SOWP: Spatially Ordered and Weighted Patch Descriptor for Visual Tracking

Han-Ul Kim[1], Dae-Youn Lee[2], Jae-Young Sim[3], and Chang-Su Kim[1]
[1]School of Electrical Engineering, Korea University [2]Samsung Electronics Co., Ltd.
[3]School of Electrical and Computer Engineering, Ulsan National Institute of Science and Technology
{hanulkim, daeyounlee}@mcl.korea.ac.kr, jysim@unist.ac.kr, changsukim@korea.ac.kr

Abstract

A simple yet effective object descriptor for visual tracking is proposed in this paper. We first decompose the bounding box of a target object into multiple patches, which are described by color and gradient histograms. Then, we concatenate the features of the spatially ordered patches to represent the object appearance. Moreover, to alleviate the impacts of background information possibly included in the bounding box, we determine patch weights using random walk with restart (RWR) simulations. The patch weights represent the importance of each patch in the description of foreground information, and are used to construct an object descriptor, called spatially ordered and weighted patch (SOWP) descriptor. We incorporate the proposed SOWP descriptor into the structured output tracking framework. Experimental results demonstrate that the proposed algorithm yields significantly better performance than the state-of-the-art trackers on a recent benchmark dataset, and also excels in another recent benchmark dataset.

1. Introduction

Visual tracking has drawn much attention for its various applications, such as self-driving cars, security and surveillance systems, and augmented reality. With recent advances in machine learning, numerous tracking-by-detection algorithms [16, 17, 3, 4, 24, 18, 33, 40] have been proposed, which yield promising performances. A typical tracking-by-detection algorithm detects a target object over time, while updating a classifier using tracking results. Positive and negative samples, which correspond to foreground and background image regions, respectively, are employed to train the classifier. However, when training samples are assigned false labels or when sample descriptors cannot clearly distinguish between positive and negative samples, a classifier may be corrupted, degrading the tracking-by-detection performance.

To overcome these drawbacks, lots of algorithms have been proposed to reduce the impacts of false labeling [17, 4, 18, 24, 40] and to describe positive and negative samples distinguishably [33, 10]. However, it is still challenging to develop a reliable object descriptor, since the bounding box of a target object often contains background features due to deformation, occlusion, and object size variation. Attempts have been made to alleviate the effects of undesirable background features [7, 19, 31], but these approaches require prior knowledge or predefined parameters, the validity of which is hard to be verified.

We propose a novel object descriptor, called spatially ordered and weighted patch (SOWP) descriptor, to represent the appearance of an object faithfully and suppress background information in a bounding box systematically. To construct the SOWP descriptor, we divide the bounding box of a target object into non-overlapping patches and then describe each patch using an RGB histogram and a gradient histogram. We then concatenate the patch descriptors within the bounding box to convey structural information of the object. Moreover, to reduce the effects of background, we scale each patch descriptor with a weight, which represents the importance of the patch in the object description. For the patch weight computation, we perform two random walk with restart (RWR) simulations: one for foreground clustering and the other for background clustering. We incorporate this SOWP descriptor into the structured output tracking framework [18]. When updating the classifier, we use only the samples with high confidence scores to prevent bad updates. Experimental results show that the proposed tracker achieves the best performance on the CVPR2013 visual tracking benchmark dataset [38], and improves the performance of [18] by 36.3% in precision and 30.6% in success rate. Moreover, the proposed tracker also provides a competitive performance in the VOT2014 challenge dataset [28].

This work has three main contributions: First, we experimentally find a simple yet effective method to construct a spatially ordered patch descriptor for visual tracking. Second, we develop an RWR scheme to suppress the impacts of background information in an object bounding box through adaptive weighting. Third, the proposed algorithm significantly outperforms the recent state-of-the-art trackers in [21, 9, 1, 39, 5, 22, 14, 40, 32], as well as all 29 trackers in the CVPR2013 benchmark [38].

The remainder of this paper is organized as follows: Section 2 reviews related work, and Section 3 describes preliminaries. Section 4 presents the proposed algorithm, and Section 5 discusses experimental results. Finally, Section 6 draws conclusions.

2. Related Work

Since Avidan [2] first formulated visual tracking as a binary classification problem, many tracking-by-detection algorithms have been proposed [3, 16]. However, they may degrade the detection performance with falsely labeled samples. Grabner *et al.* [17] tried to prevent false labeling by training the classifier with labeled samples in the first frame and unlabeled samples in subsequent frames. Babenko *et al.* [4] employed a bag of multiple samples, instead of a single sample, for updating the classifier reliably. Hare *et al.* [18] adopted the structured output SVM [37] to alleviate the problem, caused by the ambiguity in binary labeling of samples, and provided excellent tracking results.

Comaniciu *et al.* [7] employed a color histogram to describe object appearance, but their method suffers from occlusion and cluttered background due to the lack of structural information. Patch-based object description has been adopted to exploit structural information [18, 36, 41], which divides the bounding box of an object into multiple patches and describes the object by concatenating low-level features of patches, such as the histogram of oriented gradients(HOG) [8]. Also, multiple features have been combined together to describe an object. Li *et al.* [33] combined intensity histograms, local binary pattern histograms, gradient histograms, and Haar-like features, based on a random decision tree, for compact and discriminative description. Chen *et al.* [6] extracted intensity and gradient data in patches and processed those data to yield a compound object descriptor. However, these object descriptors may become inadequate, when a bounding box contains background regions.

Researches have been carried out to reduce the effects of background information on object description. For example, Comaniciu *et al.* [7] assigned smaller weights to boundary pixels within a bounding box during the histogram construction. He *et al.* [19] also assumed that pixels far from a box center should be weighted less. Note that [7, 19] may fail to provide reliable tracking results, when a target object has a complicated shape or is occluded. Lee *et al.* [31] assigned a pertinence score to each patch by computing the appearance similarity of the patch to foreground and background regions. However, [31] uses fixed parameters for the pertinence score computation, which may cause tracking failures depending on input sequences. A related approach is to integrate a segmentation step into tracking in order to identify object patches and trace those patches directly, instead of a bounding box [15, 12]. These algorithms, however, are sensitive to segmentation results.

3. Preliminaries

The proposed algorithm performs RWR simulations [35] to reduce the impacts of background information in object description. Thus, we explain the notion of RWR in this section. We also describe the structured output tracking framework [18], into which we incorporate the proposed SOWP descriptor to design a tracker.

3.1. Random Walk with Restart

Consider a graph $G(V, E)$ with nodes $v \in V$ and edges $e \in E$. A random walker traverses the graph according to a transition matrix $\mathbf{A}$, in which the (i, j)th element a_{ij} is the probability that the walker at node v_j moves to node v_i. In an RWR simulation, the walker is forced to return to specified nodes, according to a restart distribution $\mathbf{r}$, with a probability ϵ. Let $\mathbf{p}^{(t)}$ denote the probability distribution of the walker at the tth iteration. Then, we have the recursion

$$\mathbf{p}^{(t)} = (1 - \epsilon)\mathbf{A}\mathbf{p}^{(t-1)} + \epsilon\mathbf{r}. \tag{1}$$

As t approaches infinity, $\mathbf{p}^{(t)}$ converges to the stationary distribution $\boldsymbol{\pi} = \lim_{t\to\infty} \mathbf{p}^{(t)}$, regardless of an initial condition $\mathbf{p}^{(0)}$. The stationary distribution $\boldsymbol{\pi}$ satisfies

$$\boldsymbol{\pi} = (1 - \epsilon)\mathbf{A}\boldsymbol{\pi} + \epsilon\mathbf{r}, \tag{2}$$

and it can be obtained by applying (1) recursively.

In the interactive image segmentation in [26], the restart distribution $\mathbf{r}$ represents the positions of user scribbles. Then, the stationary distribution indicates the affinity of each node to the scribbled nodes, and hence can be used for the segmentation. RWR has many applications, such as data mining [35], and saliency detection [25], as well as interactive image segmentation.

3.2. Structured Output Tracking

Let $\Phi(\mathbf{x}_t, \mathbf{y})$ denote a descriptor representing a bounding box $\mathbf{y}$ in the tth frame $\mathbf{x}_t$. The Struck algorithm [18] estimates the object bounding box $\mathbf{y}_t$ in the tth frame to maximize a classifier score $\langle \mathbf{w}, \Phi(\mathbf{x}_t, \mathbf{y})\rangle$, where $\mathbf{w}$ is the normal vector of a decision plane;

$$\mathbf{y}_t = \arg\max_{\mathbf{y}} \langle \mathbf{w}, \Phi(\mathbf{x}_t, \mathbf{y})\rangle. \tag{3}$$

When training the classifier, Struck attempts to avoid the labeling ambiguity by employing structured samples instead of binary-labeled samples. A structured sample consists of an object bounding box and nearby boxes in the same

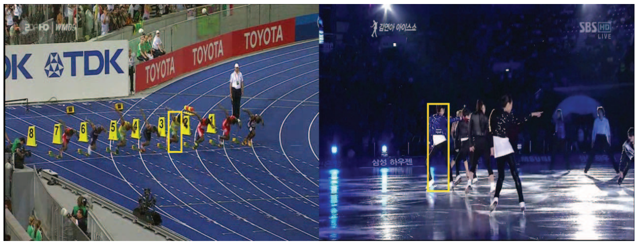

Figure 1. Examples of video sequences, in which non-target objects have the same category as a target object does. Target objects are within yellow bounding boxes.

frame. Struck constrains that the classifier score of an object bounding box should be larger than that of a nearby box by a margin, which is determined by the overlap ratio between the two boxes. Thus, Struck does not require a heuristic method to assign binary labels to training samples, and can reduce adverse effects of false labeling. Struck achieves excellent tracking performance, and its learning strategy has been adopted by many recent trackers [41, 5].

4. Proposed Algorithm

We propose an accurate and robust object description technique for visual tracking. First, we divide the bounding box of a target object into non-overlapping patches and construct a descriptor for each patch using multiple low-level features. We describe the object appearance by concatenating the patch descriptors to convey structural information of the target object. Second, we assign different weights to the patches adaptively, based on RWR [35], to yield the SOWP descriptor. We integrate the proposed SOWP descriptor into the structured output tracking framework in [18] to design a tracker.

4.1. Spatially Ordered Patch Descriptor

In visual tracking, it is important to construct a descriptor $\Phi(\mathbf{x}_t, \mathbf{y})$, which represents the contents in a bounding box $\mathbf{y}$ in the tth frame $\mathbf{x}_t$ distinguishably. To design an accurate and reliable descriptor, we first decompose the bounding box into 64 non-overlapping patches and characterize each patch using low-level features. Then, we construct the spatially ordered patch (SOP) descriptor for the bounding box, by concatenating the feature vectors of all patches according to their spatial orders, given by

$$\Phi(\mathbf{x}_t, \mathbf{y}) = [\mathbf{f}_1{}^T, \cdots, \mathbf{f}_{64}{}^T]^T \tag{4}$$

where $\mathbf{f}_i$ is the feature vector of the ith patch. SOP conveys structural information of the bounding box, by preserving the orders (or locations) of the patches within the box.

The concept of the SOP descriptor has been used in computer vision techniques. For example, the histogram of oriented gradients (HOG), which represents a bounding box by concatenating the oriented gradient histograms of patches within the box, is used in object detection [8, 13]. However, whereas the goal of object detection is to detect objects of a certain category, that of object tracking is to track a target object and distinguish it from possibly many non-target objects. Furthermore, it makes object tracking more difficult that the non-target objects may belong to the same category as the target object does. For instance, in Figure 1, for successful tracking, the object descriptor should be able to distinguish the target sprinter (or dancer) from the other sprinters (or dancers), who have similar shapes or colors as the target. In such a case, the descriptor, composed of a single type of features, may be inadequate.

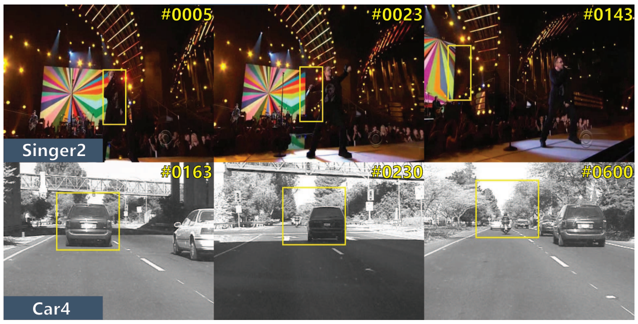

Figure 2. Inclusion of background information within an object bounding box due to complex object shape and size variation. In each of these sequence, background information causes a drift problem, and a tracker loses the target object eventually.

To design a more informative SOP descriptor, we tested various combinations of five low-level features: RGB color histogram, Lab color histogram, HSV color histogram, intensity histogram, and gradient histogram. We found that the combination of a 24-dimensional RGB color histogram and an 8-dimensional oriented gradient histogram results in superb tracking performance. Consequently, we describe each patch by using the 32-dimensional feature vector.

We also empirically set the number of patches within a bounding box. Too many patches increase the complexity due to a high descriptor dimension. In contrast, only a few patches cannot describe object structures faithfully. We tested various numbers of patches from 16 to 100, and selected 64 to strike a balance between the tracking performance and the complexity.

4.2. Adaptive Patch Weighting

It is desirable that the bounding box of a target object contains foreground information only. However, as shown in Figure 2, it may contain background information as well, which corrupts the object descriptor and the classifier. More specifically, the background information may be used for

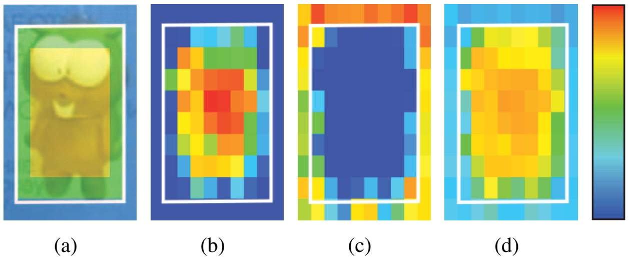

Figure 3. The computation of the foreground weight vector $\boldsymbol{\rho}_t$. In (a), three sets Ω_t^{in}, Ω_t^{bnd}, and Ω_t^{out} are defined over a bounding box to construct a graph. Then, the stationary distributions $\boldsymbol{\pi}_t^F$ in (b) and $\boldsymbol{\pi}_t^B$ in (c) are computed by carrying out the RWR simulations with different restart distributions. In (d), the foreground weight vector $\boldsymbol{\rho}_t$ is obtained by comparing $\boldsymbol{\pi}_t^F$ with $\boldsymbol{\pi}_t^B$.

training the classifier improperly, and the wrongly trained classifier, in turn, detects a bounding box with more background regions. This series of background expansion within a bounding box eventually results in tracking failures in Figure 2. To overcome this problem, we assign different weights to the patches within a bounding box according to their relevance to a target object.

Among the patches within a bounding box, we refer to those patches, sharing the boundary with the bounding box, as boundary patches, and the other patches as inner patches. In addition, we consider outer patches, which are located outside the bounding box but share the boundary. Let Ω_t^{in}, Ω_t^{bnd}, and Ω_t^{out} denote the sets of inner, boundary, and outer patches at the tth frame, respectively, which are depicted in yellow, green, and blue colors in Figure 3(a). We then construct a graph G, by taking each patch in these three sets as a node. If nodes v_i and v_j are 8-neighbors, they are connected by edge e_{ij}, which is assigned a weight

$$w_{ij} = \exp(-\gamma \|\mathbf{f}_i - \mathbf{f}_j\|^2) \tag{5}$$

where γ denotes a scaling parameter. A higher w_{ij} means that the two patches of v_i and v_j exhibit a stronger similarity. Then, we define a Markov transition matrix $\mathbf{A} = [a_{ij}]$ of a random walker, by normalizing the edge weights as

$$a_{ij} = \frac{w_{ij}}{\sum_i w_{ij}}. \tag{6}$$

We perform the RWR simulations on G twice using two different restart distributions $\mathbf{r}_t^F$ and $\mathbf{r}_t^B$, which are associated with the foreground and background regions, respectively. Then, we obtain the foreground and background stationary distributions, $\boldsymbol{\pi}_t^F$ and $\boldsymbol{\pi}_t^B$, via (2). Specifically,

$$\begin{aligned} \boldsymbol{\pi}_t^F &= (1-\epsilon)\mathbf{A}\boldsymbol{\pi}_t^F + \epsilon \mathbf{r}_t^F, & (7)\\ \boldsymbol{\pi}_t^B &= (1-\epsilon)\mathbf{A}\boldsymbol{\pi}_t^B + \epsilon \mathbf{r}_t^B, & (8) \end{aligned}$$

where ϵ is a restart probability. These stationary distributions $\boldsymbol{\pi}_t^F$ and $\boldsymbol{\pi}_t^B$ represent probabilistic shapes of a target object and its background, by forming clusters around the restart distributions $\mathbf{r}_t^F$ and $\mathbf{r}_t^B$, respectively [26].

As observed in [31], patches around the center of the bounding box tend to contain foreground features, whereas those near the box boundary are likely to have background features. It means that most patches in Ω_t^{in} tend to be foreground patches, while most patches in Ω_t^{out} background ones. Based on these assumptions, we determine the restart distributions as

$$\mathbf{r}_t^F(i) = \begin{cases} \kappa_t^F \times \boldsymbol{\pi}_{t-1}^F(i) & \text{if } v_i \in \Omega_t^{\text{in}} \cup \Omega_t^{\text{bnd}}, \\ 0 & \text{if } v_i \in \Omega_t^{\text{out}}, \end{cases} \tag{9}$$

$$\mathbf{r}_t^B(i) = \begin{cases} 0 & \text{if } v_i \in \Omega_t^{\text{in}}, \\ \kappa_t^B \times \boldsymbol{\pi}_{t-1}^B(i) & \text{if } v_i \in \Omega_t^{\text{bnd}} \cup \Omega_t^{\text{out}}, \end{cases} \tag{10}$$

where κ_t^F and κ_t^B are the normalizing parameters to make $\mathbf{r}_t^F$ and $\mathbf{r}_t^B$ probability distributions. Notice that we employ the stationary distributions at the previous frame to estimate the restart distributions at the current frame, since a target object tends to change its shape smoothly between consecutive frames in a typical video sequence. Figures 3(b) and (c) illustrate the stationary distributions $\boldsymbol{\pi}_t^F$ and $\boldsymbol{\pi}_t^B$, respectively, which reflect the likelihoods that each patch belongs to the foreground and the background, respectively.

We regard the ith patch to be a foreground patch, when it yields a large $\boldsymbol{\pi}_t^F(i)$ but a small $\boldsymbol{\pi}_t^B(i)$. Specifically, we compute the foreground weight $\boldsymbol{\rho}_t(i)$ of the ith patch by

$$\boldsymbol{\rho}_t(i) = \frac{1}{1 + \exp\left(-\alpha(\boldsymbol{\pi}_t^F(i) - \boldsymbol{\pi}_t^B(i))\right)} \tag{11}$$

where α controls the steepness of the logistic function. Figure 3(d) shows the foreground weights, where red and blue colors depict large and small weights, respectively. We see that the patches, which are assigned relatively large weights, reveal the shape of the target object effectively. Thus, we incorporate the foreground weights of the patches to concatenate the patch descriptors in (4), and consequently obtain the proposed SOWP descriptor, given by

$$\Phi(\mathbf{x}_t, \mathbf{y}_t) = [\boldsymbol{\rho}_t(1)\mathbf{f}_1{}^T, \cdots, \boldsymbol{\rho}_t(64)\mathbf{f}_{64}{}^T]^T. \tag{12}$$

Figure 4 exemplifies the evolution of computed foreground weights, which reflect the shapes of target objects over time faithfully. In particular, without the adaptive weighting, the tracking fails on the "Singer2" and "Car4" sequences in Figure 2. On the contrary, by using the foreground weights, the proposed tracker successfully trace the target objects in these sequences in Figure 4. It is worth pointing out that adaptive foreground weights implicitly reshape rectangular bounding boxes according to the deformation of target objects at different frames, and thus facilitate more reliable object description, especially when a target object is occluded or changes its shape and size.

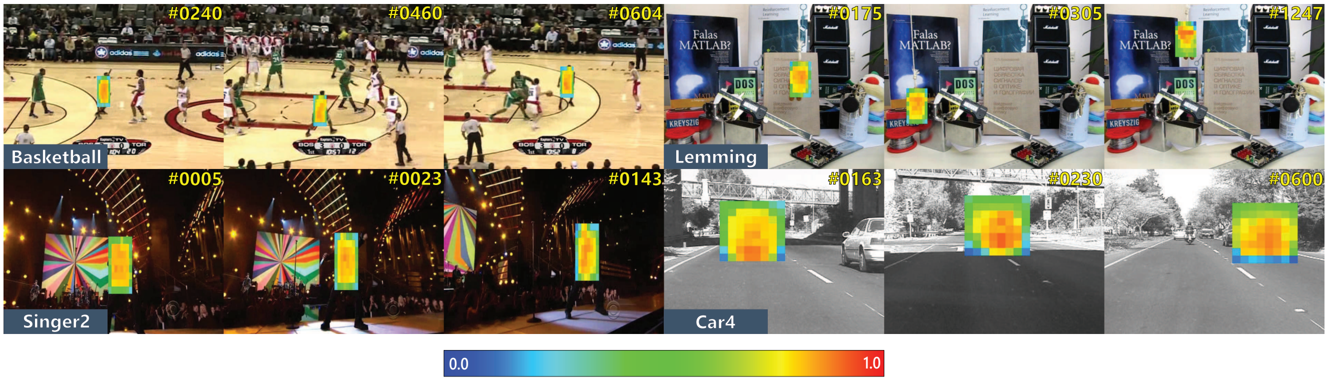

Figure 4. The evolution of patch weights through video sequences.

4.3. Tracking

We apply the proposed SOWP descriptor to trace a target object, by incorporating it into the conventional tracking-by-detection algorithm, Struck [18], which excels in the recent benchmark in [38]. Although Struck is adopted in this work, the SOWP descriptor also can be combined with other tracking-by-detection algorithms, such as [5, 40].

Given the object bounding box in a previous frame, we set a searching window, centered at the corresponding bounding box in the current frame. We construct the SOWP descriptor for each candidate bounding box within the searching window. Then, we select the optimal bounding box to yield the maximum classification score, and update the object location. We then update the foreground weights of the patches and the classifier. Finally, we proceed to the next frame.

Struck updates the classifier, even when an object changes its appearance abruptly, *e.g*., due to illumination variation and severe occlusion. In such cases, the classifier is trained with corrupted samples, degrading the tracking performance. To overcome this issue, we detect abrupt changes of object appearance. We define a confidence score θ_t to measure the reliability of the object bounding box in the tth frame. Specifically, θ_t is defined as the average similarity between the SOWP descriptor of the object bounding box and the positive support vectors, given by

$$\theta_t = \frac{1}{|\mathcal{S}_t|} \sum_{\mathbf{s} \in \mathcal{S}_t} \langle \mathbf{s}, \Phi(\mathbf{x}_t, \mathbf{y}_t) \rangle \tag{13}$$

where $\mathcal{S}_t$ is the set of the positive support vectors at time instance t. Note that the positive support vectors provide a compact summary of the object appearance in the previous frames [18]. Therefore, we update the classifier only when the confidence scores are greater than a threshold η.

5. Experimental Results

Implementation Details The proposed algorithm is implemented in C++ and performed at about 7.3 frames per second with an i7-4707HQ 2.40GHz CPU. We employ a linear kernel for the structured output SVM. The parameters are empirically set as $\gamma = 5.0$ in (5), $\epsilon = 0.75$ in (7), $\alpha = 35$ in (11), and $\eta = 0.3$ as the confidence score threshold. For computational efficiency, we scale each frame so that the minimum side length of a bounding box is 32 pixels. The side length of a searching window is fixed to $2\sqrt{WH}$, where W and H denote the width and height of the scaled bounding box, respectively.

CVPR2013 Visual Tracking Benchmark We evaluate the performance of the proposed tracker on the CVPR2013 visual tracking benchmark dataset [38] in Section 5.1 and Section 5.2. Precision (PR) and success rate (SR) are used to measure the quantitative performances of a tracker. The precision is the ratio of the frames, in which the distance between an estimated object location and the ground truth is smaller than a threshold. The success rate is the ratio of the frames, in which the overlap ratio between an estimated bounding box and the ground truth is larger than a threshold. In particular, the precision at the distance threshold of 20 pixels is employed as the representative PR score, and the average success rate, which is the area under the success rate curve over all overlap thresholds, is used as the representative SR score.

VOT2014 Challenge For more comprehensive evaluation, we also run the proposed tracker on the VOT2014 challenge dataset [28], which is an extended version of [27], in Section 5.3. Accuracy (ACC) and robustness (ROB) are used to assess the performance of a tracker. The accuracy computes the overlap ratio between an estimated bounding box and the ground truth. The robustness indicates the number of tracking failures, *i.e.,* the number of frames in which the overlap ratios are zero.

Table 1. PR/SR scores of the SOP and SOWP descriptors using various low-level features. A boldface number denotes the highest score of each descriptor in terms of each measure. The best PR/SR scores are achieved by the SOWP using 'RGB+Gradient' features, which is hence employed in the proposed algorithm.

	SOP	SOWP	Parameter
RGB	0.775 / 0.568	0.805 / 0.573	$\gamma = 10.0$
Lab	0.761 / 0.550	0.775 / 0.559	$\gamma = 20.0$
HSV	0.762 / 0.545	0.768 / 0.564	$\gamma = 20.0$
Intensity	0.731 / 0.526	0.765 / 0.543	$\gamma = 10.0$
RGB + Gradient	0.805 / 0.577	**0.870 / 0.604**	$\gamma = 5.00$
Lab + Gradient	**0.820 / 0.580**	0.838 / 0.579	$\gamma = 5.00$
HSV + Gradient	0.791 / 0.565	0.865 / 0.603	$\gamma = 10.0$
Intensity + Gradient	0.772 / 0.558	0.839 / 0.586	$\gamma = 10.0$

5.1. Component Analysis

Let us analyze the amount of the performance gain, which is achieved by each component of the proposed algorithm. Table 1 compares the performances of the descriptors using various low-level features on the benchmark dataset [38]. SOP uses the spatially ordered patch descriptor only. On the other hand, SOWP performs the patch weighting, based on RWR, in addition to SOP. Each color histogram is 24-dimensional, while each intensity or gradient histogram is 8-dimensional. We set different values of the parameter γ in (5), to take into account the characteristics of different features when computing the patch similarity. From Table 1, we can make the following observations: First, the SOP descriptor with the simple intensity feature only provides even better PR/SR scores than the best tracking performances (PR 0.656 by Struck [18] and SR 0.499 by SCM [42]) reported in the benchmark [38]. Second, the performance of the SOP descriptor is improved by combining a color histogram with a gradient histogram, which convey different types of information about an object. Third, the SOWP descriptor further improves the performance of SOP, regardless of the used features, by employing the adaptive patch weighting and reducing the impacts of background information in a bounding box. Last, the SOWP descriptor yields the best performance when using RGB color and gradient features together. Therefore, this combination of RGB and gradient histograms is adopted for the SOWP descriptor in the proposed algorithm.

Table 2. The performance of three versions of the proposed algorithm, as compared with the Struck algorithm [18]. Numbers in parenthesis are the performance gains over Struck.

	Struck	SOP	SOWP	SOWP+SU
PR	0.656	0.805 (22.7%)	0.870 (32.6%)	0.894 (36.3%)
SR	0.474	0.577 (21.7%)	0.604 (27.4%)	0.619 (30.6%)

Table 2 shows the performance gains of three versions of the proposed algorithm over the base tracker, Struck [18]. While Struck describes a bound box with Haar-like features, SOP descriptor divides the box into patches and describes each patch using RGB and gradient features. This simple modification improves the PR and SR scores by 22.7% and 21.7%, respectively. Moreover, by performing the adaptive patch weighting, the proposed SOWP descriptor further improves the performances and outperforms Struck by 32.6% and 27.4%. Finally, SOWP+SU, which uses the SOWP descriptor together with the selective classifier update in Section 4.3, achieves 36.3% better PR score and 30.6% better SR score than Struck.

5.2. Comparison with Conventional Trackers

Next, we compare the performances of the proposed algorithm (SOWP+SU) with those of the 29 conventional trackers whose results were reported in [38]. Figure 5 presents the PR and SR curves of the top ten trackers in the one-pass evaluation (OPE). Notice that the proposed algorithm outperforms the second best trackers significantly, achieving 36.3% gain over Struck in PR and 24.1% gain over SCM in SR. In supplemental materials, we show that the proposed algorithm also provides the best performances in the spatial robustness evaluation (SRE) and the temporal robustness evaluation (TRE) as well.

Table 3 compares the PR/SR scores of the proposed and conventional trackers according to various challenging factors. We see that the proposed algorithm always yields the best scores in terms of both PR and SR metrics. In particular, the proposed algorithm achieves relatively larger performance gains, as compared with the second best trackers, when target objects experience deformation or occlusion. Even when a target object changes its shape due to deformation, the proposed algorithm can reflect this structural change by updating patch weights adaptively. Also, in case of occlusion, the proposed algorithm efficiently suppresses the weights of background patches within a bounding box, based on the RWR simulation. Note that, for scale variation, the proposed algorithm yields a relatively low SR score, but maintains a high PR score. The SR performance, based on the overlap ratio between true and estimated bounding boxes, is degraded, since we do not adapt the size of a bounding box in this work. However, even when an object size varies, the proposed algorithm still estimates the object location accurately through the adaptive weighting of patches, yielding the high PR score.

Figure 6 compares the proposed algorithm with the five best conventional trackers in the benchmark. The "Basketball," "Bolt," "Lemming," "Liquor," and "Tiger1" sequences contain object deformation, occlusion, or cluttered backgrounds. Hence the conventional trackers cannot trace the target objects accurately on these sequences. For example, Struck loses the target sprinter on the "Bolt" sequence, due to another sprinter with similar appearance. In contrast, the proposed algorithm tracks the target successfully.

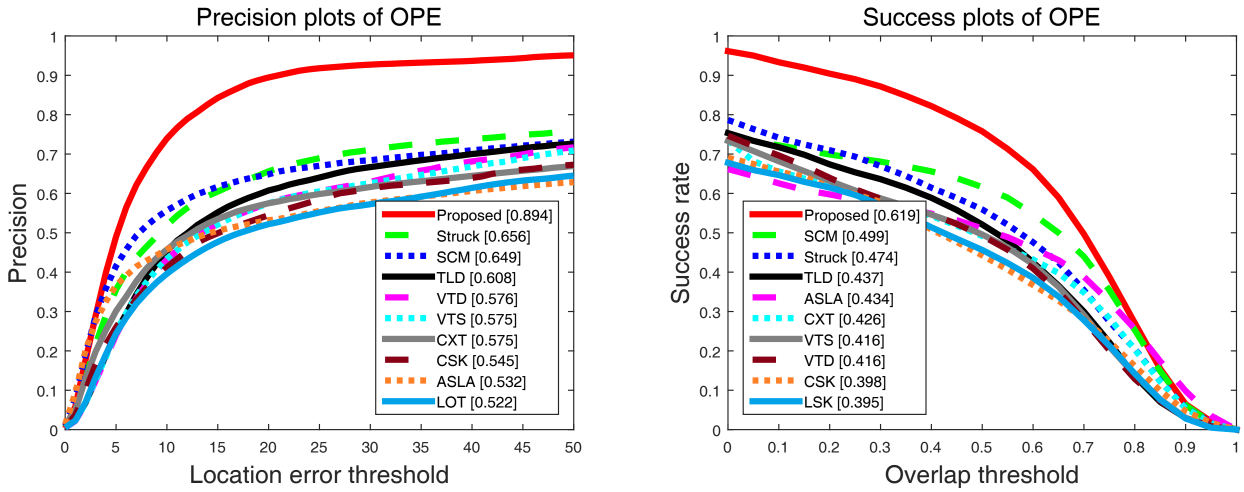

Figure 5. Comparison of the PR/SR plots in the OPE method. The representative scores in parenthesis measure the PR score at the threshold of 20 pixels and the average SR score over all overlap thresholds, respectively.

Table 3. Comparison of the PR/SR scores in the OPE method, according to 11 challenging factors: IV (illumination variation), SV (scale variation), OCC (occlusion), DEF (deformation), MB (motion blur), FM (fast motion), IPR (in-plane-rotation), OPR (out-of-plane rotation), OV (out-of-view), BC (background clutters), and LR (low resolution). Numbers in parenthesis in the first column refer to the numbers of sequences with the corresponding attributes. A boldface number denotes the highest score in each test.

	ASLA [23]	CXT [11]	CSK [20]	LSK [34]	VTD [29]	VTS [30]	TLD [24]	SCM [42]	Struck [18]	Proposed
IV(25)	0.517 / 0.429	0.501 / 0.368	0.481 / 0.369	0.449 / 0.371	0.557 / 0.420	0.573 / 0.429	0.537 / 0.399	0.594 / 0.473	0.558 / 0.428	**0.842 / 0.596**
SV(28)	0.552 / 0.452	0.550 / 0.389	0.503 / 0.350	0.480 / 0.373	0.597 / 0.405	0.582 / 0.400	0.606 / 0.421	0.672 / 0.518	0.639 / 0.425	**0.849 / 0.523**
OCC(29)	0.460 / 0.376	0.491 / 0.372	0.500 / 0.365	0.534 / 0.409	0.545 / 0.403	0.534 / 0.398	0.563 / 0.402	0.640 / 0.487	0.564 / 0.413	**0.867 / 0.603**
DEF(19)	0.445 / 0.372	0.422 / 0.324	0.476 / 0.343	0.481 / 0.377	0.501 / 0.377	0.487 / 0.368	0.512 / 0.378	0.586 / 0.448	0.521 / 0.393	**0.918 / 0.666**
MB(12)	0.278 / 0.258	0.509 / 0.369	0.342 / 0.305	0.324 / 0.302	0.375 / 0.309	0.375 / 0.304	0.518 / 0.404	0.339 / 0.298	0.551 / 0.433	**0.716 / 0.567**
FM(17)	0.253 / 0.247	0.515 / 0.388	0.381 / 0.316	0.375 / 0.328	0.352 / 0.302	0.353 / 0.300	0.551 / 0.417	0.333 / 0.296	0.604 / 0.462	**0.744 / 0.575**
IPR(31)	0.511 / 0.425	0.610 / 0.452	0.547 / 0.399	0.534 / 0.411	0.599 / 0.430	0.579 / 0.416	0.584 / 0.416	0.597 / 0.458	0.617 / 0.444	**0.847 / 0.584**
OPR(39)	0.518 / 0.422	0.574 / 0.418	0.540 / 0.386	0.525 / 0.400	0.620 / 0.434	0.604 / 0.425	0.596 / 0.420	0.618 / 0.470	0.597 / 0.432	**0.896 / 0.615**
OV(6)	0.333 / 0.312	0.510 / 0.427	0.379 / 0.349	0.515 / 0.430	0.462 / 0.446	0.455 / 0.443	0.576 / 0.457	0.429 / 0.361	0.539 / 0.459	**0.802 / 0.635**
BC(21)	0.496 / 0.408	0.443 / 0.345	0.585 / 0.421	0.504 / 0.388	0.571 / 0.425	0.578 / 0.338	0.428 / 0.428	0.578 / 0.450	0.585 / 0.458	**0.839 / 0.618**
LR(4)	0.156 / 0.157	0.371 / 0.312	0.411 / 0.350	0.304 / 0.235	0.168 / 0.177	0.187 / 0.168	0.349 / 0.309	0.305 / 0.279	0.545 / 0.372	**0.606 / 0.410**
Average(50)	0.532 / 0.434	0.575 / 0.426	0.545 / 0.398	0.000 / 0.395	0.576 / 0.416	0.575 / 0.416	0.608 / 0.437	0.649 / 0.499	0.656 / 0.474	**0.894 / 0.619**

Table 4. Comparison of the PR/SR scores of the proposed tracker and recent state-of-the-art trackers in the OPE method, according to the 11 challenging factors. For the descriptions of the challenging factors, refer to the caption of Table 3

	KCF [21]	DSST [9]	SAMF [1]	LLP [39]	MQT [22]	TGPR [14]	DDCT [5]	MEEM [40]	MTA [32]	Proposed
IV(25)	0.728 / 0.493	0.727 / 0.534	0.735 / 0.563	0.720 / 0.525	0.628 / 0.489	0.687 / 0.486	0.665 / 0.499	0.778 / 0.548	0.738 / 0.547	**0.842 / 0.596**
SV(28)	0.679 / 0.427	0.723 / 0.516	0.730 / **0.541**	0.644 / 0.498	0.692 / 0.464	0.703 / 0.443	0.687 / 0.484	0.809 / 0.506	0.721 / 0.478	**0.849** / 0.523
OCC(29)	0.749 / 0.514	0.845 / **0.619**	0.716 / 0.534	0.710 / 0.524	0.654 / 0.519	0.708 / 0.494	0.723 / 0.534	0.815 / 0.560	0.772 / 0.563	**0.867** / 0.603
DEF(19)	0.740 / 0.534	0.813 / 0.622	0.660 / 0.510	0.754 / 0.566	0.785 / 0.589	0.768 / 0.556	0.804 / 0.602	0.859 / 0.582	0.851 / 0.622	**0.918 / 0.666**
MB(12)	0.650 / 0.497	0.651 / 0.519	0.547 / 0.464	0.533 / 0.427	0.618 / 0.488	0.578 / 0.440	0.691 / 0.553	**0.740** / 0.565	0.695 / 0.540	0.716 / **0.567**
FM(17)	0.602 / 0.459	0.663 / 0.515	0.517 / 0.435	0.586 / 0.444	0.614 / 0.494	0.575 / 0.441	0.685 / 0.534	**0.757** / 0.568	0.677 / 0.524	0.744 / **0.575**
IPR(31)	0.725 / 0.497	0.691 / 0.507	0.765 / 0.560	0.652 / 0.477	0.671 / 0.482	0.706 / 0.487	0.720 / 0.524	0.810 / 0.531	0.773 / 0.547	**0.847 / 0.584**
OPR(39)	0.729 / 0.495	0.763 / 0.554	0.733 / 0.535	0.622 / 0.470	0.711 / 0.521	0.741 / 0.507	0.726 / 0.518	0.854 / 0.566	0.777 / 0.557	**0.896 / 0.615**
OV(6)	0.650 / 0.550	0.708 / 0.609	0.515 / 0.459	0.639 / 0.511	0.484 / 0.502	0.495 / 0.431	0.622 / 0.524	0.730 / 0.597	0.612 / 0.534	**0.802 / 0.635**
BC(21)	0.753 / 0.535	0.708 / 0.524	0.694 / 0.517	0.659 / 0.488	0.673 / 0.518	0.761 / 0.543	0.660 / 0.502	0.808 / 0.578	0.795 / 0.592	**0.839 / 0.618**
LR(4)	0.381 / 0.312	0.459 / 0.361	0.497 / 0.409	0.256 / 0.230	0.461 / 0.326	0.539 / 0.351	0.526 / **0.411**	0.494 / 0.367	0.579 / 0.397	**0.606** / 0.410
Average(50)	0.740 / 0.514	0.777 / 0.570	0.737 / 0.554	0.723 / 0.543	0.723 / 0.529	0.759 / 0.539	0.762 / 0.557	0.840 / 0.570	0.812 / 0.583	**0.894 / 0.619**

In “Ironman” and “Matrix,” the targets change their appearances abruptly at 34th and 2nd frames, respectively, which causes false updates of the conventional trackers. However, the proposed algorithm skips updating the classifier on such frames and thus traces the target objects more reliably.

In Table 4, we also compare the proposed tracker with recent state-of-the-art trackers, which are not in the benchmark: KCF [21], DSST [9], SAMF [1], LLP [39], MQT [22], DDCT [5], TGPR [14], MEEM [40], MTA [32]. We use the PR/SR scores reported in the literatures for the

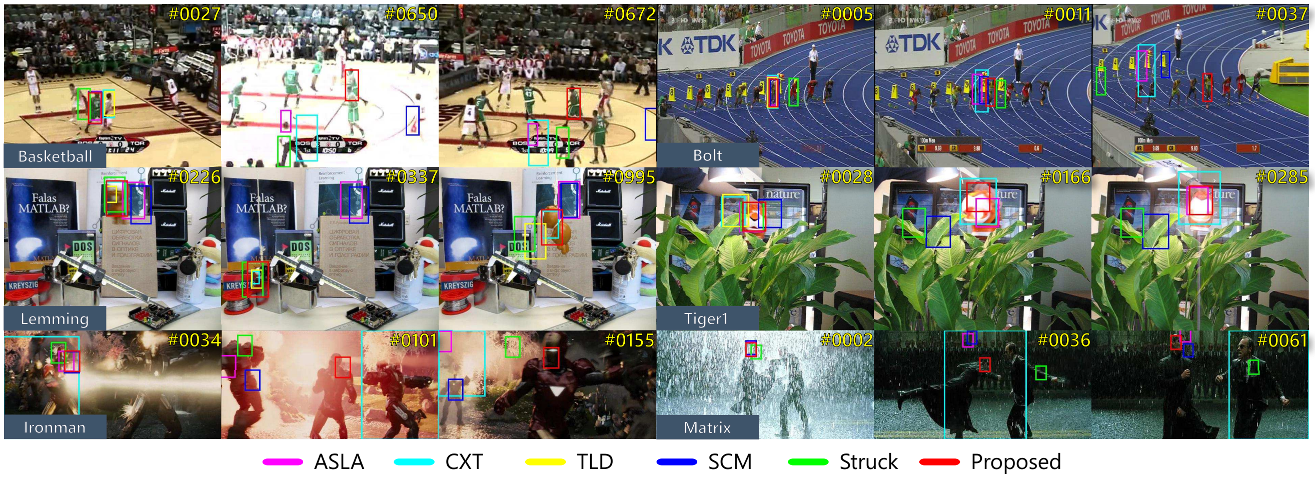

Figure 6. Comparison of tracking results of the proposed algorithm and the conventional methods: ASLA [23], CXT [11], TLD [24], SCM [42], and Struck [18].

LLP, and obtain the scores using the published codes or results for the other trackers. We observe that the proposed algorithm still achieves the best performance, as compared with these recent trackers. Also note that the proposed tracker is relatively simple, whereas MEEM and MTA require multiple base trackers to restore corrupted classifiers or false trajectories. Despite that, the proposed algorithm achieves performance gains of 6.4% in the PR score over MEEM and 6.2% in the SR score over MTA.

5.3. Results on VOT2014 Challenge Dataset

Finally, we evaluate the performance of the proposed algorithm on the VOT2014 challenge dataset [28]. Table 5 compares the proposed tracker with the top three trackers DSST [9], SAMF [1], and KCF [21] in the VOT2014 challenge. Two types of tests are carried out. In the baseline test, a tracker is initialized with a ground truth. On the other hand, in the region noise test, a tracker gets a perturbed ground truth.

It can be observed from Table 5 that the proposed tracker provides relatively low ACC scores, but achieves the best ROB performances in both tests. This can be explained by the re-initialization step in the evaluation methodology in [28]. When a tracking failure is detected, re-initialization is triggered. Then, a tracker obtains a new ground truth to trace a target object in remaining frames. However, each new ground truth tends to improve the overlap ratio. In other words, the top three trackers fail more frequently than the proposed tracker, they request more ground truths, and thus they yield higher overlap ratios. To confirm these effects of the re-initialization, we perform an additional experiment without the re-initialization step. The proposed tracker yields the best ACC scores without the re-initialization (ACC w/o) in Table 5.

Table 5. Comparison of the proposed tracker and the top three trackers in the VOT2014 challenge [28]. 'ACC w/o' denotes the ACC score without the re-initialization step. The best performance in each test is in boldface.

	Baseline			Region Noise		
	ACC	ROB	ACC w/o	ACC	ROB	ACC w/o
DSST [9]	0.622	1.160	0.469	**0.578**	1.283	0.432
SAMF [1]	0.616	1.280	0.504	0.572	1.435	0.484
KCF [21]	**0.629**	1.320	0.395	0.575	1.515	0.360
Proposed	0.575	**0.560**	**0.515**	0.552	**0.685**	**0.486**

6. Conclusions

In this work, we proposed an efficient object descriptor, called SOWP, to achieve accurate and robust visual tracking. We first divided a bounding box into multiple local patches, and extracted color and gradient histogram features at each patch. We then described the object appearance by concatenating the spatially ordered patch descriptors. We also assigned different weights to those patches adaptively according to their relevance to the object appearance, by performing the RWR simulations of the foreground and background random walkers. We thus suppressed the background information efficiently. We incorporated the SOWP descriptor into the structured output tracking framework [18]. Experimental results demonstrated that the proposed algorithm significantly outperforms all 29 trackers in the CVPR2013 benchmark [38], as well as the recent state-of-the-art trackers in [21, 9, 1, 39, 5, 22, 14, 40, 32]. Moreover, the proposed algorithm also excels on the VOT2014 challenge dataset.

Acknowledgements

This work was supported partly by the National Research Foundation of Korea(NRF) grant funded by the Ministry of Science, ICT & Future Planning (MSIP) (No. 2009-0083495), and partly by Samsung Electronics Co., Ltd.

References

[1] L. Agapito, M. M. Bronstein, and C. Rother. A scale adaptive kernel correlation filter tracker with feature integration. In *ECCVW*, 2014.

[2] S. Avidan. Support vector tracking. *TPAMI*, 26(8):1064–1072, Aug. 2004.

[3] S. Avidan. Ensemble tracking. *TPAMI*, 29(2):261–271, Feb. 2007.

[4] B. Babenko, M.-H. Yang, and S. Belongie. Robust object tracking with online multiple instance learning. *TPAMI*, 33(8):1619–1632, Aug. 2011.

[5] D. Chen, Z. Yuan, G. Hua, Y. Wu, and N. Zheng. Description-discrimination collaborative tracking. In *ECCV*, 2014.

[6] D. Chen, Z. Yuan, Y. Wu, G. Zhang, and N. Zheng. Constructing adaptive complex cells for robust visual tracking. In *ICCV*, 2013.

[7] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *TPAMI*, 25(5):564–577, May. 2003.

[8] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[9] M. Danelljan, G. Hager, F. Khan, and M. Felsberg. Accurate scale estimation for robust visual tracking. In *BMVC*, 2014.

[10] M. Danelljan, F. Khan, M. Felsberg, and J. van de Weijer. Adaptive color attributes for real-time visual tracking. In *CVPR*, 2014.

[11] T. B. Dinh, N. Vo, and G. Medioni. Context tracker: Exploring supporters and distracters in unconstrained environments. In *CVPR*, 2011.

[12] S. Duffner and C. Garcia. Pixeltrack: A fast adaptive algorithm for tracking non-rigid objects. In *ICCV*, 2013.

[13] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *TPAMI*, 32(9):1627–1645, Sep. 2010.

[14] J. Gao, H. Ling, W. Hu, and J. Xing. Transfer learning based visual tracking with gaussian processes regression. In *ECCV*, 2014.

[15] M. Godec, P. Roth, and H. Bischof. Hough-based tracking of non-rigid objects. *CVIU*, 117(10):1245–1256, Oct. 2013.

[16] H. Grabner, M. Grabner, and H. Bischof. Real-time tracking via on-line boosting. In *BMVC*, 2006.

[17] H. Grabner, M. Grabner, and H. Bischof. Semi-supervised on-line boosting for robust tracking. In *ECCV*, 2008.

[18] S. Hare, A. Saffari, and P. Torr. Struck: Structured output tracking with kernels. In *ICCV*, 2011.

[19] S. He, Q. Yang, R. Lau, J. Wang, and M.-H. Yang. Visual tracking via locality sensitive histograms. In *CVPR*, 2013.

[20] J. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *ECCV*, 2012.

[21] J. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *TPAMI*, 37(3):583–596, Mar. 2015.

[22] Z. Hong, C. Wang, X. Mei, D. Prokhorov, and D. Tao. Tracking using multilevel quantizations. In *ECCV*, 2014.

[23] X. Jia, H. Lu, and M.-H. Yang. Visual tracking via adaptive structural local sparse appearance model. In *CVPR*, 2012.

[24] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *TPAMI*, 34(7):1409–1422, Jul. 2012.

[25] J.-S. Kim, J.-Y. Sim, and C.-S. Kim. Multiscale saliency detection using random walk with restart. *TCSVT*, 24(2):198–210, Jun. 2013.

[26] T. H. Kim, K. M. Lee, and S. U. Lee. Generative image segmentation using random walks with restart. In *ECCV*, 2008.

[27] M. Kristan, R. Pflugfelder, et al. The visual object tracking VOT2013 challenge results. In *ICCVW*, 2013.

[28] M. Kristan, R. Pflugfelder, et al. The visual object tracking VOT2014 challenge results. In *ECCVW*, 2014.

[29] J. Kwon and K. M. Lee. Visual tracking decomposition. In *CVPR*, 2010.

[30] J. Kwon and K. M. Lee. Tracking by sampling and integratingmultiple trackers. *TPAMI*, 36(7):1428–1441, July 2014.

[31] D.-Y. Lee, J.-Y. Sim, and C.-S. Kim. Visual tracking using pertinent patch selection and masking. In *CVPR*, 2014.

[32] D.-Y. Lee, J.-Y. Sim, and C.-S. Kim. Multihupothesis trajectory analysis for robust visual tracking. In *CVPR*, 2015.

[33] X. Li, C. Shen, A. Dick, and A. van den Hengel. Learning compact binary codes for visual tracking. In *CVPR*, 2013.

[34] B. Liu, J. Huang, L. Yang, and C. Kulikowsk. Robust tracking using local sparse appearance model and k-selection. In *CVPR*, 2011.

[35] J.-Y. Pan, H.-J. Yang, C. Faloutsos, and P. Duygulu. Automatic multimedia cross-modal correlation discovery. In *ACM SIGKDD*, 2004.

[36] J. Supancic and D. Ramanan. Self-paced learning for long-term tracking. In *CVPR*, 2013.

[37] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 6:1453–1484, Sep. 2005.

[38] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *CVPR*, 2013.

[39] Y. Wu, M. Pei, M. Yang, J. Yuan, and Y. Jia. Robust discriminative tracking via landmark-based label propagation. *TIP*, 24(5):1510–1523, May. 2015.

[40] J. Zhang, S. Ma, and S. Sclaroff. MEEM: Robust tracking via multiple experts using entropy minimization. In *ECCV*, 2014.

[41] L. Zhang and L. van der Maaten. Preserving structure in model-free tracking. *TPAMI*, 36(4):756–769, Apr. 2014.

[42] W. Zhong, H. Lu, and M.-H. Yang. Robust object tracking via sparsity-based collaborative model. In *CVPR*, 2012.