Cascaded Regression Tracking: Towards Online Hard Distractor Discrimination

Ning Wang, Wengang Zhou, *Senior Member, IEEE*, Qi Tian, *Fellow, IEEE*, and Houqiang Li, *Senior Member, IEEE*

***Abstract*—Visual tracking can be easily disturbed by similar surrounding objects. Such objects as hard distractors, even though being the minority among negative samples, increase the risk of target drift and model corruption, which deserve additional attention in online tracking and model update. To enhance the tracking robustness, in this paper, we propose a cascaded regression tracker with two sequential stages. In the first stage, we filter out abundant easily-identified negative candidates via an efficient convolutional regression. In the second stage, a discrete sampling based ridge regression is designed to double-check the remaining ambiguous hard samples, which serves as an alternative of fully-connected layers and benefits from the closed-form solver for efficient learning. During the model update, we utilize the hard negative mining technique and an adaptive ridge regression scheme to improve the discrimination capability of the second-stage regressor. Extensive experiments are conducted on 11 challenging tracking benchmarks including OTB-2013, OTB-2015, VOT2018, VOT2019, UAV123, Temple-Color, NfS, TrackingNet, LaSOT, UAV20L, and OxUvA. The proposed method achieves state-of-the-art performance on prevalent benchmarks, while running in a real-time speed.**

***Index Terms*—Visual tracking, regression tracking, cascaded framework, hard distractor.**

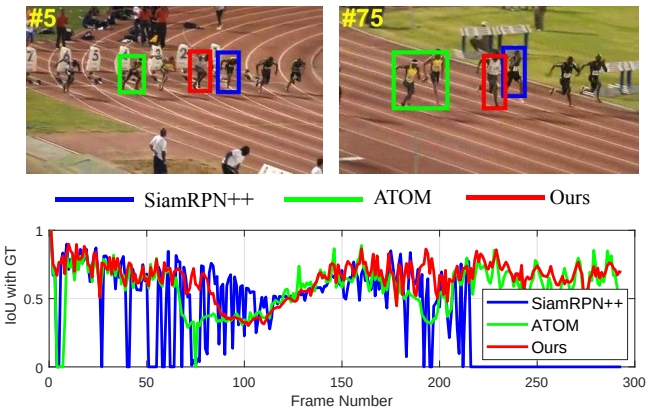


Fig. 1. **Top**: the tracking results of SiamRPN++ [1], ATOM [2], and our approach on the *Bolt2* video. Note that the prior motion model is removed in these methods. **Bottom**: the per-frame overlap between the tracking results and ground-truth bounding box on the *Bolt2*. Without the cosine window, previous methods tend to switch between the target and distractors (e.g., 5-th and 75-th frame), while ours steadily tracks the target without drift.

I. Introduction

As a fundamental task in computer vision, visual object tracking has received lots of attention over the last decades. It plays an important role in many applications such as autonomous driving, robotics, human-computer interaction, etc. In generic visual tracking, the target is arbitrary with only the initial bounding box available. With such limited prior information, the tracker is still highly required to both model the target appearance and distinguish the negative samples on the fly, which is challenging due to the blurry boundary between appearance changes of the target itself and unforeseen similar distractors.

Recently, thanks to the strong representational power of deep CNN models, a simple two-stream template matching based Siamese pipeline [3], [4] has been proved effective in visual tracking, even without the online model update. However, as reported in the Visual Object Tracking (VOT) challenge [5], the robustness of Siamese trackers still has a margin with the discriminative trackers equipped with an update mechanism. In the latest literature [6], [7], substantial attentions have been cast to the updatable deep trackers with superior discrimination capability. Despite the rapid advances, in the tracking and updating stages, how to distinguish similar distractor objects from the target and effectively leverage these hard negative samples to boost the model discrimination capability still leaves exploration space. There exist vast uninformative samples that can be easily distinguished without much effort (i.e., easy sample), while a handful of distracting examples heavily mislead the tracker, enlarging the error accumulation and causing the tracking failures (Figure 1). These unexpectedly emerged distractors, even though being the minority, have a non-trivial effect on degrading the tracking performance, and deserve to be carefully checked *online* for robust tracking.

In this paper, we propose a cascaded regression tracker, which consists of two sequential stages with different regression models for high-performance visual tracking. In the first stage, we employ an efficient convolutional regression [2] to densely predict all the searching locations, which filters out plentiful easy samples. In the second stage, we only consider the remaining ambiguous candidates and propose a discrete sampling based ridge regression for further discrimination.

The work of W. Zhou was supported in part by NSFC under Contract 61632019 and Contract 61822208, and in part by the Youth Innovation Promotion Association CAS under Grant 2018497. The work of H. Li was supported in part by NSFC under Contract 61836011. *(Corresponding authors: Wengang Zhou; Houqiang Li.)*

Ning Wang, Wengang Zhou, and Houqiang Li are with the CAS Key Laboratory of Technology in Geo-spatial Information Processing and Application System, Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei, China.
E-mail: wn6149@mail.ustc.edu.cn, {zhwg, lihq}@ustc.edu.cn.

Qi Tian is with the Huawei cloud & AI, Huawei Technologies.
E-mail: tian.qi1@huawei.com.

The ridge regressor performs as an alternative of the fully-connected layer but exhibits superior efficiency thanks to its closed-form solution. These two stages complement each other as follows. The dense prediction with the convolutional regressor in the first stage [2] covers a large search area, while its model tends to be disturbed by an overwhelming number of easy samples. In contrast, the second-stage regressor trained using the carefully selected hard samples naturally avoids the class-imbalance issue and yields better discrimination on distractors, while its sampling manner fails to perfectly cover the search area and will increase the computational cost when drawing plentiful candidates. By virtue of such a dense-to-discrete and coarse-to-fine two-tier verification, these two stages contribute to a superior robust tracking system. More importantly, both of them allow to update the corresponding models, achieving the *online* adaptability.

During online tracking, to enhance the tracker discrimination, we employ the hard negative mining [8], [9] for the second stage. Moreover, we dynamically reweigh the training samples based on their reconstruction errors in an adaptive ridge regression formula, forcing the second-stage regressor to focus more on valuable samples. Benefited from the high robustness, the second-stage regressor is able to re-detect the lost target when the first stage fails to confidently track the target, and search a large region without excessively worrying about the risk of tracking drift. As a consequence, our framework differs from most existing short-term trackers typically focusing on a limited search region with a prior cosine window to penalize the far-away distractors (e.g., Siamese trackers [3], [10]). It is worth mentioning that our method shows outstanding performance on *both* short-term and long-term tracking datasets without adding additional sophisticated modules thanks to our excellent online discrimination capability.

We summarize the contributions of our work as follows:

- We propose a discrete sampling based ridge regression, which can flexibly absorb the online hard samples and is efficient to learn under a closed-form formula. Furthermore, we propose a cascaded regression tracker, which achieves favorable robustness via a dense-to-discrete large-scale search and a coarse-to-fine two-tire verification.
- To improve the online distractor discrimination, we propose an adaptive ridge regression to further exploit the valuable samples selected by the hard negative mining technique [8], [9]. With the merit of promising discrimination, the second-stage regressor also serves as an effective re-detection module to complement the first stage.
- We extensively evaluate the proposed method on 11 short-term and long-term tracking benchmarks including OTB-2013 [11], OTB-2015 [12], Temple-Color [13], UAV123 [14], VOT2018 [5], VOT2019 [15], NfS [16], TrackingNet [17], LaSOT [18], UAV20L [14], and OxUvA [19]. The proposed approach exhibits state-of-the-art performance on prevalent datasets with a real-time speed.

In the following of the paper, we first survey related works in Section II. Then, we elaborate the proposed cascaded framework in Section III. After that, we evaluate our method with extensive experiments in Section IV. Finally, we conclude this work in Section V.

II. Related Work

In recent years, the Siamese network has gained significant popularity in visual tracking, which deals with the tracking task by searching for the image region most similar to the initial template [3], [4]. The GOTURN algorithm [20] adopts a Siamese pipeline to regress the target bounding box. By introducing the RPN module [10], [21], ensemble learning [22], attention mechanism [23], and target-aware formulation [24], the Siamese trackers gain substantial improvements. Besides visual tracking, similar ideas such as one-shot learning [25] and online adaptation scheme [26] are widely explored in the video object segmentation task. Without video annotations, the unsupervised deep tracking framework is explored in UDT [27]. In [1], SiamRPN++ adopts a deeper backbone network to achieve superior performance. By switching multiple Siamese trackers using an agent network, POST tracker [28] achieves a good balance of accuracy and efficiency. Recently, model update mechanisms are incorporated with the Siamese network [7], [29]–[31]. However, these approaches mainly focus on the template adaptation and still fail to exploit the background context. Since most Siamese trackers ignore the informative negative samples for discrimination enhancement, they tend to drift when similar distractors appear. Recently, the cascaded framework has been investigated within the Siamese tracking framework [32], [33]. SPM [33] combines the SiamRPN with a relation network to further classify the candidates. C-RPN [32] utilizes cascaded region proposal networks for accurate target localization. Nevertheless, they do not involve the online model update. The overlook of online emerged samples heavily limits the performance. In other words, how to take advantage of the hard negative samples to distinguish potential distractors in future frames is ignored in the recent cascaded frameworks. Compared with them, the main distinction of this work is that our cascaded framework is built on two complementary regression models, both of which are able to absorb the online samples for the persistent model update.

Another popular tracking family is the regression based approach, which generally regresses a large Region of Interest (RoI) to a response map for target localization. The Correlation Filter (CF) solves the ridge regression in the Fourier domain, showing extremely attractive efficiency [34]–[43]. To alleviate the unwanted boundary effect, regularization terms [44]–[46] and background-aware formulation [47] are proposed. ECO tracker [48] introduces a factorized convolution operator, a generative sample space model, and the sparse update strategy to further boost the efficiency of correlation tracking. Recently, by jointly compressing and transferring the heavyweight feature extractors in deep CF trackers, CPU real-time efficiency is also feasible [49]. Besides CF, with the recent astonishing development of deep learning, convolutional regression gains an increasing attention in visual tracking [2], [6], [50], [51]. In these approaches, a CNN kernel is learned to convolve with the RoI feature for response generation, which effectively avoids

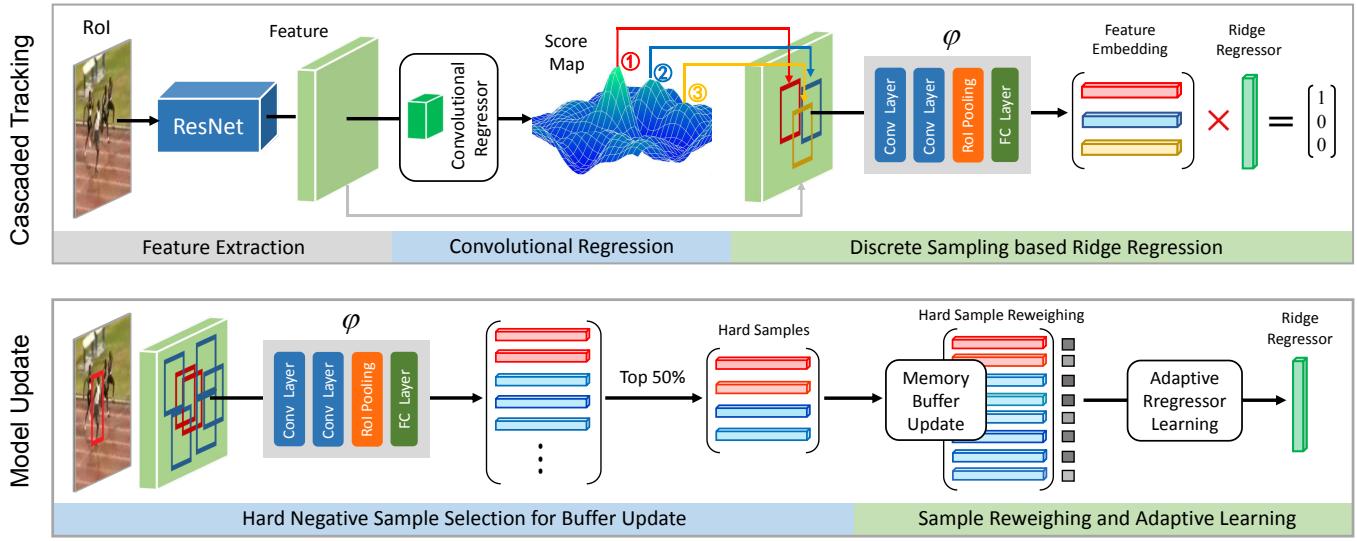


Fig. 2. **Top**: an overview of our cascaded regression tracker. In the first stage, we employ a convolutional regression for dense response prediction. In the second stage, a discrete sampling based ridge regression is designed to discriminate the ambiguous candidates. **Bottom**: online model update for the second-stage regressor. Based on the tracking result, only hard negative samples are selected to update the memory buffer. Besides, the training samples in the buffer are dynamically reweighed for adaptive regressor learning.

the boundary effect in CF. Unfortunately, this convolutional formulation does not have a closed-form solution, and needs the gradient back-propagation to learn the filter. Besides, the large RoI size in the regression approach brings in the class-imbalance issue. In CREST [50], residual terms are incorporated into the convolutional regression to cope with the target appearance changes. DSLT [51] introduces shrinkage loss to balance the training samples in the convolutional regression. To accelerate the kernel learning process, ATOM [2] exploits the conjugate gradient in the deep learning framework. The recent DiMP approach [6] proposes an iteratively optimized discriminative model for classification and trains the whole framework in an end-to-end manner. Despite the recent progress, the discrimination capability in regression trackers, especially for hard distractors, still leaves room for improvement.

In contrast to the aforementioned regression methods that generate a dense prediction, previous discriminative trackers learn a binary classifier to classify the discretely sampled candidates for tracking (e.g., MDNet [52]). In spite of their shallow backbone networks and limited discrete samples (e.g., 256 candidates per frame), by an effective model update with hard negative mining, these approaches [52]–[54] still exhibit impressive robustness on various tracking benchmarks, suggesting the importance of online learning.

Our proposed approach is partially inspired by the above observations to retain both the dense and discrete predictions in a coarse-to-fine manner. Hard negative mining, as a powerful technique in object detection [8], [9], has been successfully equipped into some discrete sampling based visual trackers such as MDNet [52]. However, existing regression based trackers fail to effectively explore the hard negative samples since they train the regression model using densely sampled candidates and generally equally weigh them. The recent ATOM tracker [2] reduces the training weights of easy samples to focus on the valuable negative samples to some extent, but we observe that it still struggles to distinguish hard distractors. In this work, our first stage densely searches a large RoI to generate high-quality proposals, while the second stage is more flexible in the model update and hard negative mining to better distinguish the hard negative samples. Even though aiming at predicting discrete samples, unlike [52]–[54] that leverage fully-connected layers for classification, we learn an efficient closed-form solver in the feed-forward pass without back-propagation, potentially alleviating the overfitting issue due to much fewer parameters to be optimized online. By design, we absorb the strength of both regression trackers and discrete sampling based tracking-by-detection approaches to form a unified cascaded tracking framework. Our method is also motivated by the two-stage framework in object detection (e.g., faster RCNN [55]), which has witnessed tremendous success in recent years. Differently, we exploit two regression models specially designed for the online tracking task with an incremental model update.

III. Methodology

In Figure 2 (top), we show an overview of the proposed cascaded tracker. In the first stage, a convolutional regressor densely predicts the target location over a large RoI. Then, the ambiguous proposals are fed to the second regression stage for further discrimination. Under such a dense-to-discrete and coarse-to-fine verification, the proposed tracking framework achieves favorable tracking robustness. In Figure 2 (bottom), we exhibit the online update process of the second-stage regression model. By virtue of the hard negative mining and an adaptive ridge regression formulation, the learned regressor is readily ready for distinguishing hard distractors.

In the following, we first review the regression based tracking in Section III-A for the sake of completeness. In

Section III-B, we present our discrete sampling based ridge regression and provide a detailed analysis in comparison with the previous methods. Then, in Section III-C, we depict the cascaded regression tracking and re-detection mechanism. Finally, we introduce the details of the online model update in Section III-D.

A. Revisiting Regression Tracking

In this subsection, we briefly review the correlation filter and convolutional regression.

Correlation Filter. The correlation filter (CF) [34], [35] tackles visual tracking by solving the following regression problem:

$$\min_{\mathbf{W}_{\mathrm{CF}}} \|\mathbf{X} \star \mathbf{W}_{\mathrm{CF}} - \mathbf{Y}_{\mathrm{G}}\|_2^2 + \lambda \|\mathbf{W}_{\mathrm{CF}}\|_2^2, \tag{1}$$

where $\star$ denotes the circular correlation, λ is a regularization parameter that controls overfitting, $\mathbf{X} \in \mathbb{R}^{M \times N \times C}$ is the feature map of the RoI patch, $\mathbf{Y}_{\mathrm{G}} \in \mathbb{R}^{M \times N}$ is the Gaussian-shaped label, and $\mathbf{W}_{\mathrm{CF}} \in \mathbb{R}^{M \times N \times C}$ is the desired correlation filter.

Let $\mathbf{A}$ denote the data matrix that contains all the circulant shifts of the base feature representation $\mathbf{X}$. Then, the circular correlation $\mathbf{X} \star \mathbf{W}_{\mathrm{CF}}$ is equal to $\mathbf{A}\mathbf{W}_{\mathrm{CF}}$, and the filter $\mathbf{W}_{\mathrm{CF}}$ has the following closed-form solution [34], [56], [57]:

$$\mathbf{W}_{\mathrm{CF}} = (\mathbf{A}^{\mathrm{T}}\mathbf{A} + \lambda \mathbf{I})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{Y}_{\mathrm{G}}, \tag{2}$$

where $\mathbf{I}$ is the identity matrix. Due to the circulant structure of $\mathbf{A}$, it can be diagonalized via $\mathbf{A} = \mathbf{F}\ \mathrm{diag}(\hat{\mathbf{X}})\ \mathbf{F}^{\mathrm{H}}$, where $\hat{\mathbf{X}}$ is the Discrete Fourier Transform (DFT) of $\mathbf{X}$, $\mathbf{F}$ is the DFT matrix and $\mathbf{F}^{\mathrm{H}}$ is the Hermitian transpose of $\mathbf{F}$. Therefore, Eq. 2 results in a very efficient element-wise multiplication solution in the Fourier domain without matrix inversion. Please refer to [34] for more details.

Convolutional Regression. The convolutional regression [2], [50], [51] considers the following minimization problem:

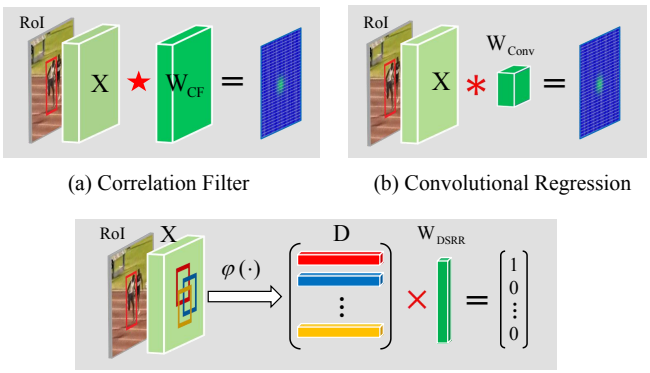
$$\min_{\mathbf{W}_{\mathrm{Conv}}} \|\mathbf{X} * \mathbf{W}_{\mathrm{Conv}} - \mathbf{Y}_{\mathrm{G}}\|_2^2 + \lambda \|\mathbf{W}_{\mathrm{Conv}}\|_2^2. \tag{3}$$

Different from the circular correlation in Eq. 1, the $*$ operation in Eq. 3 denotes the standard multi-channel convolution, which is the core component in CNNs.

Without a closed-form formula, the solution of Eq. 3 can be optimized via the standard gradient descent as follows:

$$\mathbf{W}_{\mathrm{Conv}}^{i+1} = \mathbf{W}_{\mathrm{Conv}}^{i} - \alpha \nabla \mathcal{L}(\mathbf{W}_{\mathrm{Conv}}^{i}), \tag{4}$$

where α is the learning rate of the gradient descent and $\mathcal{L}(\cdot)$ denotes regression error presented in Eq. 3. Given the feature map $\mathbf{X} \in \mathbb{R}^{M \times N \times C}$, the learned filter (or convolutional kernel) $\mathbf{W}_{\mathrm{Conv}} \in \mathbb{R}^{m \times n \times C}$ regresses the feature map $\mathbf{X}$ to the desired Gaussian label $\mathbf{Y}_{\mathrm{G}}$. Note that the correlation filter $\mathbf{W}_{\mathrm{CF}}$ in Eq. 1 has the same spatial size with $\mathbf{X}$, while the convolutional filter $\mathbf{W}_{\mathrm{Conv}}$ requires to be smaller than $\mathbf{X}$, i.e., $m < M,\ n < N$, as shown in Figure 3 (b).



(a) Correlation Filter (b) Convolutional Regression

(c) Discrete Sampling based Ridge Regression

Fig. 3. Illustration of correlation filter, convolutional regression, and our discrete sampling based ridge regression.

B. Discrete Sampling based Ridge Regression

In the CF and convolutional regression, the learned filters regress the RoI to a dense response map. This continuous prediction generally brings in the class-imbalance issue [51], where plentiful uninformative samples will overwhelm the valuable ones in the filter training. Actually, there is no need to limit ourselves to the dense prediction in a regression scheme. To focus on the hard samples, we propose a simple, flexible yet effective Discrete Sampling based Ridge Regression (DSRR). The *discrete* lies in two aspects: (1) The training data are sampled discretely (Figure 3 (c)), which is similar to the classic classification based tracking approach [52]. By carefully selecting the training samples, the learned filter pays more attention to the hard negative samples and naturally avoids the class-imbalance issue. (2) The label is discrete (binary) instead of the soft Gaussian shape, which introduces the label margin between positive and hard negative samples. As shown in Figure 3 (c), the learned discrete ridge regressor can be interpreted as a fully-connected layer with a single node, but provides a fast solution in a single pass to learn the model instead of learning with time-consuming back-propagation.

To train this regressor, we represent each sample by a high-dimensional feature embedding via a CNN mapping function $\varphi(\mathbf{X}, \mathbf{B}_i)$, whose inputs consist of the base feature map $\mathbf{X}$ and the i-th sample's bounding box $\mathbf{B}_i \in \mathbb{R}^4$. These training samples are discretely sampled with binary labels, representing the target or background. As shown in Figure 2, the mapping function $\varphi(\cdot)$ first refines the backbone feature $\mathbf{X}$ through two convolutional layers, and further generates the feature embedding via an RoI pooling operation followed by a fully-connected layer. Then we assemble these feature embeddings to form the data matrix $\mathbf{D} = [\varphi(\mathbf{X}, \mathbf{B}_1), \cdots, \varphi(\mathbf{X}, \mathbf{B}_P)]^{\mathrm{T}} \in \mathbb{R}^{P \times L}$, which contains P embeddings and each of them is L-dimensional. Based on the overlap ratios between candidates' boxes $\mathbf{B}$ and ground-truth box, these feature embeddings are assigned by positive or negative labels. Leveraging data matrix $\mathbf{D}$ and its binary label $\mathbf{Y}_{\mathrm{B}}$, the discrete sampling based ridge regressor $\mathbf{W}_{\mathrm{DSRR}}$ can be obtained by solving the following minimization problem:

$$\min_{\mathbf{W}_{\mathrm{DSRR}}} \|\mathbf{D}\mathbf{W}_{\mathrm{DSRR}} - \mathbf{Y}_{\mathrm{B}}\|_2^2 + \lambda \|\mathbf{W}_{\mathrm{DSRR}}\|_2^2, \tag{5}$$

where $\mathbf{Y}_{\mathrm{B}} \in \mathbb{R}^{P}$ is the binary label.

Primal Domain. Since Eq. 5 still follows the standard ridge regression, similar to Eq. 2, it has the closed-form solution $\mathbf{W}_{\mathrm{DSRR}} = (\mathbf{D}^{\mathrm{T}}\mathbf{D}+\lambda\mathbf{I})^{-1}\mathbf{D}^{\mathrm{T}}\mathbf{Y}_{\mathrm{B}}$. Compared with the solution to CF, the main advantage is that this data matrix $\mathbf{D}$ no longer contains fake (cyclically shifted) samples, while the tradeoff is that the Fourier domain solution becomes unfeasible. In the above solution, the main computational burdern lies in the matrix inverse, whose time complexity is $O(L^3)$ for the matrix $(\mathbf{D}^{\mathrm{T}}\mathbf{D} + \lambda\mathbf{I}) \in \mathbb{R}^{L\times L}$.

Dual Domain. Eq. 5 can also be solved in the dual domain, where the regressor $\mathbf{W}_{\mathrm{DSRR}}$ is expressed by a linear combination of the samples, i.e., $\mathbf{W}_{\mathrm{DSRR}} = \mathbf{D}^{\mathrm{T}}\boldsymbol{\alpha}$. The variables under optimization are thus $\boldsymbol{\alpha}$ instead of $\mathbf{W}_{\mathrm{DSRR}}$. The dual variables $\boldsymbol{\alpha}$ can be solved by $\boldsymbol{\alpha} = (\mathbf{D}\mathbf{D}^{\mathrm{T}} + \lambda\mathbf{I})^{-1}\mathbf{Y}_{\mathrm{B}}$ [56]. Therefore, the ridge regressor can be computed in the dual domain as follows:

$$\mathbf{W}_{\mathrm{DSRR}} = \mathbf{D}^{\mathrm{T}}\boldsymbol{\alpha} = \mathbf{D}^{\mathrm{T}}(\mathbf{D}\mathbf{D}^{\mathrm{T}} + \lambda\mathbf{I})^{-1}\mathbf{Y}_{\mathrm{B}}. \tag{6}$$

Since $(\mathbf{D}\mathbf{D}^{\mathrm{T}} + \lambda\mathbf{I}) \in \mathbb{R}^{P\times P}$, the matrix inverse in Eq. 6 has the time complexity of $O(P^3)$ instead of $O(L^3)$ in primal domain, which relates to the sample number P instead of feature dimension L. Thanks to the limited number of hard examples, a small K is generally practicable. While in case of a low feature dimension L, the primal domain solution will be more efficient. Overall, depending on the sizes of L and P, we can always find a good efficiency balance between the primal and dual solutions.

Offline Training. In the training stage, we aim to learn a CNN function $\varphi(\cdot)$ to ensure the learned feature representation suitable for the designed ridge regression. To this end, we adopt a Siamese-like pipeline in the training stage, where the template branch is utilized to learn the ridge regressor while the search branch is used to generate plentiful test candidates for loss computation. In the large Region of Interest (RoI), we randomly draw plentiful samples. The positive and negative samples are collected following the ratio of 1 : 3, which have $\geqslant 0.7$ and $\leqslant 0.5$ overlap ratios with ground-truth bounding boxes, respectively. In our experiment, the total sample number is 400 in each frame, i.e., 100 positive samples and 300 negative samples.

Instead of using the prototype $\varphi(\cdot)$ in Figure 2 for simplicity, to achieve better performance, we exploit the multi-scale feature representations from both Block3 and Block4 of the ResNet-18 [58] as the inputs of two individual $\varphi(\cdot)$ networks. The Precise RoI Pooling (PrPool) [59] is utilized in $\varphi(\cdot)$ to crop the Block3 and Block4 features, whose output sizes are 5×5 and 3×3, respectively. The following fully-connected layer maps the pooled features to a 256-dimensional feature vector. Finally, the Block3 and Block4 feature vectors are concatenated along the channel dimension as the 512-dimensional output feature embedding.

Thanks to the closed-form solution of ridge regression, it can be embedded as a differentiable layer for end-to-end training. Leveraging the regressor $\mathbf{W}_{\mathrm{DSRR}}$ learned via template branch, the regression scores of the test candidates in the search branch can be calculated by $\mathbf{Y}_{\mathrm{Test}} = \mathbf{D}_{\mathrm{Test}}\mathbf{W}_{\mathrm{DSRR}}$. To train the network $\varphi(\cdot)$, we adopt the standard L_2 loss as the training objective: $\ell = \|\mathbf{Y}_{\mathrm{Test}} - \mathbf{Y}_{\mathrm{GT}}\|_2^2$, where $\mathbf{Y}_{\mathrm{GT}}$ is the ground-truth binary label of the test samples. After offline training, $\varphi(\cdot)$ is fixed in the tracking stage.

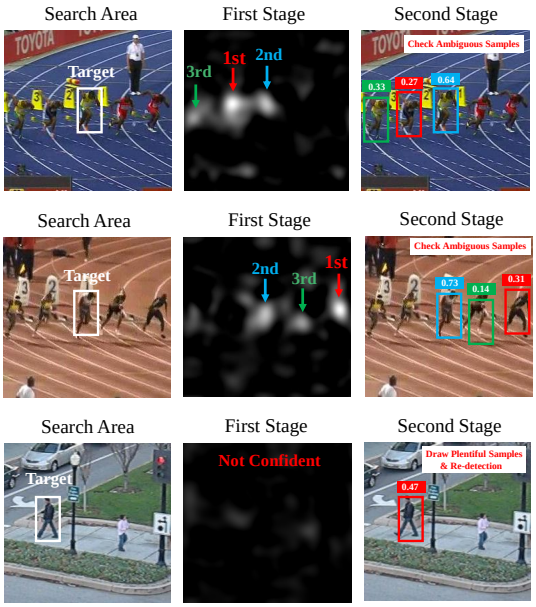


Fig. 4. Tracking examples of the proposed cascaded framework. In the first stage, we select top three peaks as the high-quality proposals, which are further checked via the second stage. If the first stage fails to predict confidently, we draw plentiful candidates in the second stage for target re-detection.

Connection with Related Methods. We compare the CF, convolutional regression, and our discrete sampling based ridge regression in the following 4 aspects. (1) **Efficiency.** Convolutional regression typically requires gradient back-propagation to learn the filter. CF exploits the closed-form solution in the Fourier domain, showing extremely attractive efficiency. The proposed DSRR also has a closed-form solution, yielding satisfactory efficiency. (2) **Label.** Both CF and convolutional regression predict dense response scores. In contrast, our approach considers discrete proposals, which is flexible to focus on the hard examples and eliminate the class-imbalance issue. (3) **Effectiveness.** The performance of CF is heavily limited by the boundary effect, i.e., the data matrix $\mathbf{A}$ consists of plentiful fake samples. In contrast, the convolutional regression and our DSRR are learned using *real* samples. (4) **Flexibility.** The CF can only detect the RoI with a fixed size (Figure 3). In contrast, the convolutional regression and DSRR are more flexible, which can be applied to the RoI of any size and explore a larger area when necessary (e.g., target out-of-view). Considering the above characteristics, we choose the convolutional regression and discrete ridge regression as the first and second stages in our approach, respectively.

Our discrete ridge regression also shares partial similarity with the classification based approach (MDNet [53]). The main distinction is that we learn a closed-form solver to *regress* the proposals instead of leveraging several fully-connected (FC) layers to *classify* them, which is much more efficient via a

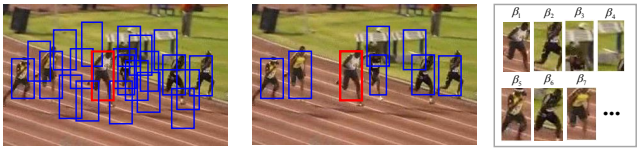


Fig. 5. An illustration of the hard negative mining and sample reweigh for the proposed adaptive ridge regression learning.

feed-forward computation without back-propagation to update the FC parameters.

C. Online Tracking

Cascaded Regression Tracking. Before tracking, we first learn the aforementioned two regressors using the initial frame. For the first stage, instead of adopting the stochastic gradient descent (SGD) to learn the convolutional filter, we follow Danelljan *et al.* [2] to use Newton-Gaussian descent as the optimization strategy for fast convergence, and learn a 4×4 kernel to regress the Gaussian response map. To learn the second-stage regressor, based on the initial ground-truth label, we crop the positive and negative samples following a ratio of 1 : 3 to form the data matrix. Then, the discrete ridge regressor $\mathbf{W}_{\text{DSRR}}$ is obtained by the primal or dual solution, depending on the sample number and feature dimension. In the initial few frames, the sample number is smaller than the feature dimension (i.e., $P < L$), and we choose the dual domain. With the arrival of new frames, if $P > L$, we switch to the primal domain.

During online tracking, in each frame, top-3 peaks in the first stage's score map are selected, and the corresponding proposals are fed to the network $\varphi(\cdot)$ to generate the feature embeddings. These ambiguous proposals are further checked by the second-stage regressor, as shown in Figure 4. Finally, we equally combine the prediction scores of these two stages, and select the highest proposal as the current target. After target localization, we utilize the IoU predictor proposed in [2] to further refine the target scale.

Cascaded Re-detection. As a common strategy in many visual trackers [2], [6], [52], we set two reliability thresholds τ_1 and τ_2 for the two regressors, respectively. In case that the first stage cannot confidently predict the target, i.e., the highest response score is lower than τ_1, we sample abundant candidates (512 per frame) and leverage the second-stage regressor for re-detection, as shown in Figure 4. If the confidence score of the re-detected target exceeds τ_2, we regard it as the target. Otherwise, we keep the target position as in the previous frame. Since the backbone features are shared, this re-detection process and the following model update only involve a slight computational burden.

D. Online Model Update

Hard Negative Mining. Model update is the core component for discriminating the online distractors. To alleviate the corruption of the memory buffer, we only collect the training samples in reliable frames. Here, a *reliable* frame represents that both two regressors predict confidently, i.e., their estimated scores exceed τ_1 and τ_2, respectively.

Algorithm 1 Online Model Update

Input: Video sequence and initial ground-truth.
Initialize two regressors $\mathbf{W}_{\text{Conv}}$ and $\mathbf{W}_{\text{DSRR}}$.
Initialize the reliable frame buffer $\mathcal{T} = \{1\}$.
for $t = 2$ **to** N **do**
  Conduct cascaded regression tracking; ▷ Section 3.3
  if current result is reliable **then**
    Collect the current RoI region $\mathcal{R}_t$; ▷ first stage
    Draw pos/neg samples $\mathcal{S}_t^+$ and $\mathcal{S}_t^-$; ▷ second stage
    Drop 50% easy negative samples from $\mathcal{S}_t^-$;
    $\mathcal{T} \leftarrow \mathcal{T} \cup \{t\}$; ▷ merge the reliable frame
    if $|\mathcal{T}| > \gamma$ **then** ▷ maintain a fixed buffer size
      $\mathcal{T} \leftarrow \mathcal{T} \setminus \{\min_{k\in\mathcal{T}} k\}$; ▷ drop the oldest index
    end if
  end if
  if t mod 10 == 0 **then** ▷ sparse model update
    Update $\mathbf{W}_{\text{Conv}}$ using $\mathcal{R}_{k\in\mathcal{T}}$; ▷ first stage
    Update $\mathbf{W}_{\text{DSRR}}$ using $\mathcal{S}^{+/-}_{k\in\mathcal{T}}$; ▷ second stage
  end if
end for

The first stage is incrementally updated by the Gauss-Newton descent using newly collected RoI samples following [2]. The second stage is expected to distinguish ambiguous samples. To this end, we discretely draw two times of the desired negative samples and select only half of them with a high regression score, as shown in Figure 5. These hard training samples are added to the buffer for the model update.

Adaptive Ridge Regression. For the second stage, under consistent model update, the ambiguity degrees of different training samples dynamically change. Therefore, we further assign a weight β_i to each training sample $\varphi(\mathbf{X}, \mathbf{B}_i)$ in the memory buffer. As a result, the discrete ridge regression is re-formulated as follows:

$$\min_{\mathbf{W}_{\text{DSRR}}} \sum_i \beta_i \|\varphi(\mathbf{X}, \mathbf{B}_i)\mathbf{W}_{\text{DSRR}} - y_i\|_2^2 + \lambda \|\mathbf{W}_{\text{DSRR}}\|_2^2. \quad (7)$$

By defining a weight matrix $\mathbf{M} = [\sqrt{\beta_1}, \cdots, \sqrt{\beta_P}]^{\text{T}}$, Eq. 7 can be converted into the matrix form as follows:

$$\min_{\mathbf{W}_{\text{DSRR}}} \|\mathbf{M} \odot \mathbf{D}\mathbf{W}_{\text{DSRR}} - \mathbf{M} \odot \mathbf{Y}_{\text{B}}\|_2^2 + \lambda \|\mathbf{W}_{\text{DSRR}}\|_2^2. \quad (8)$$

As a result, the the solution to Eq. 7 can be computed by

$$\mathbf{W}_{\text{DSRR}} = (\widetilde{\mathbf{D}}^{\text{T}}\widetilde{\mathbf{D}} + \lambda \mathbf{I})^{-1}\widetilde{\mathbf{D}}^{\text{T}}\widetilde{\mathbf{Y}_{\text{B}}} = \widetilde{\mathbf{D}}^{\text{T}}(\widetilde{\mathbf{D}}\widetilde{\mathbf{D}}^{\text{T}} + \lambda I)^{-1}\widetilde{\mathbf{Y}_{\text{B}}}, \quad (9)$$

where $\widetilde{\mathbf{D}} = \mathbf{M} \odot \mathbf{D}$, $\widetilde{\mathbf{Y}_{\text{B}}} = \mathbf{M} \odot \mathbf{Y}_{\text{B}}$, and $\odot$ is the element-wise product. We empirically define the weight matrix $\mathbf{M}$ as the reconstruction error of the sample label by previous ridge regressor, as follows:

$$\mathbf{M} = \text{norm}\left(\mathbf{Y}_{\text{B}} - \mathbf{D}\mathbf{W}_{\text{DSRR}}^{t-1}\right) \cdot P, \quad (10)$$

where $\mathbf{W}_{\text{DSRR}}^{t-1}$ is the ridge regressor in the previous frame, $\text{norm}(\mathbf{x}) = \mathbf{x}/\|\mathbf{x}\|_1$ denotes L_1 normlization, and P is the

total sample number in the data matrix. Intuitively, Eq. 10 normalizes the reconstruction errors of different samples and then rescales the weights to ensure the summation of $\mathbf{M}$ equals to P. A large prediction error means the corresponding sample performs as a hard one for the previously learned $\mathbf{W}_{\text{DSRR}}^{t-1}$, which deserves more attention in the current learning.

In our experiments, a new discrete regressor is learned every 10 frames, and is updated to the previous model in a moving average manner: $\mathbf{W}_{\text{DSRR}}^{t} = (1-\eta)\mathbf{W}_{\text{DSRR}}^{t-1} + \eta\mathbf{W}_{\text{DSRR}}$. An overview of the above model update process is presented in Algorithm 1.

IV. Experiments

A. Implementation Details

In offline training, we freeze all the weights of the backbone network (ResNet-18 [58]) and adopt a multi-task training strategy to train the network $\varphi(\cdot)$ and IoU predictor. Note that the inputs of IoU predictor and ridge regression are different. Following ATOM [2], the IoU predictor leverages the samples that have a certain overlap with the ground-truth box (at least 0.1). In contrast, our ridge regression branch utilizes the aforementioned positive and negative samples to learn the discriminative model. The input RoI region is 5 times of the target size and is further resized to 288×288. We utilize the training splits of LaSOT [18], TrackingNet [17], GOT-10k [60], and COCO [61] for offline training. The model is trained for 50 epochs with 1000 iterations per epoch and 36 image pairs per batch. The ADAM optimizer [62] is employed with an initial learning rate of 0.01, and use a decay factor 0.2 for every 15 epochs. The first-stage regressor uses ResNet-18 Block3 features as in ATOM, while the second-state regressor and the IoU predictor takes both Block3 and Block4 backbone features as input. In online tracking, to update the second-stage regressor, we collect 30 positive and 90 hard negative samples per reliable frame, and maintain a buffer for the last 30 frames. The learning rate η of the second stage is 0.2. The reliability thresholds τ_1 and τ_2 are set to 0.25 and 0.4, respectively.

We denote our CAscaded REgression method as CARE in the following experiments. Our tracker is implemented in Python using PyTorch, and operates about 25 frames per second (FPS) on a single Nvidia GTX 1080Ti GPU. We evaluate our method on each benchmark 3 times and report the average performance.

B. Ablation Experiments

We utilize the OTB-2015 [12], UAV123 [14], and LaSOT testing set [18], with total 503 videos, to comprehensively verify the effectiveness of our framework.

Cascaded Framework. In Table I, we compare the performance of each single stage and their cascaded combination. Note that we draw 512 samples per frame if the second stage is tested alone, aiming to obtain satisfactory performance. From Table I, we can observe that the first and second stages almost perform identically. The main reason is that the convolutional regression is not discriminative enough, while the discrete sampling strategy fails to well cover a large search region. By combining them in a cascaded manner, superior performance can be obtained. For example, on OTB-2015, our final cascaded tracker outperforms the first and second stages by 3.0% and 3.1%, respectively. On the recent large-scale dataset LaSOT, our final framework surpasses the first stage by 3.0% in AUC. Note that the first stage in our framework is adopted from the ATOM, which already achieves a high performance level on various challenging datasets. Under the same backbone network and bounding box regression manner (i.e., IoUNet), our performance gains can be attributed to the superior discrimination capability of our cascaded framework. As for the tracking speed, with the abundant candidates (512 samples per frame), the second-stage regressor is less efficient than the first stage. In contrast, our cascaded framework achieves a balanced speed and outstanding performance, which only slightly reduces the first-stage efficiency but notably outperforms it in tracking accuracy.

TABLE I
Analysis of each component in our method. We first compare the performance of the first stage, second stage and their cascaded combination. Then, we analyze the second stage by adding re-detection mechanism and adaptive ridge regression (ADRR). The performance is verified on OTB-2015, UAV123, and LaSOT in terms of the area-under-curve (AUC) score of success plot.

First Stage	Second Stage	Re-detection	ADRR	OTB-2015 [12]	UAV123 [14]	LaSOT [18]	Speed FPS
✓				67.5	63.3	51.7	**30**
	✓			68.0	62.1	49.3	20
✓	✓			69.2	64.4	53.7	27
✓	✓	✓		69.5	65.0	54.1	25
✓	✓	✓	✓	**70.5**	**65.4**	**54.7**	25

Target Re-detection. As discussed in Section III-C, our second-stage regressor also acts as a re-detection module due to its high discrimination. As shown in Table I, with additional performance improvements, the re-detection mechanism further exploits the potential of the second stage.

Adaptive Ridge Regression. Online model update plays a vital role in our framework. Based on the collected hard samples in the memory buffer, to better concentrate on the valuable ones, we propose an adaptive ridge regression that dynamically reweighs the training samples. As illustrated in Table I, our adaptive ridge regression (ADRR) steadily improves the tracking accuracy. Besides, it is worth mentioning that our ADRR is extremely efficient with a negligible computational cost.

C. Comparison with State-of-the-art Methods

We compare our proposed CARE tracker with the recent state-of-the-art trackers on 11 challenging tracking benchmarks including OTB-2013 [11], OTB-2015 [12], UAV123 [14], LaSOT [18], VOT2018 [5], VOT2019 [15], TrackingNet [17], Temple-Color [13], UAV20L [14], Need for Speed [16], and OxUvA [19].

OTB-2013 [11]. OTB-2013 is a widely evaluated tracking dataset with 50 videos. Figure 6 (left) shows the success plot on the OTB-2013. On this dataset, our method achieves an

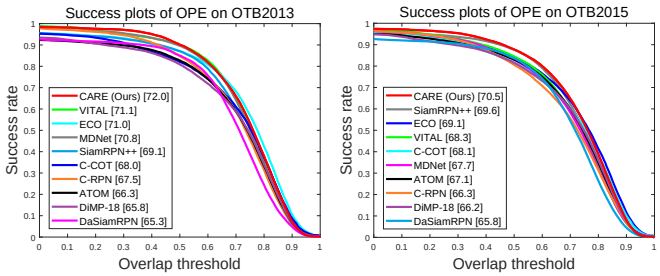


Fig. 6. Success plots on the OTB-2013 [11] (left) and OTB-2015 [12] (right) datasets. The legend shows the AUC score. The proposed CARE method outperforms all the comparison trackers.

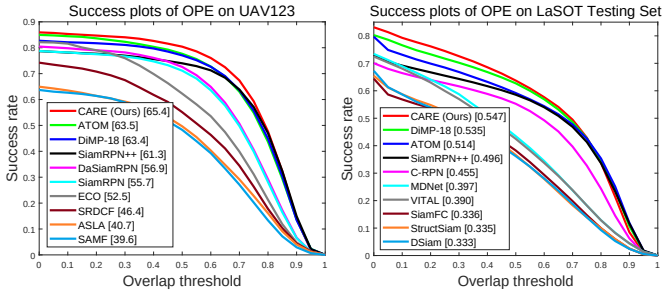


Fig. 7. Success plots on the UAV123 [14] (left) and LaSOT [18] (right) datasets. The legend shows the AUC score. The proposed CARE method outperforms all the comparison trackers.

AUC score of 72.0%, outperforming all previous state-of-the-art trackers such as VITAL [54] and ECO [48]. Note that the top-performing trackers on this benchmark cannot operate at a real-time speed, e.g., the speeds of VITAL and MDNet are only 1 FPS by using fully-connected layers for candidate classification, while ours is real-time since our closed-form regressor is free of gradient back-propagation. Compared with other state-of-the-art trackers with the same ResNet backbone (e.g., ATOM), our approach exhibits competitive efficiency with a speed of about 25 FPS.

OTB-2015 [12]. OTB-2015 benchmark extends OTB-2013 with additional 50 videos, resulting in 100 videos in total. Figure 6 (right) shows the success plot over 100 videos on the OTB-2015. Our method achieves an AUC score of 70.5% on this benchmark, surpassing the recently proposed SiamRPN++ [1], ECO [48], and VITAL [54] trackers. Compared with the recent single-stage regression trackers such as ATOM [2] and DiMP-18 [6], our CARE method outperforms them by 3.4% and 4.3% in terms of AUC score, respectively. Note that DiMP-18 is the recently proposed regression method with discriminative model learning, which represents the state-of-the-art performance on several datasets.

UAV123 [14]. This dataset includes 123 aerial videos collected by a low-attitude UAV platform. Therefore, UAV123 focuses on evaluating visual trackers in the UAV scenarios with small and fast-moving targets. Figure 7 (left) illustrates the success plot of the state-of-the-art trackers including SiamRPN++, ATOM, and DiMP-18. Compared with the recent remarkable approaches, our method achieves the best result. Especially, our approach shows an AUC score of 65.4%, outperforming SiamRPN++, ATOM, and DiMP-18 by 4.1%, 1.9%, and 2.0% AUC score, respectively. Since our approach is equipped with the same backbone network and IoU predictor compared with ATOM and DiMP-18, our performance advantage verifies the superiority of the proposed cascaded tracking framework.

LaSOT [18]. LaSOT is a recent large-scale tracking dataset including 1200 videos, which is more challenging than the previous short-term benchmarks with an average of 2500 frames per video. We evaluate our approach on the test set of 280 videos. Except for the top-performing trackers like MDNet and VITAL on this dataset, we also include the recent C-RPN [32], SiamRPN++, ATOM, and DiMP-18 for comparison. The success plot on LaSOT is shown in Figure 7 (right). On

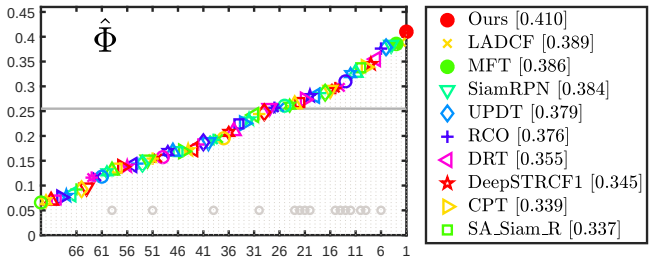


Fig. 8. Expected average overlap (EAO) graph with trackers ranked from right to left. Our method obviously outperforms all the participants on the VOT2018 [5].

this dataset, our approach achieves an AUC score of 54.7%, outperforming the previous best method on this benchmark (i.e., MDNet) by a considerable margin of 15.0% AUC score. Compared with the recent C-RPN, SiamRPN++, ATOM, and DiMP-18, our CARE surpasses them by 9.2%, 5.1%, 3.3%, and 1.2% in AUC, respectively.

In Figure 9, we further provide the attribute evaluation on the LaSOT benchmark [18]. On this large-scale dataset, our approach shows good results on fast motion, out-of-view, and viewpoint change. On the above attributes, our method even surpasses the recently remarkable DiMP-18 tracker [6] by a large margin, which can be attributed to the strong discrimination of our second stage. Our second-stage regressor further checks the ambiguous candidates and serves as a re-detection module, which significantly improves the tracking performance on the challenging scenarios such as fast motion, out-of-view, and target occlusion. Besides, our method outperforms its baseline method ATOM [2] in all attributes on the LaSOT dataset. In particular, our method significantly outperforms ATOM in viewpoint change, low resolution, and partial occlusion by 8.2%, 4.7%, and 4.3%, respectively, which demonstrates the effectiveness of our second stage for cascaded verification. Since our framework mainly focuses on the target re-identification and re-detection, in the attributes such as aspect ratio change and scale variation, our method is less effective and slightly improves the baseline.

VOT2018 [5]. VOT2018 dataset contains 60 challenging videos for short-term tracking evaluation, which will reset the tracker to the ground-truth position when tracking failure occurs. On this benchmark, trackers are evaluated by the Expected Average Overlap (EAO), which considers both accu-

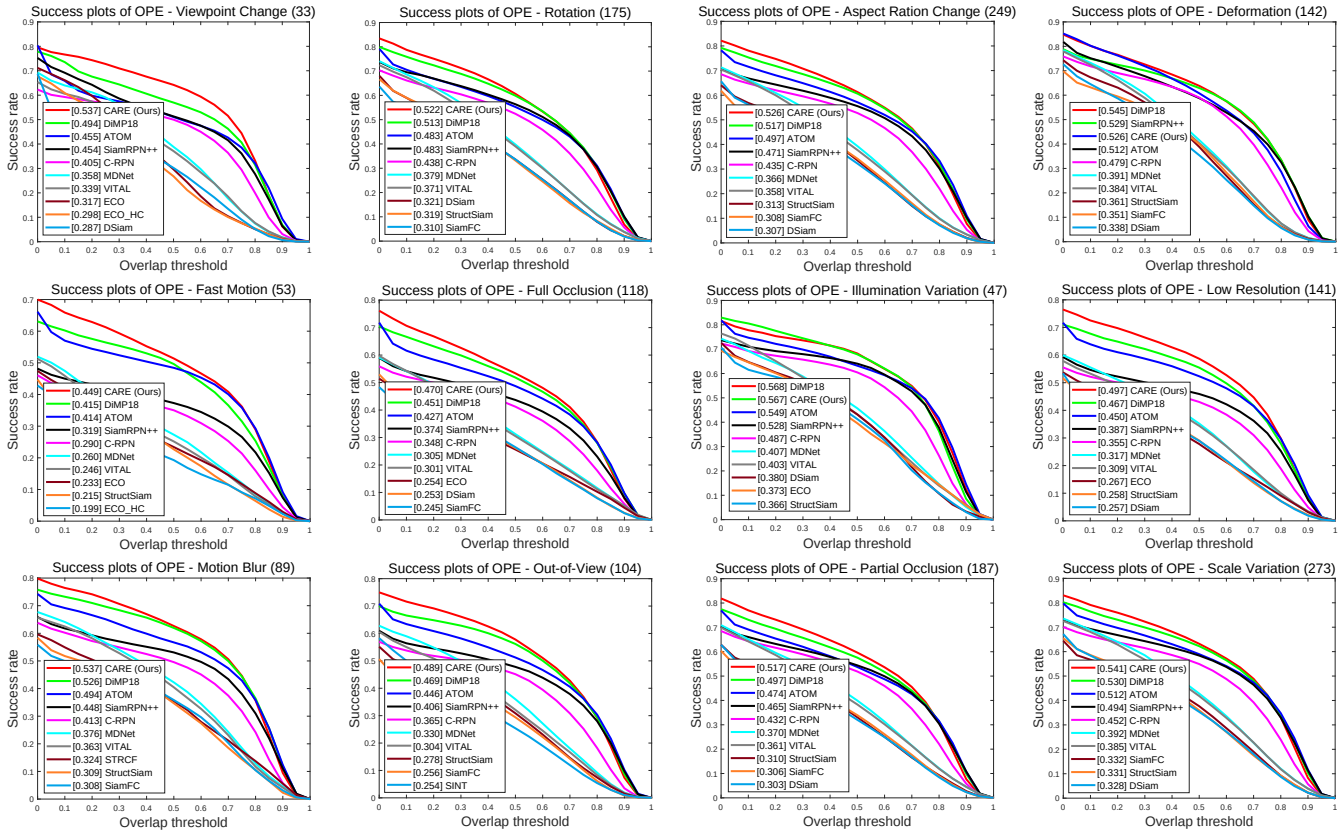


Fig. 9. Attribute-based evaluation on the LaSOT benchmark [18] including viewpoint change, rotation, aspect ration change, deformation, fast motion, full occlusion, illumination variation, low resolution, motion blur, out-of-view, partial occlusion, and scale variation. The legend shows the AUC score of the comaprison tracker.

TABLE II
COMPARISON WITH RECENT STATE-OF-THE-ART TRACKERS ON THE VOT2018 [5] IN TERMS OF ROBUSTNESS (R), ACCURACY (A), AND EXPECTED AVERAGE OVERLAP (EAO).

	SPM [33]	C-RPN [32]	DWSiam [63]	SiamMask [64]	SiamRPN++ [1]	ATOM [2]	DiMP-18 [6]	**CARE**
R	0.30	-	-	0.276	0.234	0.204	**0.182**	**0.201**
A	0.58	-	-	**0.609**	**0.600**	0.590	0.594	0.597
EAO	0.338	0.289	0.301	0.380	**0.414**	0.401	0.402	**0.410**
FPS	110	32	150	55	35	30	46	25

TABLE III
COMPARISON WITH RECENT STATE-OF-THE-ART TRACKERS ON THE VOT2019 [15] IN TERMS OF ROBUSTNESS (R), ACCURACY (A), AND EXPECTED AVERAGE OVERLAP (EAO).

	SPM [33]	SiamMask [64]	SiamMask-E [65]	SiamRPN++ [1]	ATOM [2]	SiamDW [63]	**CARE**
R	0.507	0.461	0.487	0.482	**0.411**	0.467	**0.343**
A	0.577	0.594	**0.652**	0.599	**0.603**	0.600	0.601
EAO	0.275	0.287	0.309	0.285	0.292	**0.299**	**0.323**
FPS	110	55	50	35	30	-	25

racy (average overlap over successful frames) and robustness (failure rate). From Figure 8, we can observe that our approach outperforms all the participants on the VOT2018. Compared with the recent state-of-the-art approaches, our approach still exhibits satisfactory results. As shown in Table II, our method surpasses the recent regression based methods such as ATOM and DiMP-18 with a relative gain of 2.2% and 2.0% in terms of EAO, respectively. Compared with the cascaded Siamese trackers including SPM [33] and C-RPN [32], our method significantly outperforms them thanks to our online adaptation capability. Among all the compared trackers, only SiamRPN++ slightly outperforms ours, which adopts a deeper ResNet-50 as the backbone network.

VOT2019 [15]. VOT2019 is the recently released challenging benchmark, which replaces 12 easy videos in VOT2018 [5] by 12 more difficult videos. Therefore, the EAO scores of the state-of-the-art trackers such as SiamRPN++ drop sharply. We compare our approach with the representative approaches in Table III. Compared with the SiamRPN++ [1] and SiamDW [63] with deeper ResNet-50, our method with a ResNet-18 obviously surpasses them. The ATOM [63] is a top-performing single-stage regression tracker, while ours outperforms it with a relative gain of 6.5% in terms of EAO.

TrackingNet [17]. The recent TrackingNet benchmark contains more than 30K videos with more than 14 million dense bounding box annotations. The videos are collected on the YouTube, providing large-scale high-quality data for assessing visual trackers in the wild. We evaluate our method on the test set of the recently released large-scale TrackingNet dataset, which consists of 511 videos. Note that the recent trackers already achieve outstanding AUC scores of more than

TABLE IV
COMPARISON WITH STATE-OF-THE-ART TRACKERS ON THE TRACKINGNET [17] TEST SET IN TERMS OF PRECISION, NORMALIZED PRECISION, AND SUCCESS (AUC SCORE OF THE SUCCESS PLOT).

	BACF [47]	Staple [66]	Staple-CA [41]	CSR-DCF [37]	ECOhc [48]	ECO [48]	SiamFC [3]	CFNet [67]	MDNet [52]	UPDT [68]	DaSiamRPN [21]	SPM [33]	C-RPN [32]	ATOM [2]	DiMP-18 [6]	**CARE**
Precision	46.1	47.0	46.8	48.0	47.6	49.2	53.3	53.3	56.5	55.7	59.1	66.1	61.9	64.8	**66.6**	**66.7**
Norm. Prec.	58.0	60.3	60.5	62.2	60.8	61.8	66.3	65.4	73.3	70.2	73.3	77.8	74.6	77.1	**78.5**	**79.0**
Success	52.3	52.8	52.9	53.4	54.1	55.4	57.1	57.8	63.8	61.1	63.8	71.2	66.9	70.3	**72.3**	**71.8**
Speed (FPS)	35	70	55	18	45	8	86	55	1	-	**160**	**110**	32	30	46	25

TABLE V
COMPARISON WITH STATE-OF-THE-ART TRACKERS ON THE UAV-20L [14], TEMPLE COLOR [13], AND NEED FOR SPEED [16] DATASETS. THE EVALUATION METRIC IS THE AUC SCORE OF THE SUCCESS PLOT.

	KCF [34]	DSST [35]	SRDCF [45]	HCF [36]	SiamFC [3]	CFNet [67]	ECOhc [48]	MDNet [52]	C-COT [43]	ECO [48]	SiamRPN [10]	SiamRPN++ [1]	ATOM [2]	DiMP-18 [6]	**CARE**
UAV20L [14]	19.8	27.0	34.3	-	39.9	34.9	-	-	-	43.5	45.4	56.1	55.4	**57.1**	**60.3**
TC128 [13]	38.4	40.6	50.9	48.2	50.5	45.6	56.1	56.3	58.3	59.7	-	56.2	59.3	**60.6**	**61.2**
NfS [16]	21.7	28.0	35.1	29.5	-	-	-	42.2	-	46.6	-	50.0	58.4	**61.0**	**60.5**
Speed (FPS)	**270**	45	5	12	86	55	45	1	0.3	8	**160**	35	30	46	25

TABLE VI
COMPARISON WITH STATE-OF-THE-ART TRACKERS ON THE OXUVA [19] DATASET IN TERMS OF TRUE POSITIVE RATE (TPR), TRUE NEGATIVE RATE (TNR), AND MAXIMUM GEOMETRIC MEAN (MAXGM). MAXGM IS THE FINAL EVALUATION METRIC.

	MDNet [52]	LCT [69]	TLD [70]	SiamFC+R [3]	MBMD [71]	SPLT [72]	**CARE**
MaxGM	0.343	0.396	0.431	0.454	0.544	**0.622**	**0.749**
TPR	0.472	0.292	0.208	0.427	**0.609**	0.498	**0.609**
TNR	0	0.537	**0.895**	0.481	0.485	0.776	**0.922**

70%, which means the improvement room on this dataset is limited. As shown in Table IV, the proposed tracker achieves a normalized precision score of 79.0% and a success score of 71.8%, which is comparable or superior to previous state-of-the-art trackers such as ATOM and DiMP-18.

Need for Speed [16]. NfS dataset contains 100 challenging videos with fast-moving targets, which aims at evaluating the tracking robustness in object fast-moving scenarios. We evaluate our approach on the 30 FPS version of NfS. The AUC scores of comparison approaches are shown in Table V. Since the search range is limited in SiamRPN++, its performance is relatively unsatisfactory (10.5% lower than ours in AUC). The state-of-the-art DiMP-18 and ATOM represent the top performance on this dataset. Our method is comparable with DiMP-18 and outperforms ATOM by 2.1% AUC.

Temple-Color [13]. Temple-Color benchmark is a challenging dataset consisting of 128 color videos. In Table V, we show the AUC score of state-of-the-art trackers on this benchmark. Compared with the SiamRPN++ with ResNet-50, our method outperforms it by a large margin of 5.0% AUC score. The recent single-stage regression tracker ATOM and DiMP yield AUC scores of 59.3% and 60.6%, respectively. The proposed approach also outperforms the recent single-stage regression tracker ATOM and DiMP-18 by 1.9% and 0.6% AUC score, respectively.

UAV20L [14]. This is a long-term tracking benchmark consisting of 20 long UAV videos with an average length of 2934 frames. Our second-stage regressor ensures the tracking robustness and helps re-detect the lost target. As a result, our method significantly surpasses previous methods such as ATOM, DiMP-18, and SiamRPN++ (Table V).

OxUvA [19]. This is a recent large-scale long-term tracking benchmark with 366 videos. The targets in OxUvA undergo frequent partial/full occlusion and out of view. On this dataset, the visual trackers are required to predict the target state (presence or absence) in each frame. We test our method on the test set of 166 videos. The comparison results are shown in Table VI. We do not add any additional mechanisms (e.g., global search) and merely use the reliability thresholds to predict the target presence/absence. Note that the recently proposed Skimming-Perusal method (SPLT) [72] leads the top performance on this dataset, which is specially designed for long-term tracking with a local-global search. Without bells and whistles, our approach outperforms SPLT by a relative gain of 20.4% in terms of MaxGM, showing the importance of online discrimination learning. The prior motion model (e.g., cosine window) in short-term trackers heavily limits their long-term performance. Benefited from strong discrimination, our tracker is free of the motion model (e.g., cosine window) and simultaneously handles short-term and long-term scenarios.

D. Performance with a Deeper Backbone Network

For fair comparison, in this section, we compare our method with state-of-the-art trackers with the same backbone network. In our approach, we follow ATOM [2] and use a shallow backbone network of ResNet-18 for high efficiency. By adopting the deeper ResNet-50 [58], our CARE approach obtains further performance improvements and still maintains a near real-time speed of about 21 FPS on a single Nvidia GTX 1080Ti GPU. In Table VII, we include the recent SiamRPN++ [1] and DiMP-50 [6] for comparison, both of which leverage the deep ResNet-50 model. From the results in Table VII, we can observe that our CARE-50 steadily outperforms SiamRPN++ and is comparable with the recent DiMP-50. It is worth mentioning that DiMP-50 represents the current state-of-the-art tracker in various tracking benchmarks.

TABLE VII
COMPARISON RESULTS OF STATE-OF-THE-ART DEEP TRACKERS WITH DIFFERENT BACKBONE NETWORKS. BY ADOPING A DEEPER BACKBONE NETWORK, OUR CARE TRACKER GAINS FURTHER PERFORMANCE IMPROVEMENT AND IS COMPARABLE WITH THE RECENT STATE-OF-THE-ART DIMP-50 APPROACH [6].

	Backbone Network	OTB2015 [12] AUC score	TC128 [13] AUC score	UAV123 [14] AUC score	NfS [16] AUC score	VOT2018 [5] EAO score	VOT2019 EAO score	LaSOT [18] AUC score	TrackingNet [17] Success score	Speed FPS
DaSiamRPN [21]	AlexNet	65.8	-	58.6	-	0.326	-	41.5	63.8	**160**
C-RPN [32]	AlexNet	66.3	-	-	-	0.289	-	45.5	66.9	32
SPM [33]	AlexNet	68.7	-	-	-	0.338	0.275	-	71.2	**110**
ATOM [2]	ResNet-18	67.1	59.3	63.5	58.4	0.401	0.292	51.4	70.3	30
DiMP-18 [6]	ResNet-18	66.2	60.6	63.4	61.0	0.402	-	53.5	72.3	46
CARE-18 (Ours)	ResNet-18	**70.5**	61.2	**65.4**	60.5	0.410	0.323	54.7	71.8	25
SiamRPN++ [1]	ResNet-50	69.6	56.2	61.3	50.0	0.414	0.285	49.6	73.3	35
DiMP-50 [6]	ResNet-50	68.4	**61.5**	64.5	**62.0**	**0.440**	**0.379**	**56.9**	**74.0**	40
CARE-50 (Ours)	ResNet-50	**71.2**	**61.7**	**64.6**	**62.3**	**0.427**	**0.353**	**56.1**	**74.2**	21

V. CONCLUSION

In this paper, we propose a conceptually simple yet effective discrete sampling based ridge regression, which performs as an alternative of the fully-connected layers to discriminate the candidates, but exhibits promising efficiency under a closed-form solution. Its high flexibility allows the incorporation of hard negative mining as well as our proposed adaptive ridge regression to enhance online discrimination. We further complement it with the convolutional regression to develop a cascaded framework for robust visual tracking. The first stage enables a fast and dense search, while the second stage guarantees distractor discrimination. The proposed method exhibits outstanding results on several challenging benchmarks with a real-time speed.

REFERENCES

[1] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "Siamrpn++: Evolution of siamese visual tracking with very deep networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[2] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "Atom: Accurate tracking by overlap maximization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[3] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[4] R. Tao, E. Gavves, and A. W. Smeulders, "Siamese instance search for tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[5] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Cehovin Zajc, T. Vojir, G. Bhat, A. Lukezic, A. Eldesokey *et al.*, "The sixth visual object tracking vot2018 challenge results," in *European Conference on Computer Vision Workshops (ECCV Workshop)*, 2018.

[6] G. Bhat, M. Danelljan, L. Van Gool, and R. Timofte, "Learning discriminative model prediction for tracking," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

[7] L. Zhang, A. Gonzalez-Garcia, J. v. d. Weijer, M. Danelljan, and F. S. Khan, "Learning the model update for siamese trackers," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

[8] K.-K. Sung and T. Poggio, "Example-based learning for view-based human face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 20, no. 1, pp. 39–51, 1998.

[9] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 32, no. 9, pp. 1627–1645, 2009.

[10] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[11] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[12] ——, "Object tracking benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 37, no. 9, pp. 1834–1848, 2015.

[13] P. Liang, E. Blasch, and H. Ling, "Encoding color information for visual tracking: algorithms and benchmark," *IEEE Transactions on Image Processing (TIP)*, vol. 24, no. 12, pp. 5630–5644, 2015.

[14] M. Mueller, N. Smith, and B. Ghanem, "A benchmark and simulator for uav tracking," in *European Conference on Computer Vision (ECCV)*, 2016.

[15] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, R. Pflugfelder, J.-K. Kamarainen, L. Cehovin Zajc, Drbohlav, and et al, "Proceedings of the ieee international conference on computer vision workshops (iccv workshop)," 2019.

[16] H. Kiani Galoogahi, A. Fagg, C. Huang, D. Ramanan, and S. Lucey, "Need for speed: A benchmark for higher frame rate object tracking," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[17] M. Müller, A. Bibi, S. Giancola, S. Al-Subaihi, and B. Ghanem, "Trackingnet: A large-scale dataset and benchmark for object tracking in the wild," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[18] H. Fan, L. Lin, F. Yang, P. Chu, G. Deng, S. Yu, H. Bai, Y. Xu, C. Liao, and H. Ling, "Lasot: A high-quality benchmark for large-scale single object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[19] J. Valmadre, L. Bertinetto, J. F. Henriques, R. Tao, A. Vedaldi, A. Smeulders, P. Torr, and E. Gavves, "Long-term tracking in the wild: A benchmark," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[20] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 fps with deep regression networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[21] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu, "Distractor-aware siamese networks for visual object tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[22] A. He, C. Luo, X. Tian, and W. Zeng, "A twofold siamese network for real-time object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[23] Q. Wang, Z. Teng, J. Xing, J. Gao, W. Hu, and S. Maybank, "Learning attentions: Residual attentional siamese network for high performance online visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[24] X. Li, C. Ma, B. Wu, Z. He, and M.-H. Yang, "Target-aware deep tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[25] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool, "One-shot video object segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[26] P. Voigtlaender and B. Leibe, "Online adaptation of convolutional neural networks for video object segmentation," *arXiv preprint arXiv:1706.09364*, 2017.

[27] N. Wang, Y. Song, C. Ma, W. Zhou, W. Liu, and H. Li, "Unsupervised deep tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[28] N. Wang, W. Zhou, G. Qi, and H. Li, "Post: Policy-based switch tracking," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[29] J. Gao, T. Zhang, and C. Xu, "Graph convolutional tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[30] Q. Guo, W. Feng, C. Zhou, R. Huang, L. Wan, and S. Wang, "Learning dynamic siamese network for visual object tracking," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[31] T. Yang and A. B. Chan, "Learning dynamic memory networks for object tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[32] H. Fan and H. Ling, "Siamese cascaded region proposal networks for real-time visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[33] G. Wang, C. Luo, Z. Xiong, and W. Zeng, "Spm-tracker: Series-parallel matching for real-time visual object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[34] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 37, no. 3, pp. 583–596, 2015.

[35] M. Danelljan, G. Häger, F. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *British Machine Vision Conference (BMVC)*, 2014.

[36] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015.

[37] A. Lukezic, T. Vojir, L. Cehovin Zajc, J. Matas, and M. Kristan, "Discriminative correlation filter with channel and spatial reliability," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[38] A. Bibi, M. Mueller, and B. Ghanem, "Target response adaptation for correlation filter tracking," in *European Conference on Computer Vision (ECCV)*, 2016.

[39] J. Choi, H. Jin Chang, J. Jeong, Y. Demiris, and J. Young Choi, "Visual tracking using attention-modulated disintegration and integration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[40] J. Choi, H. Jin Chang, S. Yun, T. Fischer, Y. Demiris, and J. Young Choi, "Attentional correlation filter network for adaptive visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[41] M. Mueller, N. Smith, and B. Ghanem, "Context-aware correlation filter tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[42] N. Wang, W. Zhou, Q. Tian, R. Hong, M. Wang, and H. Li, "Multi-cue correlation filters for robust visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[43] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg, "Beyond correlation filters: Learning continuous convolution operators for visual tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[44] K. Dai, D. Wang, H. Lu, C. Sun, and J. Li, "Visual tracking via adaptive spatially-regularized correlation filters," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[45] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015.

[46] F. Li, C. Tian, W. Zuo, L. Zhang, and M.-H. Yang, "Learning spatial-temporal regularized correlation filters for visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[47] H. K. Galoogahi, A. Fagg, and S. Lucey, "Learning background-aware correlation filters for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[48] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg, "Eco: Efficient convolution operators for tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[49] N. Wang, W. Zhou, Y. Song, C. Ma, and H. Li, "Real-time correlation tracking via joint model compression and transfer," *IEEE Transactions on Image Processing (TIP)*, 2020.

[50] Y. Song, C. Ma, L. Gong, J. Zhang, R. Lau, and M.-H. Yang, "Crest: Convolutional residual learning for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[51] X. Lu, C. Ma, B. Ni, X. Yang, I. Reid, and M.-H. Yang, "Deep regression tracking with shrinkage loss," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[52] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[53] I. Jung, J. Son, M. Baek, and B. Han, "Real-time mdnet," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[54] Y. Song, C. Ma, X. Wu, L. Gong, L. Bao, W. Zuo, C. Shen, R. W. Lau, and M.-H. Yang, "Vital: Visual tracking via adversarial learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[55] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

[56] R. Rifkin, G. Yeo, T. Poggio *et al.*, "Regularized least-squares classification," *Nato Science Series Sub Series III Computer and Systems Sciences*, vol. 190, pp. 131–154, 2003.

[57] L. Bertinetto, J. F. Henriques, P. H. Torr, and A. Vedaldi, "Meta-learning with differentiable closed-form solvers," in *International Conference on Learning Representations (ICLR)*, 2019.

[58] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[59] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang, "Acquisition of localization confidence for accurate object detection," in *European Conference on Computer Vision (ECCV)*, 2018.

[60] L. Huang, X. Zhao, and K. Huang, "Got-10k: A large high-diversity benchmark for generic object tracking in the wild," *arXiv preprint arXiv:1810.11981*, 2018.

[61] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European Conference on Computer Vision (ECCV)*, 2014.

[62] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[63] Z. Zhang and H. Peng, "Deeper and wider siamese networks for real-time visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[64] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. Torr, "Fast online object tracking and segmentation: A unifying approach," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[65] B. X. Chen and J. K. Tsotsos, "Fast visual object tracking with rotated bounding boxes," in *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV Workshop)*, 2019.

[66] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. Torr, "Staple: Complementary learners for real-time tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[67] J. Valmadre, L. Bertinetto, J. F. Henriques, A. Vedaldi, and P. H. Torr, "End-to-end representation learning for correlation filter based tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[68] G. Bhat, J. Johnander, M. Danelljan, F. S. Khan, and M. Felsberg, "Unveiling the power of deep tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[69] C. Ma, X. Yang, C. Zhang, and M.-H. Yang, "Long-term correlation tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[70] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 34, no. 7, pp. 1409–1422, 2012.

[71] Y. Zhang, D. Wang, L. Wang, J. Qi, and H. Lu, "Learning regression and verification networks for long-term visual tracking," *arXiv preprint arXiv:1809.04320*, 2018.

[72] 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 *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

Ning Wang received the B.E. degree in communication engineering from Tianjin University (TJU), in 2016. He is currently pursuing the Ph.D. degree with the Department of Electronic Engineer and Information Science, University of Science and Technology of China (USTC). His research interest is computer vision and his current research work is focused on video object tracking.

Wengang Zhou received the B.E. degree in electronic information engineering from Wuhan University, China, in 2006, and the Ph.D. degree in electronic engineering and information science from University of Science and Technology of China (USTC), China, in 2011. From September 2011 to 2013, he worked as a post-doc researcher in Computer Science Department at the University of Texas at San Antonio. He is currently a Professor at the EEIS Department, USTC.

His research interests include multimedia information retrieval and computer vision.

Qi Tian is currently a Chief Scientist in Artificial Intelligence at Cloud BU, Huawei. From 2018-2020, he was the Chief Scientist in Computer Vision at Huawei Noah's Ark Lab. He was aslo a Full Professor in the Department of Computer Science, the University of Texas at San Antonio (UTSA) from 2002 to 2019. During 2008-2009, he took one-year Faculty Leave at Microsoft Research Asia (MSRA).

Dr. Tian received his Ph.D. in ECE from University of Illinois at Urbana-Champaign (UIUC) and received his B.E. in Electronic Engineering from Tsinghua University and M.S. in ECE from Drexel University, respectively. Dr. Tian's research interests include computer vision, multimedia information retrieval and machine learning and published over 530 refereed journal and conference papers. His Google citation is over 19300+ with H-index 69. He was the co-author of best papers including IEEE ICME 2019, ACM CIKM 2018, ACM ICMR 2015, PCM 2013, MMM 2013, ACM ICIMCS 2012, a Top 10% Paper Award in MMSP 2011, a Student Contest Paper in ICASSP 2006, and co-author of a Best Paper/Student Paper Candidate in ACM Multimedia 2019, ICME 2015 and PCM 2007.

Dr. Tian research projects are funded by ARO, NSF, DHS, Google, FXPAL, NEC, SALSI, CIAS, Akiira Media Systems, HP, Blippar and UTSA. He received 2017 UTSA President's Distinguished Award for Research Achievement, 2016 UTSA Innovation Award, 2014 Research Achievement Awards from College of Science, UTSA, 2010 Google Faculty Award, and 2010 ACM Service Award. He is the associate editor of IEEE TMM, IEEE TCSVT, ACM TOMM, MMSJ, and in the Editorial Board of Journal of Multimedia (JMM) and Journal of MVA. Dr. Tian is the Guest Editor of IEEE TMM, Journal of CVIU, etc. Dr. Tian is a Fellow of IEEE.

Houqiang Li (S'12) received the B.S., M.Eng., and Ph.D. degrees in electronic engineering from the University of Science and Technology of China, Hefei, China, in 1992, 1997, and 2000, respectively, where he is currently a Professor with the Department of Electronic Engineering and Information Science.

His research interests include multimedia search, image/video analysis, video coding and communication. He has authored and co-authored over 200 papers in journals and conferences. He is the winner of National Science Funds (NSFC) for Distinguished Young Scientists, the Distinguished Professor of Changjiang Scholars Program of China, and the Leading Scientist of Ten Thousand Talent Program of China. He served as an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY from 2010 to 2013. He served as the TPC Co-Chair of VCIP 2010, and he will serve as the General Co-Chair of ICME 2021. He is the recipient of National Technological Invention Award of China (second class) in 2019 and the recipient of National Natural Science Award of China (second class) in 2015. He was the recipient of the Best Paper Award for VCIP 2012, the recipient of the Best Paper Award for ICIMCS 2012, and the recipient of the Best Paper Award for ACM MUM in 2011.