; the Most Influential Paper Over the Decade Award by the IAPR Machine Vision Application, in 2009; the ACCV Honorable Mention Award, in 2007; the Okawa Foundation Research Grant Award, in 2006; the Distinguished Professor Award from the College of Engineering of SNU, in 2009; and both the Outstanding Research Award and the Shinyang Engineering Academy Award from the College of Engineering of SNU, in 2010. He has also served as a General Chair for ICCV2019, ACMMM2018, and ACCV2018; a Program Chair for ACCV2012; a Track Chair for ICPR2020 and ICPR2012; and an Area Chair for CVPR, ICCV, and ECCV many times. He has served as an Associate Editor-in-Chief (AEIC) and an Associate Editor for the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (TPAMI); an Associate Editor for the *Machine Vision and Application* (MVA) journal, the *IPSJ Transactions on Computer Vision and Applications* (CVA), and the IEEE SIGNAL PROCESSING LETTERS (SPL); and an Area Editor for the *Computer Vision and Image Understanding* (CVIU). More information can be found on his homepage (http://cv.snu.ac.kr/kmlee).