IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 12, 2017, accepted September 11, 2017, date of publication October 12, 2017, date of current version February 1, 2018.

Digital Object Identifier 10.1109/ACCESS.2017.2759583

Correlation-Filter Based Scale-Adaptive Visual Tracking With Hybrid-Scheme Sample Learning

WENHUI HUANG[1], (Graduate Student Member, IEEE), JASON GU[2], (Senior Member, IEEE), XIN MA[1], (Member, IEEE), AND YIBIN LI[1], (Member, IEEE)

[1]School of Control Science and Engineering, Shandong University, Jinan 250000, China
[2]Department of Electrical and Computer Engineering, Dalhousie University, Halifax, Nova Scotia, B3J 2X4, Canada

Corresponding authors: Jason Gu (e-mail: jason.gu@dal.ca) and Xin Ma (e-mail: maxin@sdu.edu.cn)

This work was supported in part by the National High-Tech Research and Development Program 863, China, under Grant 2015AA042307, in part by the Shandong Provincial Scientific and Technological Development Foundation, China, under Grant 2014GGX103038, in part by the Shandong Provincial Independent Innovation and Achievement Transformation Special Foundation, China, under Grant 2015ZDXX0101E01, in part by the Fundamental Research Funds of Shandong University, China, under Grant 2015JC027, in part by the Natural Sciences and Engineering Research Council of Canada, and in part by the Chinese Scholarship Council.

ABSTRACT In visual tracking, a mature scale estimation method can greatly improve tracking performance and provide accurate target information for model training. However, many visual tracking approaches ignore the scale estimation problem or adopt a heuristic and exhaustive scale-estimation strategy. In this paper, we propose a novel correlation-filter based visual tracking approach that reveals the missing link between scale estimation and the detection response. In contrast to many multi-scale visual trackers, which generate samples at different scales using some pre-designed criteria and then select the sample with the maximal classifier response, in this paper, we deduce a scale estimation equation based on detection responses; thus, the scale of the target object can be estimated mathematically. To obtain a more stable estimated object scale, a constraint function that considers the prior knowledge of visual tracking is proposed. Moreover, a hybrid sample learning scheme is formulated to select pertinent training samples with higher learning weights to train the appearance model. Our tracker operates under a framework of correlation filters to achieve a high tracking speed. We demonstrate the efficiency and robustness of our proposed tracking algorithm by comparing it with 14 other state-of-the-art trackers on all the video sequences in the object tracking benchmark (OTB) 2013 dataset.

INDEX TERMS Visual tracking, adaptive scale estimation, sample learning, correlation-filter based tracking.

I. INTRODUCTION

Visual tracking is a fundamental research topic in fields such as computer vision and robotics, and it has experienced rapid development in recent years [1]. In contrast to the trackers proposed during the early stages of visual tracking research, current tracking methods increasingly employ machine learning approaches to continuously and adaptively learn the appearance and posture of a target object. Furthermore, many works have applied visual tracking methods to build object tracking systems in the past decades, such as the FPGA-based object tracking system [2], [3], vision-based tracking for mobile robots [4], tracking with polarized stereo cameras [5] and 3D object tracking from monocular images [6]. Although these learning-based trackers have greatly improved the visual tracking performance, and numerous visual tracking applications have been developed, many problems remain to be solved, including scale adaptive tracking, sample selection and model training strategies [7].

Scale estimation is an essential component of a visual tracking algorithm. Detection-based visual tracking involves detecting an object across consecutive frames; therefore, the tracking performance in previous frames greatly affects the tracking performance in subsequent frames. A mature scale estimation method can provide accurate target information for both subsequent frames and for model training; thus, a scale-adaptive visual tracking approach truly tracks the target rather than tracking an image in a fixed-scale bounding box. Many existing algorithms ignore the scale estimation problem in visual tracking or adopt a heuristic and exhaustive scale-estimation strategy by generating some different scale samples and selecting the sample with the maximal similarity to the appearance model or with the maximal classifier

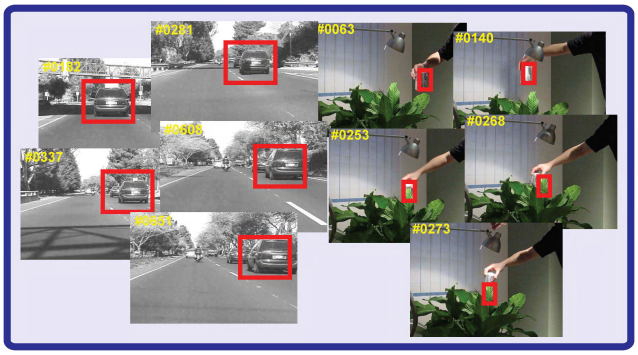

FIGURE 1. Examples of corrupt samples used for model training in a visual tracking algorithm. In many visual tracking methods, the tracking bounding box has a single scale. Thus, the box cannot adapt to the scale variations of a target object and will integrate imprecise training samples into model training. In addition, during the sample learning process of many trackers, every training sample has the same learning weight. Thus, the algorithm cannot assess the importance and reliability of different training samples and will integrate corrupt samples into training.

response. Moreover, because these multi-scale samples are produced by humans rather than by the model, they are generated discretely and may not meet the model's needs. Such practices greatly limit the precision of scale estimation.

Correlation-filter (CF) based trackers have attracted considerable interest in the visual tracking field because, in contrast to many existing trackers, they can achieve both high performance and high tracking speed. These tracking algorithms adopt the CF approach from the field of signal processing and pose the convolution of two images as an element-wise product in the Fourier domain. They also formulate the objective function in the Fourier domain; thus, the desired output of a linear classifier can be specified immediately. Among CF based trackers, tracking with a kernelized correlation filter (KCF) [8] is a typical method that exploits the circulant structure and utilizes the kernel trick to enhance the tracker. However, the KCF is a single-scale tracking algorithm, which decreases its performance in many situations, especially when the scale of the target object changes frequently and substantially. In this paper, we focus on how to adaptively estimate the target scale in CF based visual tracking.

This paper proposes a novel visual tracking method that reveals the missing link between target scale and detection response under the CF framework. A scale estimation equation is deduced based on the classifier response in each frame. Then, the adaptive target scale can be calculated mathematically rather than heuristically producing some samples with pre-designed criteria. In addition, to consider the prior knowledge of visual tracking and obtain a stable scale variable, we design an iterative strategy with a constraint function to update the scale variable and the bandwidth of the KCF. Moreover, we formulate a hybrid sample learning scheme to select the pertinent training samples with higher learning weights for training the appearance model and discarding corrupt samples. In contrast to many binary weight schemes, our approach assigns a real-valued weight to a sample, which can help determine sample importance. In addition, this hybrid scheme has a small error tolerance; that is, this method enables samples with small losses compared to the ground truth data to be fully learned.

In Section II, we review the recent related work on visual tracking. Section III introduces background information on KCF. Section IV describes the main principles of our proposed tracking algorithms, including the scale estimation equation, scale updating approach and sample learning strategies. In Section V, we present the results of experiments in which our tracking algorithm is applied to the object tracking benchmark (OTB) 2013 dataset and compare our tracker's performance with that of 14 other state-of-the-art tracking algorithms.

II. RELATED WORK

Many advanced and sophisticated methods in signal processing and machine learning have led to developments in visual tracking and computer vision in the past decades. Many outstanding works in visual tracking can be attributed to signal processing techniques such as optical flow [9] and motion estimation [10]–[13], AdaBoost [14], sparse representation [15], Kalman filters [4], particle filters [16] and the support vector machine [17].

Recently, correlation filters and deep learning have been attracting great attention in the computer vision and visual tracking fields [18], [19]. In this section, we review recent related work on visual tracking, including the CF based tracking methods, the deep learning based tracking methods, the experimental datasets used for visual tracking and scale estimation in visual tracking.

A. CORRELATION-FILTER BASED VISUAL TRACKING

CF based trackers adopt the correlation filter theory originally developed for signal processing. These trackers not only achieve high accuracy but can also track at high speeds. The KCF method [8] introduced the circulant matrix and the kernel trick to CF based methods. L. Bertinetto et al. [20] combined the strengths of the template and color-based models by proposing the Staple tracker, which uses complementary cues in a ridge regression framework. SRDCF [21] employed a spatial regularization component to update the correlation filter on larger image regions, which overcame the limitations of traditional CF based trackers. C-COT [22] used an implicit interpolation model that posed the learning problem in the continuous spatial domain, which enabled efficient integration of multi-resolution deep feature maps. ECO [23] is an extended version of C-COT that proposed using a factorized convolution operator to reduce the number of parameters in the model as well as a compact generative model of the training sample distribution to reduce the time complexity. ACFN [24] adopted an attentional mechanism to choose a subset of the associated correlation filters to increase the robustness and computational efficiency. In CACF [25], the authors proposed a framework that allows the explicit incorporation of global context within CF trackers.

B. DEEP LEARNING IN VISUAL TRACKING

In visual tracking, many works based on deep learning (DL) have been proposed. Compared with non-DL methods, the DL methods have achieved large performance improvements. The deep learning tracker (DLT) [26] was one of the first trackers to integrate a convolutional neural network (CNN) into visual tracking. The DLT used auxiliary natural images to train a stacked denoising autoencoder offline and learned generic image features that are more robust against variations. CF2 [27] analyzed features extracted from the different layers of a CNN and combined the CNN with correlation filters. MDNet [28] is based on the representations from a discriminatively trained CNN. This approach pre-trained a CNN using a large set of videos with tracking ground-truths to obtain a generic target representation, and it achieved exceptionally good performance. TCNN [29], which extended MDNet, is an online visual tracking algorithm that manages multiple target appearance models in a tree structure. SANet [30] employed the self-structure information of a target to distinguish it from distractors. A recurrent neural network (RNN) was used in this work to model object structure.

C. DATASET BUILDING

Building datasets that allow researchers to generate corresponding benchmarks has recently become an emerging trend in visual tracking. The work of Wu *et al.* [31], which included a dataset containing 50 fully annotated video sequences, is considered a milestone in this area. Moreover, this work synthesized most of the existing open source tracking methods into one code library and evaluated the methods across different metrics. The authors later extended this work to 100 video sequences [32]. NUS People and Rigid Objects (NUS-PRO) [33] proposed a large-scale dataset containing 365 challenge image sequences of pedestrians and rigid objects and analyzed the performances of 20 state-of-the-art tracking approaches on that dataset.

D. SCALE ESTIMATION IN VISUAL TRACKING

Scale estimation is a key technique in visual tracking that can significantly increase the success rate of trackers. Trackers have considered various methods to estimate the scale of the target. LCT [34] employed the HOG feature to build a multi-scale target pyramid, which is a typical practice used in visual tracking to estimate target scale. DSST [35] enabled an adaptive scale by learning discriminative correlation filters based on a scale pyramid representation. The limited correlation filter network (CFN-) [24] used the active module with the highest estimated validation score to determine the position and scale of the target. RPT [36] computed the posterior of each reliable patch and used those values to further estimate the scale and location of the tracked target using a scheme similar to Hough voting. STC [37], whose authors have contributed many outstanding works on visual tracking [16], [38], [39], took context information into consideration based on a Bayesian framework and estimated the target scale using a confidence map constructed from two consecutive frames.

Most related works on scale estimation in visual tracking use a generate-and-select technique or a pyramid model. These multi-scale samples are produced by humans rather than by the model; they are also generated discretely and may not meet the model needs. Such practices limit the precision of scale estimation.

Compared with STC, which is based on the Bayesian framework, our work is based on correlation filters, and we derive the scale estimation equation from the detection in KCF. We develop a constraint function that considers the prior knowledge of visual tracking to obtain a stable scale variable. Moreover, we propose a hybrid sample learning method that assigns different learning weights based on the quadratic loss of samples. In addition, we analyze and illustrate the effects of the different bandwidths of a Gaussian kernel on testing or training samples. In Section V, we compare the performance of our tracker with that of STC.

III. BACKGROUND

In this section, we present some preliminary information concerning the KCF tracker [8] used in our tracking algorithm. In KCF, the goal of training is to find a classifier response function $g(\mathbf{p}) = \mathbf{w}^T\mathbf{p}$ with a model parameter $\mathbf{w}$ for the image patch $\mathbf{p}$ by minimizing the following objective function:

$$\arg\min_{\mathbf{w}} \mathbb{O}(\mathbf{w}) = \sum_{i=1}^{n} D(q_i, g(\mathbf{p}_i, \mathbf{w})) + \xi||\mathbf{w}||^2, \qquad (1)$$

where $D(q_i, g(\mathbf{p}_i, \mathbf{w}))$ is the quadratic loss over the samples $\mathbf{p}_i$ and their regression targets q_i, $g(\mathbf{p}_i, \mathbf{w})$ is the response function of the classifier, and $\xi = 10^{-4}$ is a constant to prevent overfitting.

KCF employs the kernel trick to enable a more powerful, non-linear classifier. The model parameter $\mathbf{w}$ can be expressed as a linear combination of the samples, i.e., $\mathbf{w} = \sum_m a_m\varphi(\mathbf{p}_m)$, where $\varphi(\mathbf{p}_m)$ maps the inputs in a linear space to a non-linear feature space. Thus, the parameter under optimization is $\mathbf{a}$ rather than $\mathbf{w}$. According to KCF theory, the vector $\mathbf{a}$ of coefficients a_m can be learned as follows:

$$\hat{\mathbf{a}}(\mathbf{p}) = \frac{\hat{\mathbf{q}}}{\hat{\mathbf{k}}(\mathbf{p}, \mathbf{p}) + \xi}, \qquad (2)$$

where each element of $\mathbf{q}$ is a regression target q_i, ξ is the constant that prevents overfitting in Eq. (1), $\wedge$ is shorthand for a discrete Fourier transform (DFT), and $\mathbf{k}(\mathbf{p}, \mathbf{p})$ represents a kernel correlation.

The circulant matrix and kernel can also be used during detection to accelerate the tracker. The image patch $\mathbf{p}$ at the target location of the previous frame is sampled as the base sample for calculating the classifier response in the Fourier domain with the model patch $\boldsymbol{\rho}$ and model parameter $\boldsymbol{\alpha}$:

$$\hat{g}(\mathbf{p}) = \hat{\boldsymbol{\alpha}}(\mathbf{p}) \odot \hat{\mathbf{k}}(\mathbf{p}, \boldsymbol{\rho}), \qquad (3)$$

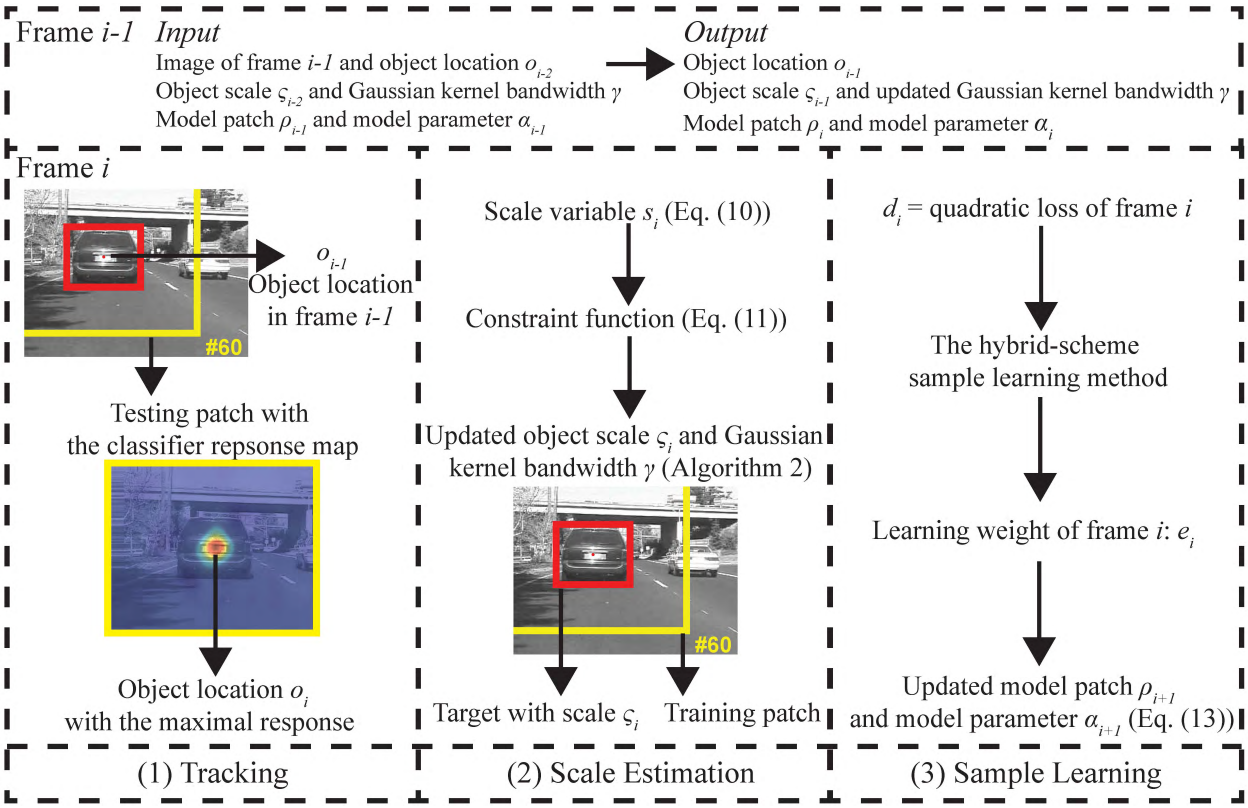

FIGURE 2. The main components of the proposed algorithm. We indicate the tracking inputs and outputs in frame $i-1$ to show the iterative process of our proposed tracking algorithm. The updated width and height of the target object in frame i are $width_i = width_{i-1} \cdot \varsigma_i$ and $height_i = height_{i-1} \cdot \varsigma_i$, respectively.

where $\mathbf{k}(\mathbf{p}, \boldsymbol{\rho})$ is the kernel correlation between the model patch $\boldsymbol{\rho}$ and the test image patch $\mathbf{p}$, and $\odot$ represents the element-wise product.

IV. OUR PROPOSED ALGORITHM

This section describes the main principles of our proposed tracking algorithm. First, we introduce the scale estimation equation, which is derived from the detection in KCF and inspired by [37]. Second, to obtain a stable estimated object scale, we propose a constraint function that considers the prior knowledge of visual tracking and then design an iterative scale-updating strategy. Third, we present the hybrid sample learning scheme used in this paper. Finally, we introduce the procedure used to update the model parameter and model patch using the sample learning weights. Fig. 2 shows the main components of our tracker, and the proposed algorithm is summarized in Algorithm 1.

A. SCALE ESTIMATION

In this subsection, we deduce the relationship between the scale variable and the detection response under the KCF framework. Based on KCF, the image patch $\mathbf{p}$ centered at the target location of the previous frame is sampled as the base sample for calculating the classifier response with the model patch $\boldsymbol{\rho}$ and model parameter $\boldsymbol{\alpha}$, and the detection can be formulated in time domain as follows :

$$g(\mathbf{p}) = \boldsymbol{\alpha}(\mathbf{p}) \otimes \mathbf{k}^{\gamma}(\mathbf{p}, \boldsymbol{\rho}), \tag{4}$$

where $\otimes$ represents convolution and $\mathbf{k}^{\gamma}(\mathbf{p}, \boldsymbol{\rho})$ is a Gaussian kernel correlation between the testing patch $\mathbf{p}$ and the model patch $\boldsymbol{\rho}$ with bandwidth γ. It is defined as follows:

$$\mathbf{k}^{\gamma}(\mathbf{p}, \boldsymbol{\rho}) = exp(-\frac{1}{\gamma^2}(\|\mathbf{p}\|^2 + \|\boldsymbol{\rho}\|^2 - 2\mathcal{F}^{-1}(\hat{\mathbf{p}}' \odot \hat{\boldsymbol{\rho}}))), \tag{5}$$

where $\mathbf{p}'$ is the complex-conjugate of $\mathbf{p}$ and $\mathcal{F}^{-1}$ is the inverse Fourier transform.

For convenience, we assume that the target is centered at (0,0) and that the image patch feature is a raw pixel. Based on the definition of convolution (i.e., $\boldsymbol{\alpha}(z) \otimes \mathbf{k}(z) = \int_{-\infty}^{\infty} \boldsymbol{\alpha}(\tau)\mathbf{k}(z-\tau)d\tau$), Eq. (4) can be represented as follows:

$$\boldsymbol{\alpha}(0,0) \otimes \mathbf{k}^{\gamma}(0,0) = \int\int_{\Omega_{x,y}} \boldsymbol{\alpha}(x,y)\mathbf{k}^{\gamma}(-x,-y)dxdy, \tag{6}$$

where (x, y) denotes the coordinates of the pixels in the image patch and $\Omega_{x,y}$ is the region of the image patch.

When the scale changes with a scale ratio s, i.e., $(sx, sy) = (x', y')$, Eq. (6) can be rewritten as follows:

$$\boldsymbol{\alpha}_i(0,0) \otimes \mathbf{k}_i^{\gamma_i}(0,0) = \int\int_{\Omega_{x',y'}} \boldsymbol{\alpha}_i(\frac{x'}{s}, \frac{y'}{s})\mathbf{k}_i^{\gamma_i}(-\frac{x'}{s}, -\frac{y'}{s})\frac{1}{s^2}dx'dy'$$

Algorithm 1 Our Proposed Tracking Algorithm

Input:

Image of frame i and previous object location o_{i-1}
Object scale ς_{i-1} and Gaussian kernel bandwidth γ
Model patch $\boldsymbol{\rho}_i$ and model parameter $\boldsymbol{\alpha}_i$

Output:

Object location o_i
Object scale ς_i and updated Gaussian kernel bandwidth γ
Model patch $\boldsymbol{\rho}_{i+1}$ and model parameter $\boldsymbol{\alpha}_{i+1}$

1: **while** $i <= n$ **do**
2: Track the target object by calculating the classifier response to the testing image patch $\mathbf{p}_i$ extracted from frame i and centered at o_{i-1} in the Fourier domain

$$\hat{g}(\mathbf{p}_i) = \hat{\boldsymbol{\alpha}}_i \odot \hat{\mathbf{k}}^{\gamma}(\mathbf{p}_i, \boldsymbol{\rho}_i)$$

Then, set o_i with the maximal classifier response g_i^* as the target location;
3: Estimate the scale variable s_i via Eq. (10) using g_{i-1}^* and g_i^*, and update the target scale ς_i and the bandwidth γ in the Gaussian kernel correlation (Eq. (5)) via Algorithm 2;
4: Obtain the learning weight e_i of the training image patch via the hybrid sample learning scheme;
5: Update $\boldsymbol{\rho}_{i+1}$ and $\boldsymbol{\alpha}_{i+1}$ via Eq. (13).
6: **end while**

$$\begin{aligned}
&\approx \int\int_{\Omega_{x',y'}} \boldsymbol{\alpha}_{i+1}(x', y')\mathbf{k}_i^{\gamma_i}(-\frac{x'}{s}, -\frac{y'}{s})\frac{1}{s^2}dx'dy' \\
&= \int\int_{\Omega_{x',y'}} \boldsymbol{\alpha}_{i+1}(x', y')\mathbf{k}_i^{s\gamma_i}(-x', -y')\frac{1}{s^2}dx'dy' \\
&\approx \int\int_{\Omega_{x',y'}} \boldsymbol{\alpha}_{i+1}(x', y')\mathbf{k}_{i+1}^{s\gamma_i}(-x', -y')\frac{1}{s^2}dx'dy'. \qquad (7)
\end{aligned}$$

In the above deduction, we employ two assumptions. In visual tracking, because target object attributes such as appearance and posture typically change only slightly between adjacent frames, the target features extracted from two adjacent frames are very similar. Based on this visual tracking characteristic, we propose the following two assumptions:

$$\begin{aligned}
\boldsymbol{\alpha}_i(\frac{x'}{s}, \frac{y'}{s}) &= \boldsymbol{\alpha}_i(x, y) \approx \boldsymbol{\alpha}_{i+1}(x', y'), \\
\mathbf{k}_i^{\gamma_i}(\frac{x'}{s}, \frac{y'}{s}) &= \mathbf{k}_i^{s\gamma_i}(x', y') \approx \mathbf{k}_{i+1}^{s\gamma_i}(x', y'). \qquad (8)
\end{aligned}$$

An intuitive explanation of our first assumption is that the model parameter $\boldsymbol{\alpha}_{i+1}$ is approximately equal to $\boldsymbol{\alpha}_i$. This approximate equality occurs because only frame $i+1$ is added to update the model parameter from frame i to frame $i+1$ and because the features of the target object in frame $i+1$ are similar to those in frame i. In the second assumption, the difference between $\mathbf{k}_i^{s\gamma_i}(x', y')$ and $\mathbf{k}_{i+1}^{s\gamma_i}(x', y')$ exists in the differences in the model patch $\boldsymbol{\rho}$. Based on the above analysis, the model patch $\boldsymbol{\rho}_{i+1}$ is approximately equal to $\boldsymbol{\rho}_i$. Thus, the second assumption is reasonable.

Therefore, we have the following:

$$\begin{aligned}
&\boldsymbol{\alpha}_i(0, 0) \otimes \mathbf{k}_i^{\gamma_i}(0, 0) \\
&\approx \int\int_{\Omega_{x',y'}} \boldsymbol{\alpha}_{i+1}(x', y')\mathbf{k}_{i+1}^{s_{i+1}\gamma_i}(-x', -y')\frac{1}{s_{i+1}^2}dx'dy' \\
&= \int\int_{\Omega_{x',y'}} \boldsymbol{\alpha}_{i+1}(x', y')\mathbf{k}_{i+1}^{\gamma_{i+1}}(-x', -y')\frac{1}{s_{i+1}^2}dx'dy' \\
&= \frac{1}{s_{i+1}^2}\boldsymbol{\alpha}_{i+1}(0, 0) \otimes \mathbf{k}_{i+1}^{\gamma_{i+1}}(0, 0) \qquad (9)
\end{aligned}$$

where γ_{i+1} indicates the updated bandwidth of the Gaussian kernel in frame $i+1$ and $\gamma_{i+1} = s_{i+1}\gamma_i$.

Then, we can obtain the relationship between the scale variable and the classifier response:

$$s_{i+1} = (\frac{\boldsymbol{\alpha}_{i+1}(0, 0) \otimes \mathbf{k}_{i+1}^{\gamma_{i+1}}(0, 0)}{\boldsymbol{\alpha}_i(0, 0) \otimes \mathbf{k}_i^{\gamma_i}(0, 0)})^{\frac{1}{2}} = (\frac{g_{i+1}^*}{g_i^*})^{\frac{1}{2}} \qquad (10)$$

where g_i^* represents the maximal classifier response in frame i.

B. CONSTRAINT FUNCTION AND SCALE UPDATING

An important target feature characteristic in visual tracking is that sample features extracted between two adjacent frames in a video sequence are similar because the appearance or scale of the target object changes only slightly between adjacent frames. Based on this characteristic, we assume that the variation in target scale will not increase or decrease greatly between consecutive frames. Therefore, we adopt a scale constraint function to integrate the above prior knowledge into scale estimation. The constraint function is defined as follows:

$$c(s) = -sign(\Delta s) \cdot \begin{cases} \theta, & log(1 + |\Delta s|) > \theta \\ log(1 + |\Delta s|), & log(1 + |\Delta s|) \le \theta, \end{cases} \qquad (11)$$

where $\Delta s = s - 1$ and $\theta = 0.5$ is a threshold. The constraint function ranges from $-\theta$ to θ. Fig. 3 shows the value of the constraint function. When the target scale is decreasing (i.e., the estimated scale variable s is less than 1), a positive constraint will be added to the scale variable to ensure that the target scale does not shrink too much. Similarly, when the target scale is increasing (i.e., the estimated scale variable s is greater than 1), a negative constraint will be added to the scale variable to ensure that the target scale does not expand too much.

The iterative scale-updating strategy is summarized in Algorithm 2. Figure 4 shows the effect of object scale on Gaussian kernel correlation. The Gaussian kernel correlation can be seen as adding Gaussian shaped weights to images. Based on Algorithm 2, when the object scale $\varsigma < 1$, the Gaussian bandwidth γ will decrease. As shown in Figure 4 (a) and (b), in the detection phase, a smaller bandwidth leads γ to a smaller effective detection area in the testing image patch. In

Algorithm 2 Scale Updating

Input:

Object scale ς_{i-1}
Gaussian kernel bandwidth γ
Current frame number i
Integer constant $j > 0$ % update object scale every j frames

Output:

Object scale ς_i
Gaussian kernel bandwidth γ

1: Calculate the average scale variable using j consecutive frames: $\bar{s}_i = \frac{1}{j}\sum_{t=0}^{j-1} s_{i-t}$;
2: Update the scale of the target in frame i with a learning ratio $0 < \lambda < 1$: $\varsigma_i = (1-\lambda)\varsigma_{i-1} + \lambda\bar{s}_i$;
3: Add the constraint (Eq. (11)) to the scale variable $\varsigma_i = \varsigma_i + c(\varsigma_{i-1})$;
4: Update the bandwidth of the Gaussian kernel via ς_i: $\gamma = \varsigma_i\gamma$.

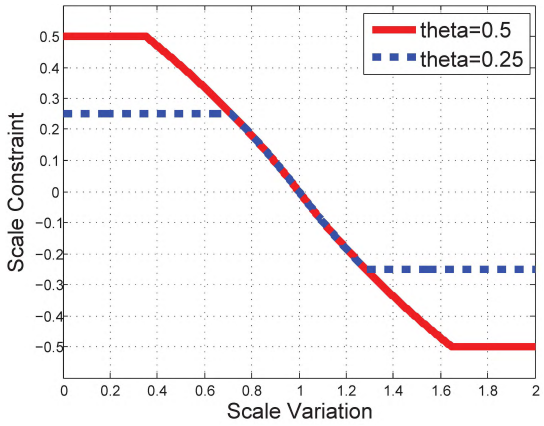

FIGURE 3. The constraint functions with different θ. This function is bounded in the range $[-\theta, \theta]$. When the scale variable is less than 1, a positive constraint will be added to the scale variable; otherwise, a negative constraint will be added. We adopt a logarithmic function because when the independent variable of a logarithmic function is increasing, the increment of the dependent variable gradually decreases, which ensures that after adding a constraint to the scale variable, the scale (i.e., $s_{after} = s_{before} + c(s_{before})$) will be a monotonically increasing function of s_{before}.

contrast, when the object scale $\varsigma > 1$, the effective detection area in the next frame will expand. Figure 4 (c) and (d) illustrate the Gaussian kernel correlation maps of two training image patches in training phrase. The image patch with the larger object scale ς (Figure 4 (c)) has a larger effective training area, while the image patch with the smaller object scale ς (Figure 4 (d)) has a smaller effective training area. Note that testing and training images always have the same dimensions. The effective testing and training area varies on both the object scale ς and the Gaussian kernel bandwidth γ.

C. SAMPLE LEARNING

In many existing sample learning methods, the learning variable can adopt only binary values, which is disadvantageous

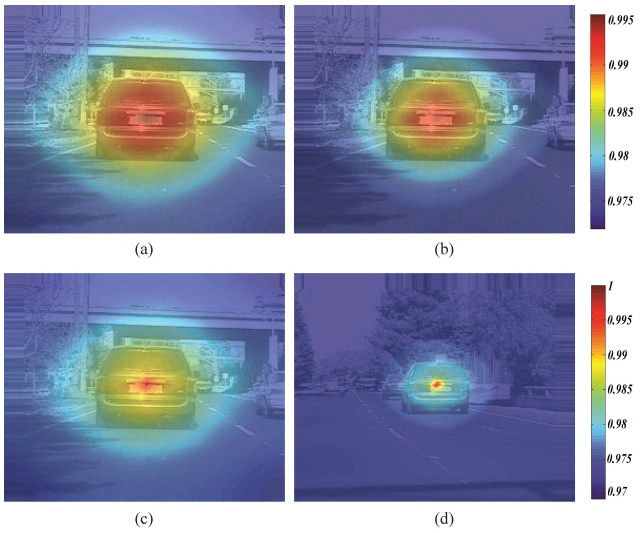

FIGURE 4. Gaussian kernel correlation maps. Subfigures (a) and (b) in the first row are the kernel correlation maps of the same testing image patch with different Gaussian bandwidths. The bandwidth in (a) is $\gamma = 0.4995$, and the bandwidth in (b) is $\gamma = 0.4495$. Subfigures (c) and (d) in the second row are the kernel correlation maps of two training image patches with different Gaussian bandwidths.

because such a hard threshold scheme cannot reflect the relative importance of samples. Samples with different losses are either selected for learning and given the same importance or omitted from learning when the sample losses exceed a threshold. In this paper, we formulate a new hybrid sample learning scheme that is a mixture of the hard and soft threshold schemes and that enables the model to select pertinent samples when learning a new appearance model [19]. On one hand, compared to the soft-weighting scheme, the hybrid scheme is fault tolerant. On the other hand, compared with the hard-weighting scheme, the hybrid scheme can assign real-valued weights that reflect the latent reliability of samples during training. When the sample loss is smaller or larger than the thresholds, the hard scheme is assigned to the learning variable; otherwise, the soft scheme is employed. After the target scale has been updated, calculating the value of the sample learning function allows pertinent training samples with higher learning weights to be selected for learning, whereas corrupt samples are removed.

The hybrid sample learning scheme proposed in this paper is introduced and analyzed from three perspectives. In addition, to define the threshold for the hard and soft schemes, an additional parameter is introduced, namely, $\sigma = [\sigma_1, \sigma_2]^T$, $(0 < \sigma_1 < \sigma_2 < 1)$. Here, $d_i \in [0, 1]$ represents the sample loss $D(q_i, g(\mathbf{p}_i, \mathbf{w}))$. Figure 5 illustrates the hybrid-scheme sample learning method.

1) When the sample loss is $d_i \in [0, \sigma_1)$, the learning weight of the sample is $e_i = f(d_i) = 1$. In this case, a hard scheme is employed for the learning weight. The sample's loss is less than σ_1, indicating that the sample is reliable enough to learn; thus, a learning weight equal to 1 is assigned to this sample. This practice is tolerant to small errors. Although a pertinent

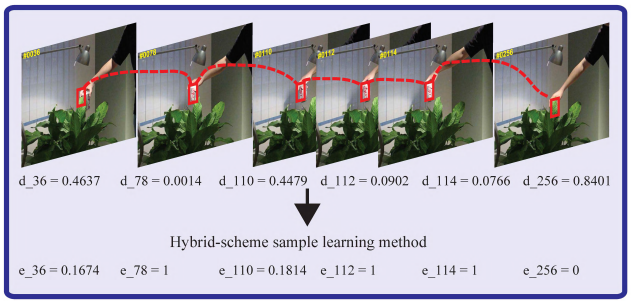

FIGURE 5. Illustration of the hybrid-scheme sample learning method. A real-valued learning weight e_i is assigned to each sample based on the image patch loss d_i. The red dashed lines represent the target trajectory. Using this approach, pertinent samples will be learned (e.g., e_{36} and e_{110}), and corrupt samples will be removed (e.g., e_{256}). Moreover, samples with small losses compared to the ground truth data will be fully learned (e.g., e_{78}, e_{112} and e_{114}).

sample may have a small loss compared to the ground truth data, it can still be fully learned.

2) When the sample loss is $d_i \in [\sigma_1, \sigma_2]$, the learning weight of the sample is $e_i = f(d_i) = \frac{\sigma_2 - d_i}{\sigma_2 d_i} \cdot \frac{\sigma_1 \sigma_2}{\sigma_2 - \sigma_1}$. The learning function is a deformation of the exponential function where the learning weight is negatively correlated to the sample loss. In this case, a soft scheme in which the sample takes a real-valued learning weight is employed. Thus, this approach can determine the relative importance of samples with different losses. Two special cases are when the loss d_i is equal to σ_1 and σ_2. When $d_i = \sigma_1$, e_i will be 1, which means this sample will be fully learned. When $d_i = \sigma_2$, e_i will be 0, which means this sample will not be learned.

3) When the sample loss is $d_i \in (\sigma_2, 1]$, the learning weight of the sample is $e_i = 0$. Under this condition, a hard scheme is employed for the learning weight. The sample's loss is greater than σ_2, which means that the sample is not sufficiently reliable to learn; thus, a learning weight equal to 0 is assigned to this sample. This approach protects the model from learning corrupt samples.

We propose a dynamic adaptive learning threshold σ in the hybrid learning scheme to replace the pre-determined threshold. In this scheme, rather than being predetermined, the learning threshold is determined according to the average losses D_M of the last M frames. For example, when the average loss of the last 40 frames D_{40} is greater than 0.2, we consider that the target has experienced a large-scale appearance change; thus, a larger threshold $\sigma = [0.45, 0.8]^T$ is adopted to ensure that more appearance information will be learned. In contrast, when the average loss of the last 40 frames D_{40} is less than 0.2, we consider the appearance of the target to have changed only slightly; thus, a smaller threshold $\sigma = [0.15, 0.8]^T$ will be adopted to better detect corrupted training samples. The threshold value is re-assigned every 40 frames during tracking.

D. UPDATE MODEL PATCH AND MODEL PARAMETER

After the pertinent samples have been selected for learning, a new model patch and model parameter will be learned from the learning weight e_i. According to KCF theory, the vector $\mathbf{a}_i$ of coefficients a_m can be learned as follows:

$$\hat{\mathbf{a}}_i = \hat{\mathbf{a}}(\mathbf{p}_i^*) = \frac{\hat{\mathbf{q}}}{\hat{\mathbf{k}}^{\gamma}(\mathbf{p}_i^*, \mathbf{p}_i^*) + \xi}, \tag{12}$$

where each element of $\mathbf{q}$ is a regression target q_i, $\mathbf{p}_i^*$ is the image patch extracted from frame i and centered at the target location o_i, and $\mathbf{k}^{\gamma}(\mathbf{p}_i^*, \mathbf{p}_i^*)$ represents the kernel correlation, as shown in Eq. (5).

Then, the model patch and model parameter can be updated as follows:

$$\begin{aligned} \hat{\boldsymbol{\rho}}_{i+1} &= (1-\lambda)\hat{\boldsymbol{\rho}}_i + \lambda \hat{\mathbf{p}}_i^* \cdot e_i, \\ \hat{\boldsymbol{\alpha}}_{i+1} &= (1-\lambda)\hat{\boldsymbol{\alpha}}_i + \lambda \hat{\mathbf{a}}_i \cdot e_i, \end{aligned} \tag{13}$$

where $0 < \lambda < 1$ is a constant that determines the ratio of new samples.

V. EXPERIMENTS

A. EXPERIMENTAL SETUP

Our proposed tracker runs at 131 fps when implemented in MATLAB on an i5 Quad-Core computer operating at 3.3 GHz with 8 GB of RAM. The experimental results shown in this section are based on all 50 video sequences in the OTB-2013 dataset [32]. Table 1 summarizes the values and ranges of all the key parameters in the proposed algorithm. The search window area is set to the same value as in CF2 [27] to obtain a more precise search area. 10 percent value of the constraint $c(s)$ is adopted in the videos *Car4*, *David*, *Liquor*, *Soccer*, *Trellis*, *Walking* and *Walking2*.

The OTB-2013 dataset includes 50 fully annotated videos and the tracking results of 29 visual trackers on these videos. The benchmark concludes that 11 attributes are mainly responsible for the types of interference that occur in video sequences: illumination variation (IV), out-of-plane rotation (OPR), in-plane rotation (IPR), scale variation (SV), occlusion (OCC), deformation (DEF), motion blur (MB), fast motion (FM), out-of-view (OV), background clutter (BC), and low resolution (LR). Various comparison criteria were proposed in OTB-2013 to evaluate the overall and attribute-based performances of trackers.

B. EVALUATION METRICS

In this study, we adopted two metrics to evaluate the performance of our tracker: precision plots and success plots [32]. A precision plot indicates the percentage of frames whose estimated target locations are within a specified threshold distance to the ground truth. A success plot illustrates the percentage of frames whose *overlap ratio* $> T$ for all threshold values is $T \in [0, 1]$. Overlap ratio is defined as $\textit{overlap ratio} = \frac{Area(B_t \cap B_g)}{Area(B_t \cup B_g)}$, where B_t is the tracked bounding box and B_g is the ground-truth bounding box. To rank the trackers, the commonly used precision score threshold is 20 pixels. Moreover, we adopted an additional measure of *overlap ratio* $= 0.5$ for each success plot to rank the trackers.

TABLE 1. The values and ranges of the parameters used in this paper.

Parameter	Value	Range
Initial object scale ς_1	1	$\varsigma_1 > 0$
Initial scale variable s_1	1	$s_1 > 0$
Initial feature bandwidth γ in Eq. (5)	0.2	$\gamma \in (0, 1)$
The threshold θ in Eq. (11)	0.5	$\theta \geq 0$
Average loss of the last M frames D_M	We use $M = 40$, i.e., D_{40}, to determin the value of $\sigma = [\sigma_1, \sigma_2]^T$	$D_M \in [0, 1]$
The threshold in the hybrid learning scheme $\sigma = [\sigma_1, \sigma_2]^T$	$\sigma = [0.45, 0.8]^T$, when $D_{40} > 0.2$, $\sigma = [0.15, 0.8]^T$, when $D_{40} \leq 0.2$	$0 \leq \sigma_1 < \sigma_2 \leq 1$
The constant λ in Eq. (13) and Algorithm 2	0.02	$\lambda \in (0, 1)$
The number of frames j used to calculate the average of the estimated scale in Algorithm 2	2	$1 \leq j \leq$ the current frame number

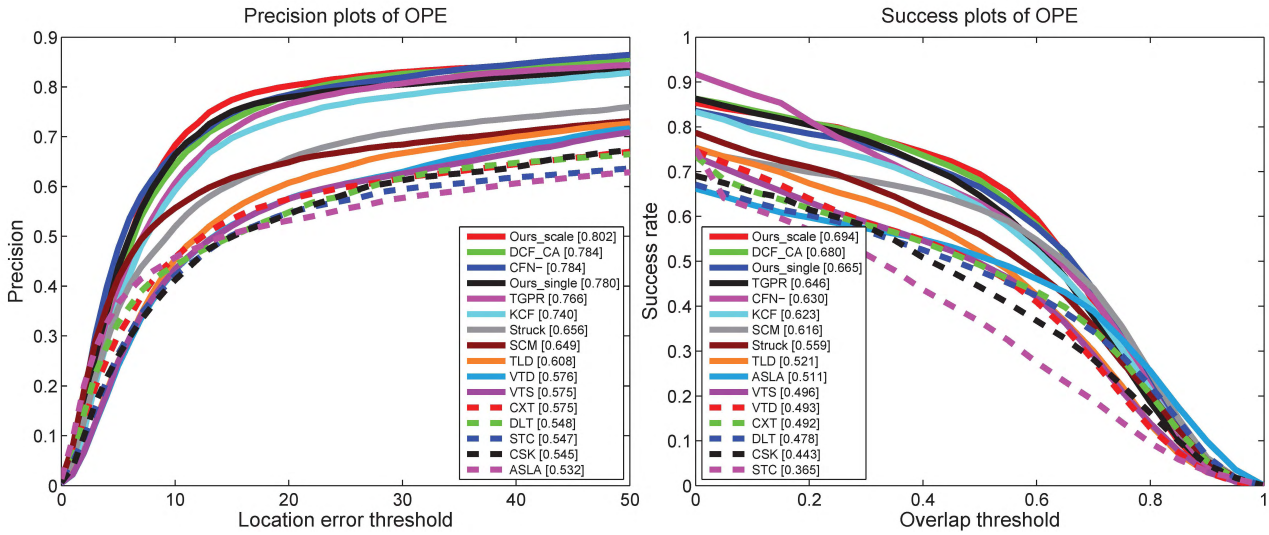

FIGURE 6. The precision and success plots of one-pass evaluation (OPE) for 16 trackers on OTB-2013. The "Ours_scale" represents our tracker with an adaptive scale, and "Ours_single" represents our tracker with a single scale. Our baseline tracker is KCF.

We compared our tracker to the top 8 trackers in the OTB-2013 dataset (i.e., Struck [17], SCM [40], TLD [41], VTD [42], VTS [43], CXT [44], CSK [45] and ASLA [46]) and to 6 other recent representative high performance trackers that are not included in the OTB dataset (i.e., DCF_CA [25], CFN- [24], TGPR [47], STC [37], DLT [26] and KCF [8]). Among these, the KCF tracker is our baseline tracker. CFN-, STC, DLT, SCM, TLD, VTD, VTS, CXT and ASLA are multi-scale trackers.

All the precision plots and success plots were generated from the OTB data. The KCF, CFN-, and TGPR data were sourced from their authors, the DCF_CA data were generated by the open source code available from its author, the STC and DLT data were sourced from the author of LCT [34], and the Struck, SCM, TLD, VTD, VTS, CXT, CSK and ALSA data were acquired from the OTB dataset. All the tracking performance scores shown in this paper are the results of one-pass evaluations [32].

C. QUANTITATIVE COMPARISONS

1) OVERALL PERFORMANCE

Comparisons of the overall performance of our proposed tracker with that of 14 other state-of-the-art trackers are presented in Fig 6. We summarize them in the precision and success plots and rank them in the legends in the left and right subfigures of Fig. 6, respectively. Our tracker achieves first place in both the precision plot and the success plot. In the precision plot, when the location error threshold is 20, our tracker's precision score is 0.802, outperforming DCF_CA (0.785) by 2.2%, and the baseline KCF method (0.740) by 8.4%. In the success plot, when the overlap threshold is 0.5, the success rate of our tracking method reaches 0.694, outperforming DCF_CA (0.680) by 2.1%, and the KCF tracker (0.623) by 11.4%.

Furthermore, we compare the performance of our proposed tracker both with scale estimation (Ours_scale in Fig.6) and without scale estimation (Ours_single in Fig.6). The precision rate (0.780) and success rate (0.665) of Ours_single are both higher than the baseline KCF tracker. This result can be attributed to the sample learning strategy adopted in this paper. Moreover, the precision rate and success rate of the Ours_scale tracker increased compared with that of Ours_single. This result can be attributed to the use of the proposed scale estimation method.

2) ATTRIBUTE-BASED PERFORMANCE

We also analyzed the performance of our proposed tracker based on the 11 attributes categorized in OTB-2013

TABLE 2. Precision plot scores of 8 trackers on OTB-2013 (the location error threshold is 20). The values in red indicate the highest scores, while those in blue are second-best.

Attribute	**Ours**	DCF_CA	CFN-	TGPR	KCF	Struck	SCM	STC
IV	0.811	0.768	0.669	0.687	0.728	0.558	0.594	0.578
OPR	0.780	0.776	0.760	0.741	0.729	0.597	0.618	0.542
IPR	0.782	0.785	0.693	0.706	0.725	0.617	0.597	0.518
SV	0.738	0.724	0.747	0.703	0.679	0.639	0.672	0.545
OCC	0.817	0.779	0.773	0.708	0.749	0.564	0.640	0.497
DEF	0.769	0.789	0.776	0.768	0.740	0.521	0.586	0.479
MB	0.762	0.703	0.635	0.578	0.650	0.551	0.339	0.330
FM	0.682	0.658	0.679	0.575	0.602	0.604	0.333	0.293
OV	0.739	0.653	0.799	0.495	0.650	0.459	0.361	0.395
BC	0.774	0.807	0.682	0.761	0.753	0.585	0.578	0.529
LR	0.507	0.400	0.474	0.539	0.381	0.545	0.305	0.265
Overall Score	0.802	0.784	0.784	0.766	0.740	0.656	0.649	0.547

to further demonstrate and analyze its strengthens and weakness.

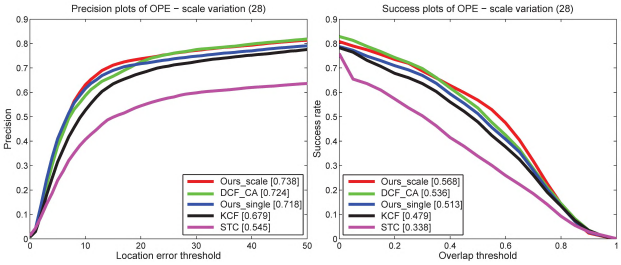

FIGURE 7. The precision and success plots of one-pass evaluation (OPE) for 5 trackers on 28 experimental video sequences with the scale variation attribute. Among them, the "Ours_scale" represents our tracker with an adaptive scale, and "Ours_single" represents our tracker with a single scale, KCF is our baseline tracker, DCF_CA is a correlation filter-based tracker, and our scale estimation method was inspired by STC.

Fig. 7 illustrates the precision and success plots of the trackers on the 28 video sequences from the OTB-2013 dataset that have the SV attribute. In terms of the precision plot, our tracker reaches 0.738 and outperforms the KCF method (0.679) by 8.7%. As the success plot shows, our tracker reaches 0.568, outperforming the KCF tracker (0.479) by 18.6%.

In the precision plot, the improvement from Ours_single (0.718) to Ours_scale (0.738) is 2.8%. However, in the success plot, the improvement from the Ours_single (0.513) to Ours_scale (0.568) is 10.7%. This result demonstrates that the hybrid sample learning strategy proposed in this paper mainly contributes to our tracker's increased precision, whereas the scale estimation method proposed in this paper mainly contributes to our tracker's increased success rate.

The precision rates of 8 of the experimental trackers on the other 10 video attributes are summarized in Table 2. Each score in the table represents the precision of one tracker with the location error threshold set to 20 pixels. The red and blue scores indicate the best and second-best performances, respectively. Our proposed tracker is the best or second-best tracker for 9 of 11 attributes, and it outperforms the KCF tracker on all attributes. Although our tracker is ranked third on the DEF attribute (0.769) and the LR attribute (0.507), it still outperforms KCF by 3.9% and 33.1%, respectively.

D. QUALITATIVE COMPARISONS

1) SCALE VARIATION

Figure 8 shows some sample results from three experimental videos in which the targets suffer from large scale variations. The "Car4" sequence contains both scale (e.g., #100 and #300) and illumination variations (e.g., #180 and #200). The DCF_CA, KCF, Struck and STC trackers cannot track the target as precisely as the other trackers when the target passes under a bridge and experiences large illumination changes (e.g., #250). As the target gradually moves forward, its scale decreases. Although the CFN- and TPGR trackers can both track the target throughout the video sequence, they cannot handle the SV and are not as precise as our tracker and SCM (e.g., #350). In the "Walking2" sequence, a pedestrian walking away from the camera undergoes large-scale changes (e.g., #50 and #300) and occlusions (e.g., #200). Only our tracker, CFN- and the SCM algorithm can persistently track the entire sequence and handle the scale variations (e.g., #300 and #500). The DCF_CA and KCF trackers drift away to the background when the target is occluded (e.g., #200 and #300). In contrast, the TPGR, Struck and STC trackers cannot adapt to the scale changes; thus, they do not track the target precisely as the scale decreases. The "Girl" video contains both scale variation challenges (e.g., #50 and #300) and in-plane or out-of-plane rotations (e.g., #100 and #120). The CFN- and STC trackers lose the target after the large-scale changes and out-of-plane rotations (e.g., from #100 to #220). In contrast to the KCF tracker, our tracker can precisely track the target (e.g., #275), and it adapts well to the scale variations.

2) OCCLUSION

Figure 9 presents some screenshots for three challenging sequences in which the targets undergo serious occlusions. The "Coke" video exhibits both serious occlusion (i.e., #200 and #258) and illumination variation challenges (e.g., #10 and #50). Overall, our tracker, CFN-, TGPR and Struck perform

FIGURE 8. Qualitative results of the 8 trackers on the videos *Car4*, *Walking2* and *Girl*, in which the targets undergo large-scale variations.

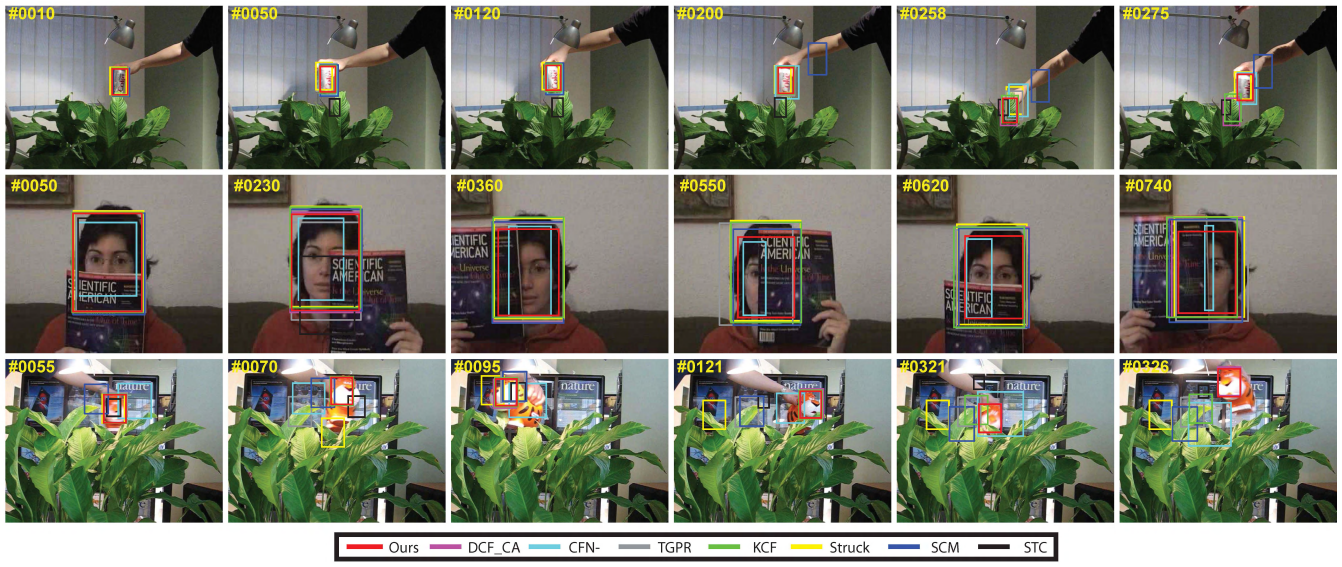

FIGURE 9. Qualitative results of the 8 trackers on the videos *Coke*, *FaceOcc1* and *Tiger1*, in which the targets undergo large-scale occlusions.

well on this sequence, whereas the other 4 compared trackers drift away to the background regions (e.g., #120 and #275). In the "FaceOcc1" sequence, the target is occluded by a book moving in from the bottom (e.g., #50 and #620), the left side (e.g., #360 and #740) and the right side (e.g., #230 and #550). Most of the experimental trackers can track the target in this sequence successfully, but the STC method loses the target starting in frame #230, and the CFN- tracker shrinks the target too much to be representative in some frames (e.g., #620 and #740). In the *Tiger1* sequence, the target rapidly moves behind some leaves (e.g., #55 and #321) and undergoes occlusions, illumination variations and rotations (#70 and #95). The CFN-, TGPR, Struck, SCM and STC methods cannot track the target when it moves rapidly (e.g., #70 and #121). Most of the trackers drift during some frames (e.g., #321) on this challenging video sequence; however, our tracker and the DCF_CA method can precisely track the target from the beginning to the end of the sequence (e.g., #326).

3) ROTATION

Figure 10 shows screenshots of the tracking results in three experimental videos where the targets undergo in-plane and out-of-plane rotations. In the "Dudek" sequence, the target moves indoors and experiences heavy out-of-plane rotations (e.g., #100, #600 and #700). The CFN- tracker cannot adapt to the scale change in some frames (e.g., #300 and #700). The STC tracker drifts to the background after the tracker experiences heavy rotations (e.g., #800). Our tracker tracks

FIGURE 10. Qualitative results of the 8 trackers on the videos *Dudek*, *David* and *Sylvester*, in which the targets undergo in-plane and out-of-plane rotations.

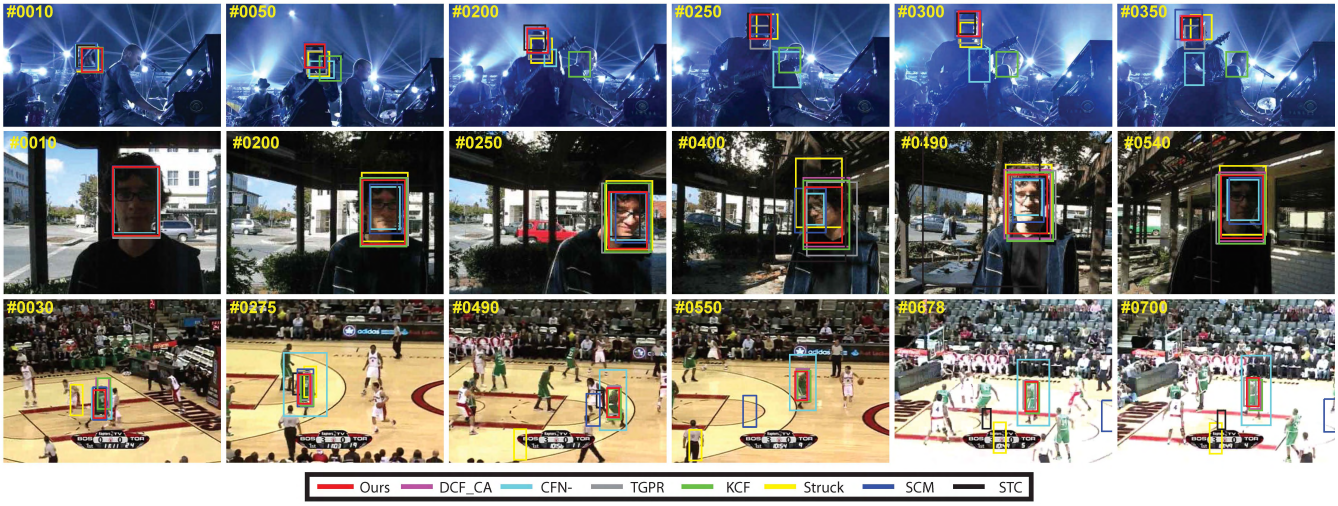

FIGURE 11. Qualitative results of the 8 trackers on the videos *Shaking*, *Trellis* and *Basketball*, in which the targets undergo illumination variations.

the target persistently and adapts well to the scale variations. In the "David" video, the target undergoes rotations (e.g., #415 and #455) and deformations (e.g., #580). Most of the tested tracking methods perform well on this sequence; however, the Struck method drifts heavily when the target rotates out-of-plane (e.g., #574). Compared with KCF, our tracker handles the scale change well and tracks the target more precisely (e.g., #610). In the *Sylvester* sequence, the target undergoes rotations (e.g., #550 and #620) and illumination changes (e.g., #275). Our tracker and the TGPR and SCM trackers perform well on all the frames, whereas the DCF_CA, CFN-, KCF, Struck and STC trackers drift when the target suffers from large-angle rotations (e.g., #1185).

4) ILLUMINATION VARIATION

Figure 11 shows screenshots of three videos in which the targets suffer from serious illumination variations. In the "Shaking" video, when the target moves in front of a dark background, the illumination around the target changes frequently (e.g., #10 and #300). The CNF- and KCF methods drift away to the background when a heavy illumination change occurs (e.g., #50). Although the Struck and SCM methods can track the target throughout the entire sequence, they achieve lower precision than our tracker and the DCF_CA, TPGR and STC trackers (e.g., #250, #300 and #350). In the "Trellis" video, the target moves from a dark area, and the illumination changes during the video sequence (e.g., #200 and #490). The STC method drifts to the background starting at frame #490 of the video clip. The CFN- method shrinks the target too much in some frames (e.g., #400 and #490). Our tracker and the DCF_CA, TPGR, KCF and SCM trackers handle the illumination variations and rotations to track the target well throughout the entire video sequence (e.g., #400 and #540). In the "Basketball" video, a man moves rapidly on a basketball court with illumination

variations (e.g., #678 and #700) and deformations (e.g., #30 and #275). The Struck and STC algorithms drift away to the background when the target is occluded or experiences heavy appearance variations (e.g., #30, #275 and #678). Moreover, the SCM does not handle fast motion and background blur well, which reduces its precision (e.g., #490 and #550). The CFN- tracker fails to adapt to the scale changes. In contrast, our tracker and the DCF_CA, TGPR, and KCF trackers, which can handle the illumination variations, perform well on the entire sequence (e.g., #700).

VI. CONCLUSION

A mature scale estimation method can greatly improve tracking performance and provide accurate target information for model training. This paper proposes a new CF based visual tracking algorithm that reveals the missing link between scale estimation and detection response by deducing a scale estimation equation from the detection response; thus, the scale is estimated mathematically rather than determined by the most common practice, namely, generating samples using some pre-designed criteria and identifying the sample with the maximal classifier response. Furthermore, to obtain a more stable estimated object scale, the estimated scale variable is updated iteratively using a constraint function that considers prior knowledge from visual tracking. Moreover, a hybrid sample learning scheme is formulated to select the pertinent training samples with higher learning weights to train the appearance model. The hybrid sample learning method not only assigns real values (as opposed to binary values) to a sample but is also error tolerant. Finally, we demonstrate the robustness and efficiency of our tracking algorithm on the OTB-2013 dataset through comparisons with 14 state-of-the-art trackers.

REFERENCES

[1] Z. Huang, Y. Yu, J. Gu, and H. Liu, "An efficient method for traffic sign recognition based on extreme learning machine," *IEEE Trans. Cybern.*, vol. 47, no. 4, pp. 920–933, Apr. 2017.

[2] P. Zhao, H. Zhu, H. Li, and T. Shibata, "A directional-edge-based real-time object tracking system employing multiple candidate-location generation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 3, pp. 503–517, Mar. 2013.

[3] G. Botella, J. A. Martín, M. Santos, and U. Meyer-Baese, "FPGA-based multimodal embedded sensor system integrating low- and mid-level vision," *Sensors*, vol. 11, no. 8, pp. 8164–8179, 2011.

[4] M. Gupta, S. Kumar, L. Behera, and V. K. Subramanian, "A novel vision-based tracking algorithm for a human-following mobile robot," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 7, pp. 1415–1427, Jul. 2017.

[5] C. V. Nguyen, M. Milford, and R. Mahony, "3D tracking of water hazards with polarized stereo cameras," in *Proc. IEEE Int. Conf. Robot. Autom.*, Jan. 2017, pp. 5251–5257.

[6] A. Crivellaro, M. Rad, Y. Verdie, K. M. Yi, P. Fua, and V. Lepetit, "Robust 3D object tracking from monocular images using stable parts," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published, doi: 10.1109/TPAMI.2017.2708711.

[7] W. Huang, J. Gu, and X. Ma, "Compressive sensing with weighted local classifiers for robot visual tracking," *Int. J. Robot. Autom.*, vol. 31, no. 5, pp. 416–427, 2016.

[8] 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.

[9] G. Botella, A. Garcia, M. Rodriguez-Alvarez, E. Ros, U. Meyer-Baese, and M. C. Molina, "Robust bioinspired architecture for optical-flow computation," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 4, pp. 616–629, Apr. 2010.

[10] D. Gonzalez *et al.*, "A low cost matching motion estimation sensor based on the NIOS II microprocessor," *Sensors*, vol. 12, no. 10, pp. 13126–13149, 2012.

[11] D. Gonzalez, G. Botella, C. Garcia, M. Prieto, and F. Tirado, "Acceleration of block-matching algorithms using a custom instruction-based paradigm on a NIOS II microprocessor," *EURASIP J. Adv. Signal Process.*, vol. 2013, p. 118, Jun. 2013.

[12] W.-H. Peng and C.-C. Chen, "An interframe prediction technique combining template matching prediction and block-motion compensation for high-efficiency video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 8, pp. 1432–1446, Aug. 2013.

[13] C. Jiang and S. Nooshabadi, "A Scalable Massively Parallel Motion and Disparity Estimation Scheme for Multiview Video Coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 2, pp. 346–359, Feb. 2016.

[14] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting," in *Proc. Brit. Mach. Vis. Conf.*, 2006, pp. 47–56.

[15] Y. Yu and Z. Sun, "Sparse coding extreme learning machine for classification," *Neurocomputing*, vol. 261, pp. 50–56, Mar. 2017.

[16] K. Zhang, L. Zhang, and M. Yang, "Fast compressive tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 10, pp. 2002–2015, Oct. 2014.

[17] S. Hare, A. Saffari, and P. H. S. Torr, "Struck: Structured output tracking with kernels," in *Proc. IEEE Int. Conf. Comput. Vis.*, Nov. 2011, pp. 263–270.

[18] Y. Yu, J. Gu, G. K. I. Mann, and R. G. Gosine, "Development and evaluation of object-based visual attention for automatic perception of robots," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 2, pp. 365–379, Feb. 2013.

[19] L. Jiang, D. Meng, Q. Zhao, S. Shan, and A. G. Hauptmann, "Self-paced curriculum learning," in *Proc. AAAI Conf. Artif. Intell.*, 2015, pp. 2694–2700.

[20] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. S. Torr, "Staple: Complementary learners for real-time tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 1401–1409.

[21] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 4310–4318.

[22] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg, "Beyond correlation filters: Learning continuous convolution operators for visual tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 472–488.

[23] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "ECO: Efficient convolution operators for tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2017, pp. 6638–6646.

[24] J. Choi, H. J. Chang, S. Yun, T. Fischer, Y. Demiris, and J. Y. Choi, "Attentional correlation filter network for adaptive visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2017, pp. 4807–4816.

[25] M. Mueller, N. Smith, and B. Ghanem, "Context-aware correlation filter tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2017, pp. 1396–1404.

[26] N. Wang and D.-Y. Yeung, "Learning a deep compact image representation for visual tracking," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2013, pp. 809–817.

[27] C. Ma, J. Huang, X. Yang, and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 3074–3082.

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

[29] H. Nam, M. Baek, and B. Han. (2016). "Modeling and propagating CNNs in a tree structure for visual tracking." [Online]. Available: https://arxiv.org/abs/1608.07242

[30] H. Fan and H. Ling, "SANet: Structure-aware network for visual tracking," in *Proc. CVPR Workshop*, Jul. 2017, pp. 2217–2224.

[31] Y. Wu, J. Lim, and M. Yang, "Online object tracking: A benchmark," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2411–2418.

[32] 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.

[33] A. Li, M. Li, Y. Wu, M.-H. Yang, and S. Yan, "NUS-PRO: A new visual tracking challenge," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 335–349, Feb. 2016.
[34] C. Ma, X. Yang, C. Zhang, and M.-H. Yang, "Long-term correlation tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 5388–5396.
[35] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *Proc. Brit. Mach. Vis. Conf.*, 2014.
[36] Y. Li, J. Zhu, and S. C. Hoi, "Reliable patch trackers: Robust visual tracking by exploiting reliable patches," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 353–361.
[37] K. Zhang, L. Zhang, Q. Liu, D. Zhang, and M.-H. Yang, "Fast visual tracking via dense spatio-temporal context learning," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 127–141.
[38] H. Song, Y. Zheng, and K. Zhang, "Robust visual tracking via self-similarity learning," *Electron. Lett.*, vol. 53, no. 1, pp. 20–22, 2017.
[39] H. Song, "Robust visual tracking via online informative feature selection," *Electron. Lett.*, vol. 50, no. 25, pp. 1931–1933, 2014.
[40] W. Zhong, H. Lu, and M.-H. Yang, "Robust object tracking via sparsity-based collaborative model," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1838–1845.
[41] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1409–1422, Jul. 2012.
[42] J. Kwon and K. M. Lee, "Visual tracking decomposition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 1269–1276.
[43] J. Kwon and K. M. Lee, "Tracking by sampling trackers," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2011, pp. 1195–1202.
[44] T. B. Dinh, N. Vo, and G. Medioni, "Context tracker: Exploring supporters and distracters in unconstrained environments," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2011, pp. 1177–1184.
[45] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 702–715.
[46] X. Jia, H. Lu, and M. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1822–1829.
[47] J. Gao, H. Ling, W. Hu, and J. Xing, "Transfer learning based visual tracking with Gaussian process regression," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 188–203.

WENHUI HUANG received the B.S. degree in electronic engineering from Beijing Information Science and Technology University, China, and the M.S. degree in computer science from The Chinese University of Hong Kong, Hong Kong. She is currently pursuing the Ph.D. degree with the School of Control Science and Engineering, Shandong University, China. Her research interests include computer vision, pattern recognition, and visual tracking.

JASON GU received the M.S. degree in biomedical engineering from Shanghai Jiaotong University, China, in 1995, and the Ph.D. degree from the University of Alberta, Canada, in 2001. He is currently a Professor with the Department of Electrical and Computer Engineering, Dalhousie University, Canada. His research areas include control, robotics, biomedical engineering, rehabilitation engineering, and neural networks.

XIN MA received the Ph.D. degree in aircraft control, guidance, and simulation from the Nanjing University of Aeronautics and Astronautics, China, in 1998. Since then, she has been with the School of Control Science and Engineering, Shandong University, China. She is currently a Professor and a Supervisor of doctoral candidates with Shandong University. Her current research interests include artificial intelligence, machine vision, human-robot interaction, and mobile robots.

YIBIN LI received the Ph.D. degree from the School of Electrical Engineering and Automation, Tianjin University, China, in 2008. He is currently a Professor with the School of Control Science and Engineering, Shandong University, China. His research interests include intelligent robots, intelligent vehicles and intelligent control, and electromechanical integration.