Neurocomputing 239 (2017) 69–80

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Compressive perceptual hashing tracking

Long Chen[1,*], Zheng Li[1], Jian-Fei Yang

School of Data and Computer Science, Sun Yat-Sen University, Guangzhou 510006, China

ARTICLE INFO

Article history:
Received 24 December 2015
Revised 27 November 2016
Accepted 1 February 2017
Available online 16 February 2017

Communicated by Yi Zhe Song

Keywords:
Tracking
Perceptual hashing
Compressive sensing

ABSTRACT

This paper proposes a novel compressive sensing based perceptual hashing algorithm for visual tracking. A tracking object is represented by dimensionality reduced feature projected from perceptual hashing feature through a sparse measurement matrix. Besides, an updating weight map is assigned for each object and the weight map is updated according to the accumulation of foreground block and the distance between the foreground block and the center of the weight map. Based on above object representation and its weight map, our tracker searches the local region with the maximum similarity in coarse-to-fine way. In addition, we introduce a visual attention knowledge that the object, namely foreground, should be always located in the center of the weight map, to handle the model drift problem. Extensive experiments demonstrate that the proposed tracking method outperforms state-of-the-art methods in challenging scenarios and our tracker is especially insensitive to the location of the initial box.

1. Introduction

Visual object tracking is one of the key, components for numerous vision-based applications, such as assistant/autonomous driving system surveillance, object identification, and security. Although a number of trackers have been proposed, building a robust and fast visual tracker that can effectively handle all changes in appearance caused by illumination change, occlusion, viewpoints variation, scale change, and motion blur, remains a challenging problem.

Generative trackers and discriminative trackers are two main types of appearance-based trackers. The generative ones [1–5] use a particular feature vector or subspace model to present a target object and search for regions with the least reconstruction error from the target object. The discriminative trackers [6–11], namely tracking-by-detection, treat tracking problems as a local search detection problem based on a binary classifier. Generally, a discriminative with a prior data set can perform better but it would involve external training cost.

Wu et al. [12] presented a comprehensive evaluation of online trackers up until 2013. Since then, several effective trackers have been proposed. Locality Sensitive Histograms Tracker (LSHT) [13] is a simple and real-time tracking framework based on the locality sensitive histogram method, which is robust to illumination change. Given the assumption that the noise of reconstruction error is Gaussian–Laplacian distributed, an efficient iteration algorithm based on maximum joint likelihood of parameters is proposed in [14], to solve the Least Soft-threshold Squares Regression problem. In [15], a discriminative tracker based on weak SVM classifiers via randomized weighting vectors is introduced. To address the model drift problem, a multi-expert framework is chosen by [16], in which, an entropy-regularized restoration scheme is utilized to correct the undesirable effects of bad model updates for the base tracker. Extended Lucas–Kanade tracking (ELK) [17] combines template matching with pixel object and background segregation, which is resistive to drift as it disregards template background pixels strategically. Nevertheless, object tracking is still a challenging problem owing to appearance changes caused by illumination change, pose variation, occlusion, and so on.

Object representation, search mechanism, and model update are three crucial parts of a tracking problem. Numerous features and models have been chosen for object representation, some of which have a certain resistance to appearance change caused by illumination, viewpoints, or motion blur. For instance, Haar-like features [6,18,19], global integral histogram [20], locality sensitive histograms [13], sparse representation [1,21,22], adaptive color attributes [23] and weighted histogram representation [24]. Unlike a strategy that uses complex appearance models for attaining robustness in object representation, we choose to represent objects using a simple binary code constructed with the hashing technique, which is efficient and robust, because the Hamming distance is used for similarity estimation with the help of our binary feature and we did not need to train a complex classifier. Recently, hashing techniques have been widely applied to solve

* Corresponding author.
E-mail addresses: chenl46@mail.sysu.edu.cn, lchen.whu@gmail.com (L. Chen).
[1] Long Chen and Zheng Li have the equal contribution.

http://dx.doi.org/10.1016/j.neucom.2017.02.004
0925-2312/

the similarity calculation problem in many vision applications, such as image retrieval [25,26] and image search [27–29], owing to its efficiency and accuracy in data organization.

Perceptual hashing works well in various applications. However, until 2015, limited research had been conducted on hashing-based tracking [30,31]. In [31], Fei et al. present an object tracking approach using the perceptual hashing algorithm. After locating the tracking objects, they scan the next frame using the sliding-window method and present each window by perceptual hashing. Then, they measure the confidence between it and the tracking object using hamming distance. The perceptual hashing feature is unstable for violently shaking objects. To overcome the bottleneck of training hash functions with a large number of samples, which is time-consuming, they proposed a two-dimensional hashing method. During tracking, hash functions are updated using an incremental learning model. The 2D hashing used in [30] is a uniform-fragments based hashing feature, which means that there is no crossover between any two fragments. Besides, for one target, there are numerous fragments with different regions and different sizes. It is obviously a high-complexity and low-efficiency operation to extract features from all the fragments in a one-by-one way. We use a sparse measurement matrix that asymptotically satisfies the restricted isometry property (RIP) in compressive sensing theory, thereby facilitating efficient projection from the image feature space to a low-dimensional compressed subspace.

Compressive sensing theory [32] tells us that sufficiently high-dimensional signals can be represented as their low-dimensional random projections, which contain enough information to reconstruct the original high-dimensional signals. As a dimensionality reduction matter, compressive sensing has been introduced recently into visual tracking to achieve real-time performance [11,33]. In [33], a sparse measurement matrix is constructed to extract the efficient features from a multiscale image feature space. The tracking track is formulated as a binary classification by a naive Bayes classifier. Compressive tracker exhibits good performance for some tracking situations in real-time. However, the performance is unstable due to the shallow Haar feature, which cannot completely represent the object well and adaptively when encountering challenging scenarios such as illumination variation, deformation and etc, and the random mapping process scheme. Considering the complementary attributes of the perceptual hashing based tracker and compressive sensing based tracker, we present a novel tracking mechanism using the compressive tracking framework with perceptual hashing appearance model. For most current advanced tracking method, the object is restricted in a rectangle to represent its position. However, we know the rectangle must contain most proportion of the tracking object as well as a little information of the background context, sometimes causing drifting problem. To tackle the common conundrum, an idea is proposed that the tracking object is most likely to be fixed in the center of the rectangular tracking region. Consequently, we propose a foreground learning method to address the drift problem, which is not considered in their method.

The main components of the proposed compressive hashing tracking algorithm are shown in Fig. 1. For tracking, we extract our simple and robust perceptual features from multi-scale patches which are extracted randomly from the positive and negative samples. However, all of the scales and the positions should be considered, which will lead to high complexity. For computational efficiency, we introduce a sparse measurement matrix under the RIP condition in compressive sensing theory [34] for efficient projection from the image feature space to a low-dimensional compressed subspace. Finally, we generate our discriminative binary code representation by locality-sensitive hashing. Above all, we construct our object representation by Compressive Perceptual Hashing (CPH, Perceptual feature+Compressive sensing+LSH). In

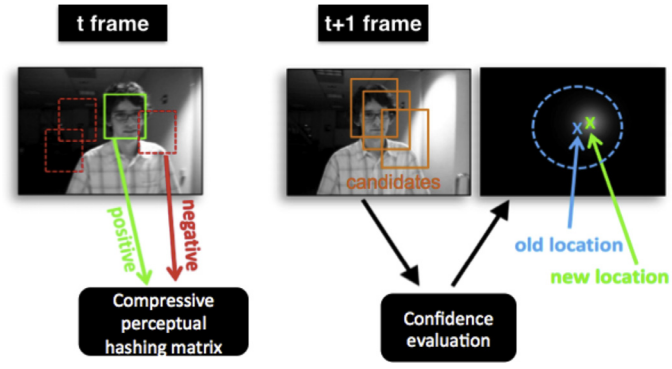


Fig. 1. Main components of the proposed compressive hashing tracking algorithm.

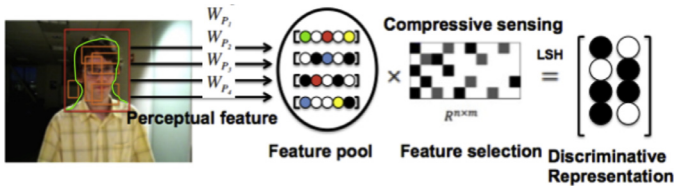


Fig. 2. Compressive perceptual hashing (CPH) representation.

the next frame, we sample a series of candidates surrounding the last object position via coarse-to-fine search method and extract CPH in the same way. Then, we generate the confidence map by calculating the similarity between candidate CPH and positive/negative CPH for each candidate, and select the location of the maximum confidence candidate as the new location in the next frame. Besides, it is important to focus on foreground information of actual object instead of background in the tracking box, thus we use improved Gaussian Mixture Model (GMM) [35] to extract foreground of tracking box and take patch foreground weight into consideration to make confidence evaluation more robust, which is shown in Fig. 2. The proposed compressive tracking algorithm runs at real-time and performs favorably against state-of-the-art trackers on challenging sequences in terms of efficiency, accuracy, and robustness.

This paper extends our previous work [36] by including a comprehensive description of the introduction, methodology and more thorough experiments. The contributions of the proposed tracking framework are as follows.

- We design a novel discriminative binary object representation method for visual tracking by combining perceptual hashing method with compressive sensing. The proposed perceptual hashing method avoids model training and achieves state-of-the-art performance with comparable computation speed among several discriminative methods.
- We assign each CPH template with an online learning foreground weight map to handle the model draft problem under challenging scenarios, which enhances the robustness of confidence evaluation.
- A visual attention knowledge, that the object should always lie in the center of the tracking box, is first imported into the tracking framework to further solve the model drift problem.

The rest of this paper is organized as follows. Section 2 describes three crucial components, discriminative compressive object representation, visual tracking framework, and foreground learning. Experimental results and comparisons are shown in Section 3. Section 4 concludes the paper.

2. Compressive perceptual hashing tracking

In this section, we present the proposed compressive perceptual hashing tracking algorithm as well as the weighted map updating by online foreground learning. The flowchart of our tracking framework is shown in Fig. 1. Originally, the task of visual tracking is to accurately detect the designated object in the sequence. The initialized tracking window is assigned a manual label at the first frame, usually the ground truth. For each of the following frame, we sample a suitable quantity of positive samples near the current object center and negative samples away from the target. Each sample is represented by compressive perceptual hashing features of several multi-scale fragments by means of discriminative sparse representation. To obtain the object location in the next frame, we detect a series of candidates in the search region by coarse-to-fine search around the current location and determine the one with the maximal confidence level.

2.1. Discriminative compressive object representation

2.1.1. Random projection and compressive sensing

To obtain complete object appearance representation, we sample many multi-scale fragments covering different locations and sizes among the object, which can cause a high dimensional representation. Besides, the feature extraction costs too much time and is prohibited from real-time visual tracking. Therefore, it is necessary to reduce the high-dimensional signal to lower-dimensional space without losing the significant information at the same time. To achieve this, we apply the recent significant concept of random projection [37] and compressive sensing [34]. In random projection, a random matrix $\mathbf{R} \in \mathbb{R}^{n\times m}$ whose rows have unit length projects feature from the high-dimensional space $\mathbf{x} \in \mathbb{R}^m$ to a lower-dimensional space $\mathbf{v} \in \mathbb{R}^n$

$$\mathbf{v} = \mathbf{R}\mathbf{x} \tag{1}$$

where $n \ll m$. Each projection $\mathbf{v}$ is equivalent to a compressive measurement in the compressive sensing encoding stage. In our algorithm, $\mathbf{x}$ is composed of features of all fragments while $\mathbf{v}$ merely consists of certain fragments. The compressive sensing theory [34] affirms that it is possible to reconstruct the signal from a small number of random measurements if a signal is K-sparse. The sparse measurement matrix $\mathbf{R}$ preserves the salient information in any K-sparse signal when projected. Let $\mathbf{R} \in \mathbb{R}^{n\times m}$ be a random matrix with $\mathbf{R}(i, j) = r_{i,j}$ where

$$r_{i,j} = \sqrt{3} \times \begin{cases} +1 with\ probability\ \frac{1}{6} \\ 0\ with\ probability\ \frac{2}{3} \\ -1 with\ probability\ \frac{1}{6} \end{cases} \tag{2}$$

which is generated in the initialization, and the RIP [38,39] for it is definitely necessary. The fragments at current are suitable for feature extraction at an excellent speed with little loss.

2.1.2. Patch-based appearance model

For each positive T_s^+ and negative sample T_s^- at a frame of time t, we adopt patch strategy based on multi-scale fragments to construct distinguishable representation. Each fragment is divided by canonical patches with the length of l x l. The positions and sizes of fragments are specified in the initialization randomly, which are same for all samples in the frame sequence. Due to the random multi-scale fragments, there are several patches that completely overlap among different fragments. It is a trick to avoid extract features from the same patches repeatedly. Therefore, we record the same patches in the initial step and extract features from them just once to improve efficiency. Then, all patches are resized to 8 * 8 to generate the perceptual hashing feature. By doing this, we further simplify the later stages of the procedure without losing too much of the structural information of the image, and we also gain some measure of scale invariance.

2.1.3. Perceptual feature

In general, perceptual feature represents the characteristics of human vision. The research of cognitive psychology and human visual system demonstrate that human eyes are sensitive to illumination, color, and texture information. Making an analogy with eyes and image representation, we construct our perceptual feature by means of intensity histogram and Discrete Cosine Transform (DCT) based on illumination and contour profile. The perceptual representation of a sample consists of features of patches. Here come the two kinds of perceptual features, intensity histogram, and low-frequency energy spectrum.

The intensity histogram for a patch P is a 1D array, each of whose value is an integer representing the frequency of occurrence of particular intensity values. The corresponding histogram H_P is a B-dimensional vector defined as:

$$H_P(b) = \sum_{i=1}^{N} C(I_i, b), \quad b = 1, 2, \ldots, B \tag{3}$$

where N denotes the number of pixels and B is the total number of bins. Here, we set $B = 8$. $C(I_i, b)$ is a binary function whose output is zero except when intensity value I_i belongs to bin b. Now that the local histogram H_P indicates grey information of a patch P. The computational complexity of Eq. (3) is linear in the number of bins at each pixel location: $O(B)$. However, the addition operation can be ignored when $C(I_i, b) = 0$, the complexity is reduced to $O(1)$. Inductively, the computational complexity of the whole image is the same as that of the local histogram.

A discrete cosine transform (DCT) expresses a finite sequence of data points in terms of a sum of cosine functions oscillating at different frequencies. It has a strong "energy compaction" property and concentrates most of the signals in a few low-frequency components of the DCT. We link all 2-dimensional patches to a vector and introduce a one-dimensional DCT to obtain the reliable results. The process of a DCT can be defined as:

$$d_m = \sum_{i=1}^{64} y_i \cos\left[\frac{\pi}{8} m\left(i + \frac{1}{2}\right)\right], m = 1, 2, \ldots, 64 \tag{4}$$

Based on Eq. (4), we can conclude that low-frequency information is concentrated in the frequency spectrum. Therefore, that the rest of the DCT coefficients prove to be useless, just eight values extracted from the upper left corner are considered as the perceptual feature of profiles. Therefore, the 8-dimension vector D_P denotes the low-frequency feature of a patch P. The perceptual features of a patch P is the combination of them: $v_P = [H_P D_P]$. Thus far, the entire perceptual feature matrix of a sample have been extracted:

$$V = \left[v_{P_1} v_{P_2} \ldots v_{P_k}\right] \tag{5}$$

where k denotes the total number of patches. With the help of DCT, we can get the sparsity in the image which enables the possibility of the use of compressive sensing while preserving the information in the image. By DCT, we can select 8 discriminative values in the upper left corner of DCT feature matrix where the low-frequency information gathers.

2.1.4. Binary matrix generation by locality-sensitive hashing

To compare different perceptual feature of images conveniently, the perceptual features matrix is necessarily processed using the hashing method, one of which is locality-sensitive hashing (LSH), widely used in similarity search [40]. For each patch feature $v_P \in \mathbb{R}^{16}$, we construct m number of corresponding hashing functions, one of which is define as follows:

$$h_{P_i}(v_P) = sign(w_{P_i}^{\mathrm{T}} v_P + b) \tag{6}$$

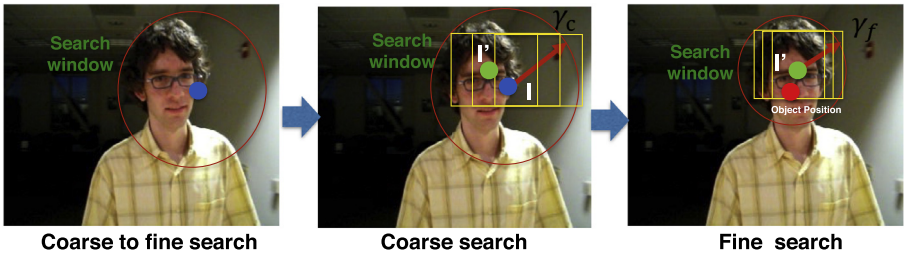


Fig. 3. The principle of coarse-to-fine search.

where $w_{P_i} \in \mathbb{R}^{16}$ is generated randomly between $[-1, 1]$ satisfying *Gaussian distribution* and b is set to 0. Each patch can generate m binary codes through m hashing functions $\{h_{P_1}(\cdot), h_{P_2}(\cdot), \ldots, h_{P_m}(\cdot)\}$, which just needs a matrix multiplication operation ${W_p}^{\mathrm{T}} v_P$. However, for several patches it will cost much. Therefore, we share the same W_P and just need a matrix multiplication operation, which can enhance computing efficiency significantly. Through LSH, we generate a unique perceptual image representation (PIR) for each sample and can further conduct confidence evaluation.

2.2. Visual tracking with PIR

Now that the perceptual hashing feature is extracted, we build two main models to measure the difference and establish the learning strategy.

2.2.1. Appearance model

Given object O at frame $t-1$ and current frame t, a tracking algorithm is required to locate the object O at t frame. In the former description, we clarify that O is presented by positive templates $\{T_s^+\}_{s=1}^{T^+}$ and background G is represented by negative templates $\{T_s^-\}_{s=1}^{T^-}$. The tracking task involves finding the target location of a candidate C in the t frame, which is most similar to positive templates but most dissimilar to negative templates. Between candidates and templates, we define the similarity measurement by hamming distance:

$$d(C, T_s) = \|\mathbf{h}(C) - \mathbf{h}(T_s)\|_H \tag{7}$$

where $\|\cdot\|_H$ is the hamming distance and $\mathbf{h}(\cdot)$ denotes hash functions for samples. The results of hash functions are binary code matrix of samples and hamming distance between them can be measured by bit-wise XOR operation. Whereas, the confidence level between candidates and templates should be more discriminative and based on similarity measurement as well as patch foreground weight. We convert the hamming distance to similarity level as $1 - \frac{1}{l^2}\|\mathbf{h}(P_i^C), \mathbf{h}(P_i^{T_s})\|_H$ where l is the size of the patch, P_i^C and $P_i^{T_s}$ denote the ith patch of candidate C and template T_s. Then, similarity measurement 7 is improved by:

$$s(C, T_s) = \sum_{i=1}^{k} W_{P_i}^t \left(1 - \frac{1}{l^2}\|\mathbf{h}(P_i^C), \mathbf{h}(P_i^{T_s})\|_H\right) \tag{8}$$

where $W_{P_i}^t$ represents the foreground weight of the ith patch at the t frame. The foreground weight, accounting for the weight of a patch in a candidate, will be introduced in Section 2.3. Comprehensively, the confidence of the jth candidate C_j is defined by the average distance to all fragments as follows:

$$Con(C_j) = \delta\left(\frac{1}{T^+}\sum_{s=1}^{T^+} s(C, T_s^+) - \frac{1}{T^-}\sum_{s=1}^{T^-} s(C, T_s^-)\right) \tag{9}$$

where δ is a normalized coefficient that sets a confidence constraint of $[-1, 1]$. The proposed appearance model summarizes samples and objects of the consecutive frames, which offers convenient and efficient conditions for tracking updating.

2.2.2. Tracking model

The task of the tracking model is to update the object location at current frame in virtue of PIR and appearance model. Thus, we generate the search candidates in a coarse-to-fine way and construct the tracking model by a Bayesian framework.

Coarse-to-fine search: as we know the ground truth I_1 at the first frame, we define the positive sample $D^a = \{Z | \|I(Z) - I_1\| < \alpha\}$ and negative sample $D^{\beta,\xi} = \{Z | \xi < \|I(Z) - I_1\| < \beta\}$, where Z denotes the sample position and α, β, ξ denote the sample distance threshold as $\alpha < \xi < \beta$. At the second frame, we choose the candidates as $D^{r^c} = \{Z | \|I(Z) - I_1\| < r^c, \Delta_c\}$ where r^c is the coarse radius and Δ_c is the coarse shifting step. Then, we calculate the confidence of each candidate and select the maximal response one as the center of the fine search circle I_1'. The fine search runs as $D^{r^f} = \{Z | \|I(Z) - I_1'\| < r^f, \Delta_f\}$ where r^f is the fine radius and Δ_f is the fine shifting step. After maximum confidence selection, we regard the results of fine search as the object position at the second frame. From then on, the search is executed repeatedly in the same manner in Fig. 3.

Bayesian framework: to address the problem of visual tracking process, Bayesian framework is utilized by means of probability theory. Firstly we define a state space model as measurement, and then estimate the most probable state, namely posterior probability, according to all measurements. Given a sequence, state $X_t = (x_t^h, x_t^v)$ is represented by the horizontal and vertical position. Y_t represents observations at frame t. We also construct the observation model as $Y_t = (y_t^h, y_t^v)$. According to the Bayesian theorem, we assume that X_t conforms to first order Markov process and the measurement sequences Y_t are mutually independent. From the aspect of Bayesian estimation, the state X_t is obtained by means of the observation and the calculation of the posterior. The posterior of the state at given time t is calculated by the equation of state prediction:

$$p(X_t | Y_{1:t-1}) = \int p(X_t | X_{t-1}) p(X_{t-1} | Y_{1:t-1}) \mathrm{d}X_{t-1} \tag{10}$$

where $Y_{1:t-1}$ denotes the observation from the first frame to the previous frame. The equation of state update is defined as:

$$p(X_t | Y_{1:t}) = \frac{p(Y_t | X_t) p(X_t | Y_{1:t-1})}{p(Y_t | Y_{1:t-1})} \tag{11}$$

where $Y_{1:t}$ denotes the observation from the first frame to the current frame and some argument of process is calculated by:

$$p(Y_t | Y_{1:t-1}) = \int p(Y_t | X_t) p(X_t | Y_{1:t-1}) \mathrm{d}X_t \tag{12}$$

Once the posterior probability is calculated, the state X_t is able to be predicted. As for our algorithm, we construct a sensible effective observation model as follows:

$$p(Y_t | X_t) \propto Con(X_t) \tag{13}$$

where $Con(X_t)$ is the confidence of state X_t defined by Eq. (9). Apart from the observation model, we need to assume that the motion model $p(X_t|X_{t-1}$ presented by Eq. (10) caters to Gaussian distribution for both location parameters:

$$p(X_t|X_{t-1}) = N(X_t; X_{t-1}, \Phi) \tag{14}$$

where Φ denotes the diagonal covariance matrix and the elements of (σ_x, σ_y) of the matrix are standard deviations of the X axis location and Y axis location. X_{t-1} stands for previous state of the target (x_{t-1}, y_{t-1}), which are the mean values of next frames for the Gaussian model. These statistic elements are macroscopical in nature and explain the change of location. At current frame, the estimate of the target $\hat{X}_t$ is defined by the MAP estimate over M samples:

$$\hat{X}_t = argmax_{X^{(i)}_t} p(X_t^{(i)}|Y_{1:t}), \forall i \in [1, M] \tag{15}$$

where $X_t^{(i)}$ denotes the ith sample's state. Through the similarity measurement of candidates and fragments and probability selection, the most possible position of candidates is assured.

2.3. Foreground learning for object tracking

For most current advanced tracking method, the object is restricted in a rectangle to represent its position. However, we know the rectangle must contain most proportion of the tracking object as well as a little information of the background context, sometimes causing drifting problem. To tackle the common conundrum, an idea is proposed that the tracking object is most likely to be fixed in the center of the rectangular tracking region. Here comes the detailed implementation of our idea.

2.3.1. Weighted map updating

In the tracking region of the current frame t, the foreground is extracted and recorded in weighted map W, all of whose elements, totally the same quantity of pixels, are initialized to 1 at the beginning. As presented in [35], Zivkovic makes contributions on an improved adaptive Gaussian mixture model for background subtraction, which extracts the foreground under multivariate circumstances. With the simple method, we obtain the region of foreground and compute its pixel center as $F_t(x_f, y_f)$ at frame t. The weighted center K_t is updated by W^t. For each pixel $K_p(x, y)$ of a patch in the tracking region, we formulate its current foreground weight at frame t as follows:

$$w_{K_p}^t = \varepsilon e^{-\frac{\|K_p - K_t\|^2}{\sigma^2}} \tag{16}$$

which is a radial basis function (RBF) satisfying Gaussian distribution, where ε is the normalized coefficient and K_p denotes the position of a pixel in the tracking region, σ^2 stands for the object occupation in the tracking region. Bigger the σ, the more attention field the tracking region embodies. Naturally, we compute the current foreground weight of the patch P_i at frame t by the mean of its region of weighted map:

$$\bar{W}_{P_i}^t = \frac{1}{l^2} \sum_{Kp \in P_i} w_{K_p}^t \tag{17}$$

where l is the size of the patch. Then, the continuous frame are relevant, so the updating is formulated as:

$$W_{P_i}^t = (1 - \lambda) W_{P_i}^{t-1} + \lambda \bar{W}_{P_i}^t \tag{18}$$

where λ denotes the learning rate, $W_{P_i}^t$ and $W_{P_i}^{t-1}$ stands for accumulated foreground weight at frame t and $t-1$, respectively, and the weighted map $W_{P_i}^t$ is used momentously in Eq. (8). The learning rate balances the current and accumulated weight while the weighted map merits the different weight contribution of distinct patch to PIR. An example of the online learning foreground weight map is shown in Fig. 4.

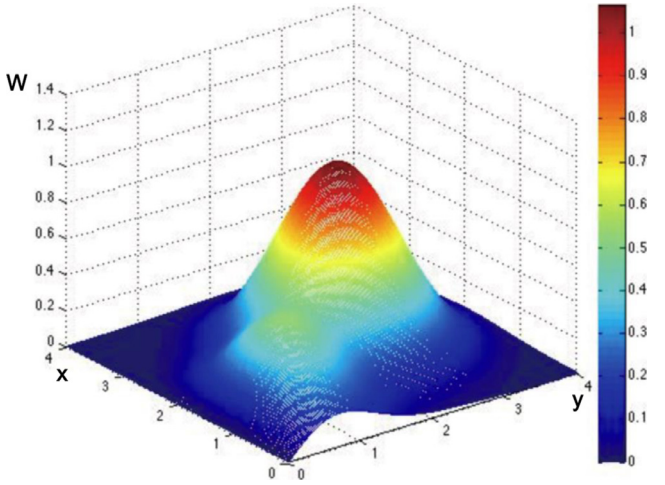


Fig. 4. One example of the online learning foreground weight map.

2.3.2. Periodic center rectification

To handle the model drift problem, we introduce a visual attention knowledge that the object, namely foreground, should be always located in the center of the weight map. Based on this knowledge, we design an operation called periodic center rectification by using the weighted map. Every τ frame, the deviation of object center may occur and we can eliminate the offset by defining a translation vector $\vec{Z}_t$ at current frame t:

$$\vec{Z}_t = \vec{K}_t - \vec{F}_t \tag{19}$$

It is obvious that the vector defined directs the real-time foreground center to the weighted learning center. It handles the problem of center shifting periodically. To sum up, we describe the whole algorithm in Algorithm 1.

Algorithm 1 Compressive perceptual hashing tracking.

Input: the t-th image frame

1: Coarsely sample a set of object candidates patches in D^{r^c} where l_{t-1} is the tracking location at the $(t-1)$-th frame by shifting in pixel step Δ_c and extract compressive perceptual hashing features.
2: Use confidence in Eq. (9) and Bayesian framework in Eq. (10) to each feature vector $V(Z)$ and find the tracking location l'_t with the maximal response.
3: Finely sample a set of object candidate patches in D^{r^f} by shifting in pixel step Δ_f and extract compressive perceptual hashing features.
4: Use confidence in Eq. (9) and Bayesian framework in Eq. (10) to each feature vector $V(Z)$ and find the tracking location l_t with the maximal response.
5: Sample positive samples set D^a and negative sample set $D^{\beta,\xi}$ with $\alpha < \xi < \beta$, respectively, and extract compressive perceptual hashing features with these two sets of samples.

Output: Tracking location l_t and two sets of samples PIR.

3. Experimental results

In this section, we evaluate our proposed tracking approach denoted as *Compressive Perceptual Hashing Tracking* (CPH) by means of both quantitative and qualitative experiment. To evaluate the performance of our tracking algorithm in various scenarios, we conducted experiments on 24 representative sequences (boy, couple, david, david2, deer, dog1, doll, dudek, FaceOcc1, FaceOcc2, fish, fleetFace, Football, Football1, freeman1, freeman3, Ironman,

Table 1
Comparisons with advanced trackers on the 24 benchmark sequences. Our method performs favorably against other methods in distance precision (DP) at 20 pixels, overlap success rate at 0.5 (OS), center location errors (CLE). The first and second highest values are highlighted by red and blue fonts.

	Ours	BSBT	Frag	KMS	LOT	LSK	MIL	MS	OAB	PD	SMS	Struck	VR	VTD	CT	FCT
DP (%)	**89.30**	47.60	70.10	61.30	67.30	72.80	75.40	44.60	63.80	50.80	54.90	**89.10**	45.60	87.00	72.90	70.80
OS (%)	**76.50**	46.20	59.60	45.50	55.90	69.10	53.60	28.20	59.80	35.20	27.70	76.00	36.10	**76.90**	61.80	55.80
CLE (pixel)	**12.07**	155.53	47.62	51.52	43.89	52.27	31.30	72.11	52.00	77.37	63.84	**16.49**	75.29	27.53	47.31	36.92
Speed (FPS)	20.24	7.00	6.30	3.16	0.70	5.50	38.10	31.08	22.40	10.91	19.20	20.20	12.26	5.70	**64.40**	**76.42**

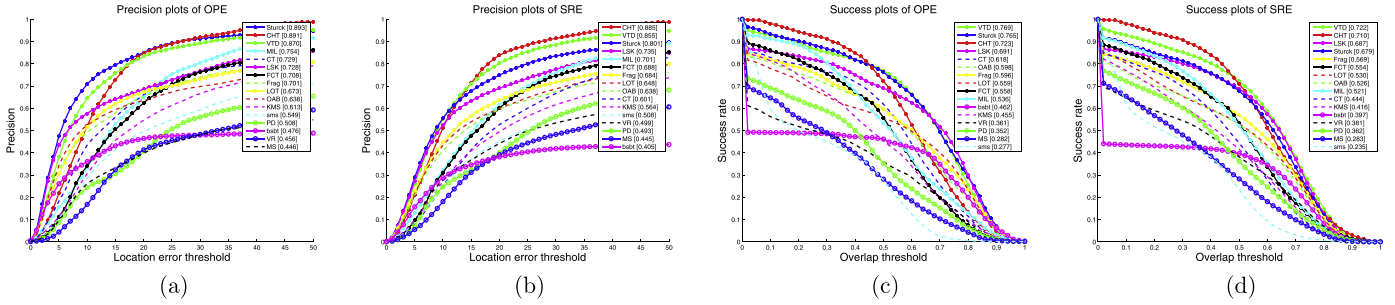


Fig. 5. Plots of OPE and SRE. The performance score for each tracker is shown in the legend. Precision and success ratio are measured by center location errors and bounding box overlap ratios, respectively. The ranks are set with center location error 25 and overlap ratio 0.5.

jumping, Mhyang, shaking, Singer1, Soccer, Sylvester, Trellis) in the tracking benchmark dataset [12] and compared it with other 15 state-of-the-art methods. The sequences from the benchmark dataset contain various challenges such as illumination variation, background clutter, occlusion, abrupt target motion, and rotation. The 15 evaluated trackers are the structured tracker (Struck) [8], visual tracking decomposition (VTD) algorithm [4], the multiple instance learning tracker (MIL) [41], the real-time compressive tracker(CT) [11], local sparse appearance model and k-selection (LSK) [3] method, the fast compressive tracking (FCT) [33], the fragment tracker (Frag) [20], the locally orderless tracker (LOT) [42], online AdaBoost method (OAB) [43], kernel-based tracker (KMS) [44], mean-shift Blob tracking through scale space (SMS) [45], Beyond semi-supervised tracker (BSBT) [46], online selection of discriminative tracking features (VR) [47], peak difference tracker (PD), and the mean shift tracker (MS). Among them, OAB, BSBT, MIL, CT, FCT and Struck belong to discriminative models and VTD, LSK belong to generative models.

3.1. Experimental setup

All of the experiments are performed on a PC with Intel i7 3770 CPU (3.4 GHz). The initial parameter of tracking region is set as $\varrho = 64$ and $\tau = 8$. The intensity histogram parameter B is 8. The process of updating using foreground information is executed every 5 frames ($\epsilon = 5$). During the experiment, the search radius for drawing positive samples α is set to 4, generating 45 positive samples. The inner ξ and outer radius β for the set $D^{\xi,\beta}$ that generates negative samples are set to 8 and 30. The search radius r^c for the set D^{r^c} is 30 and the shifting step Δ_c is 4. The radius of fine-grained search is set to 10 and Δ_f is set to 1. The learning rate δ is set to 0.85 and the σ is set to 1 while the foreground rectification period τ is executed every 5 frames.

3.2. Overall performance

We present the quantitative comparisons of distance precision (DP) at 20 pixels, overlap success rate at 0.5 (OS), center location errors (CLE) and tracking speed (FPS) in Table 1. Among the trackers in the literature, our method achieves the best results with an average DP of 89.3% and Struck achieves a little lower DP of 89.1%. Conclusively, our algorithm performs the best in 2 metrics with OS of 76.5% and CLE of 12.07 pixels. VTD also performs well with an average OS of 63.44%. We can observe that our algorithm has best DP and CLE among all the discriminative models and generative models, which indicates the proposed CPH feature can maintain most key information for tracking with low feature dimensionality.

We also conduct the experiment in a conventional way of running trackers throughout a test sequence with initialization from the ground truth position in the first frame, which reports the average precision. In Fig. 5, we refer this to one-pass evaluation (OPE), which means run them throughout a test sequence with initialization from the ground truth position in the first frame and report the average precision or success rate, and it is evident that our method performs excellent with distance precision as well as success rate. Moreover, to present the advantages of foreground learning, we conduct the spatial robustness evaluation (SRE) where different initial ground truth are set. We see that the results of SRE reveal our outstanding performance and the prominent function of foreground learning.

3.3. Attribute-based evaluation

The sequences in the experiment are annotated with 9 attributes to describe the different challenges in the tracking problem such as fast motion and background clutter. These attributes evaluate the performance of trackers in different aspects. We report eight main challenges in Fig. 6. Among the evaluated methods, the Struck method performs well with distance precision in background clutter (82.9%), fast motion (80.6%), and occlusion (85.9%) while our method achieves the same attribute of 86.0, 76.2, and 83.9%. The VTD method performs well in in-plane rotation (89.4%), out-of-plane rotation (90.9%), illumination variation (92.5%), and scale variation (87.3%) while our method performs mostly better with 89.2, 90.9, 96.5, and 88.6%. Regarding all these attributes, our method performs the best results in 5 attributes and good results in others. This result shows that the proposed binary CPH feature has better anti-interference ability comparing

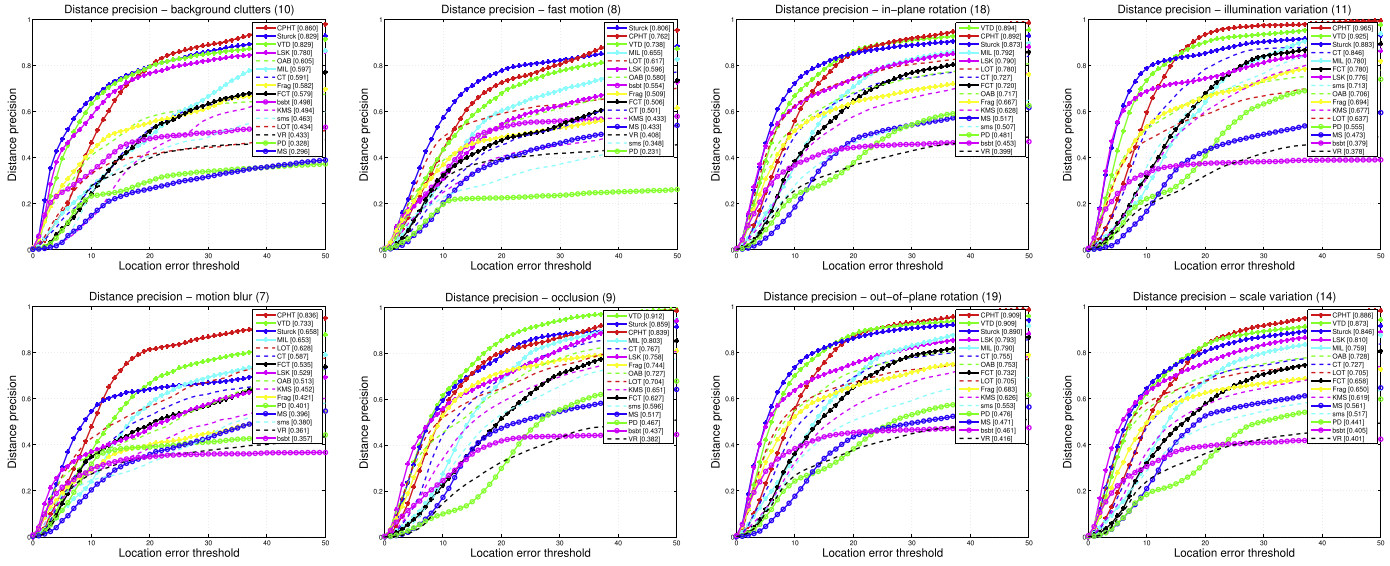


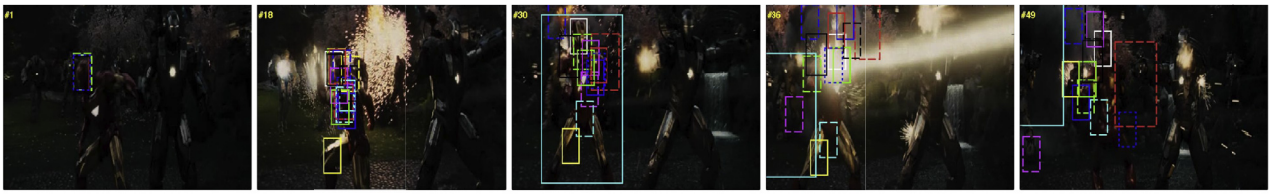
Fig. 6. Comparisons of distance precision over eight tracking challenges of background clutter, fast motion, in-plane rotation, illumination variation, motion blur, occlusion, out-of-plane rotation, and scale variation. The legend contains the DP score within 20 pixels for each tracker. The proposed method performs favorably against others.

Table 2
Center location error (CLE)(in pixels)(average). Red fonts indicate the best performance while the blue fonts indicate the second best ones.

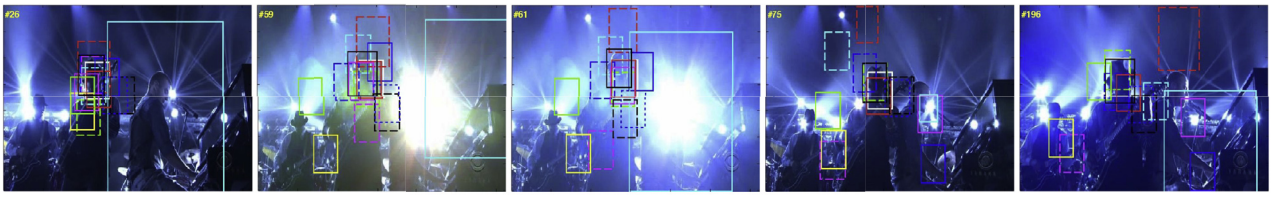
Sequence	Ours	BSBT	Frag	KMS	LOT	LSK	MIL	MS	OAB	PD	SMS	Struck	VR	VTD	CT	FCT
Boy	7.33	115.73	35.70	4.49	32.87	161.01	10.80	8.48	2.67	50.68	3.40	3.55	5.22	7.50	7.27	9.18
Couple	9.04	145.27	36.80	107.94	37.09	129.02	34.53	135.95	57.62	81.17	19.40	11.33	111.19	104.25	36.39	40.55
David2	5.05	87.83	56.89	36.68	6.03	16.54	10.63	68.05	34.87	38.32	60.27	11.49	38.42	12.89	76.36	18.89
Deer	5.29	219.05	105.09	43.88	65.22	98.77	100.73	215.75	6.99	158.02	84.94	5.87	196.74	134.85	246.42	18.89
Dog1	15.64	71.18	11.92	25.17	4.63	7.28	7.83	38.06	5.90	19.57	54.93	5.66	19.57	11.04	6.99	9.69
Doll	7.40	191.83	10.94	19.13	14.58	13.19	12.45	13.79	17.91	23.38	11.71	15.49	78.55	16.47	11.88	16.00
Dudek	19.24	42.76	82.69	81.25	85.13	19.61	19.70	85.24	31.38	185.10	104.19	20.45	138.29	20.30	26.53	34.32
FaceOcc1	25.12	46.25	10.97	26.16	34.75	30.41	29.86	32.71	27.66	27.55	29.78	28.78	25.47	30.20	25.82	35.67
FaceOcc2	9.77	102.71	15.95	28.94	14.89	14.57	13.60	77.62	19.58	62.76	39.68	5.96	62.32	10.29	18.95	15.85
Fish	11.91	105.62	21.56	29.41	33.62	50.81	24.14	37.33	87.03	27.31	28.06	3.40	27.09	16.79	12.68	13.04
FleetFace	23.57	164.24	67.66	126.84	33.74	36.40	63.12	123.50	52.10	113.47	113.31	24.01	112.22	46.13	58.43	55.64
Football	13.53	245.20	15.36	105.55	16.55	14.02	22.09	64.24	72.37	238.05	217.24	17.31	269.49	13.64	14.91	16.30
Football1	5.03	137.30	15.70	17.05	6.85	58.22	5.62	20.66	85.68	5.96	6.80	5.47	5.91	7.44	20.71	26.42
Freeman1	12.23	184.93	10.10	30.98	86.91	111.98	11.22	34.69	35.72	98.21	155.26	14.27	96.47	10.32	118.72	117.53
Freeman3	6.64	152.14	8.83	63.74	40.53	39.50	87.56	59.07	40.66	45.61	33.55	16.83	47.12	23.96	65.32	67.44
Ironman	10.20	217.68	120.90	80.31	60.47	33.79	42.40	57.63	121.18	45.52	94.86	41.45	44.30	24.76	41.82	55.96
Jumping	8.25	191.52	8.60	93.97	5.57	74.62	9.99	28.94	46.35	6.44	37.45	6.55	46.71	41.39	47.73	38.59
Mhyang	9.39	58.92	12.51	19.08	113.07	9.43	20.40	114.30	9.40	56.29	14.39	2.59	56.29	14.36	13.28	14.67
Shaking	18.09	275.09	192.15	59.17	82.60	25.52	24.03	227.89	191.56	166.84	181.71	30.70	151.05	9.05	80.02	36.83
Singer1	11.63	304.55	88.87	53.09	141.39	20.76	16.37	108.69	12.93	44.60	18.70	14.51	27.19	14.19	15.53	19.31
Soccer	22.00	351.00	95.36	81.25	42.24	167.13	77.85	46.46	127.53	219.73	107.98	71.36	59.38	23.56	92.40	136.60
Sylvester	8.11	64.56	11.33	14.43	7.59	54.81	10.21	23.68	13.66	12.33	12.62	4.73	58.10	5.42	5.67	6.60
Trellis	13.25	101.81	59.36	36.34	43.21	14.82	64.79	35.89	95.34	52.58	38.16	17.45	54.61	34.40	44.24	45.12
Average CLEs	12.07	155.53	47.62	51.52	43.89	52.27	31.30	72.11	52.00	77.37	63.84	16.49	75.29	27.53	47.31	36.92

Table 3
Success rate (SR)(%) and average FPS(frame). Red fonts indicate the best performance while the blue fonts indicate the second best ones.

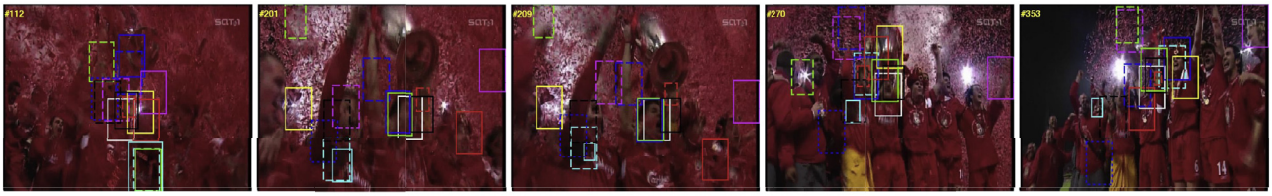
Sequence	Ours	BSBT	Frag	KMS	LOT	LSK	MIL	MS	OAB	PD	SMS	Struck	VR	VTD	CT	FCT
Boy	81.27	42.36	53.27	87.64	60.36	45.64	34.55	59.09	100	68.55	91.27	99.45	97.27	76.91	32.18	61.64
Couple	72.66	12.95	59.71	8.63	57.55	8.63	68.35	7.19	43.88	3.6	21.58	67.63	7.91	30.94	10.79	32.37
David2	100.00	15.67	75.93	9.51	67.35	72.2	29.85	5.41	79.66	58.58	0.37	100	58.58	97.95	19.00	49.81
Deer	100	41.43	5.71	64.29	2.86	17.14	48.57	27.14	98.57	21.43	2.86	98.57	7.14	4.29	4.29	51.43
Dog1	88.77	56.86	57.23	2.82	50.63	84.21	65.09	5.19	58.19	48.18	4.6	61.38	49.22	60.71	62.19	57.67
Doll	78.7	16.91	60.17	50.26	83	99.7	8.78	37.48	28.39	20.43	49.74	59.48	54	83.22	78.22	38.26
Dudek	79.63	73.08	58.3	58.3	61.8	65.73	39.16	59.97	50.7	18.18	5.24	85.14	49.13	97.73	76.10	66.08
FaceOcc1	85.3	79.01	99.89	90.12	29.18	41.98	61.05	57.91	9.54	100	68.8	99.1	100	83.05	34.12	64.98
FaceOcc2	91.49	54.01	63.26	13.93	36.74	70.04	87.05	1.48	72.38	49	0.12	98.77	25	96.79	59.43	88.66
Fish	100	30.74	37.47	33.26	10.32	28.21	51.58	15.79	18.74	59.79	0.84	100	56.21	90.32	78.53	87.16
FleetFace	78.33	54.53	55.67	19.69	53.4	68.56	74.36	7.65	66.43	3.54	0.14	82.01	3.54	70.82	65.58	75.21
Football	78.93	22.14	79.29	9.64	71.43	95.36	99.64	13.93	70.36	5	0.36	67.86	5	98.57	29.29	70.00
Football1	67.57	8.11	29.73	33.78	67.57	31.08	56.76	22.97	14.86	74.32	2.7	32.43	74.32	59.46	1.35	6.76
Freeman1	71.23	20	18.46	8.92	8	27.08	7.38	1.23	23.38	3.38	5.54	25.23	2.77	25.23	13.54	1.23
Freeman3	28.15	19.06	35	13.13	11.56	21.25	2.5	2.81	17.5	16.56	4.06	9.38	8.44	18.44	0.31	0.30
Ironman	68	2	12	14	28	38	16	26	10	10	26	12	8	52	30.00	32.00
Jumping	72.5	21.15	78.53	3.85	63.14	18.91	19.55	13.46	8.97	69.87	0.32	66.35	69.87	20.19	0.64	13.14
Mhyang	95.23	86.9	74.08	56.08	43.59	97.45	88.78	20.75	93.02	64.61	31.77	96.51	66.69	99.26	54.67	73.67
Shaking	81.1	0.55	10.71	15.66	15.11	2.47	73.63	2.75	0.55	1.65	0.55	0.55	0.55	85.16	1.10	35.71
Singer1	87.71	19.43	22.57	24	20.86	50	21.71	26	24.86	27.43	53.14	30	20	41.71	11.43	27.71
Soccer	81.99	6.91	17.9	18.16	23.27	11.76	30.18	17.65	2.05	19.44	9.97	15.35	19.69	24.3	19.18	15.60
Sylvester	94	59.78	71.56	57.44	74.78	37.89	73.67	34.56	86.67	45.78	1.33	99.78	46.33	98.33	88.00	96.78
Trellis	72.31	24.87	40.21	28.4	17.28	92.24	26.28	40.04	11.29	38.98	12.87	44.97	37.39	46.21	35.45	28.04
Average SR	80.65	33.41	48.55	31.37	41.64	48.94	47.15	22.02	43.04	36.01	17.14	63.13	37.70	63.55	35.02	46.70



(a) Tracking results of the *Ironman* sequence



(b) Tracking results of the *Shaking* sequence



(c) Tracking results of the *Soccer* sequence

(d) Tracking results of the *Sylvester* sequence

Fig. 7. Screenshots of some sample tracking results when there are pose variations and severe illumination change.

with Haar-like feature (OAB, BSBT, MIL, CT and Struck), with a lower cost of feature representation.

3.4. Detail quantitative evaluation

For comprehensive quantitative evaluation, the general evaluation criteria including *center location error* (CLE) and *tracking success rate* (SR) are used in our experiments. On condition that the public video sequences have been manually labeled ground truth, let $\frac{area(V_T \cap V_G)}{area(V_T \cup V_G)}$ denote *the overlap ratio* (VOR) where V_T and V_G are the bounding boxes of the tracker and of the ground-truth. The tracking result of the current frame is considered as a success when the overlap ratio is greater than 0.5. Tables 2 and 3 show that our proposed tracker performs well in most of the video sequences and ranks first in the average CLEs and SR. The efficiency is also evaluated in the Table 3 using *frame per second* (FPS).

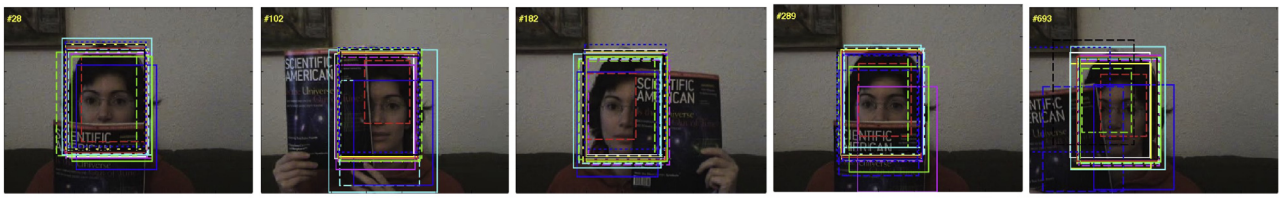
To evaluate the performance of our algorithm, we followed the same protocols in [12], where precision and success rate are measured by using densely sampled thresholds on center location error and bounding box overlap ratio, respectively. For SRE, each tracker is evaluated 12 times on each sequence, where more than 180,000 bounding box results are generated. As the success rate is defined at a specific threshold, it may not be fair or representative for tracker evaluation. Thus, we use the area under curve (AUC) of each success plot to rank our comparative trackers. The overall performance for all the trackers is summarized by the success and precision plots as shown in Fig. 5. Owing to the initialization errors, the average performance of SRE is lower than that of OPE. VTD, CPH, LSK, and Struck rank top 4 in the success plots of OPE and SRE. The success plots of CPH is higher than others when the overlap threshold is small, but less than VTD and LSK when the overlap threshold is large. The performance of our tracker in benchmark dataset is competitive with state-of-the-art online trackers, which indicates that our tracker is suitable to handle general challenges for tracking.

3.5. Qualitative evaluation

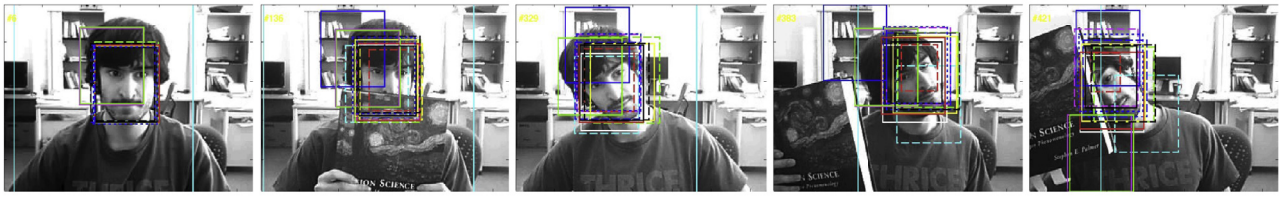
To evaluate the performance of our method for different challenging factors, we annotate the attributes of each sequence, and construct subsets with different dominant attributes including pose and illumination change, occlusion, background clutters, fast motion and motion blur, and in-plane and out-of-plane rotation. The comparison of these trackers proves the robustness of our proposed methods as follows.

3.5.1. Pose and illumination change

For the *Ironman* sequence shown in Fig. 7(a), the appearance changes quickly due to illumination and pose variation when the background changes from fireworks (♯18) to a ray of light (♯36) and the direction of the face from left (♯18), to front (♯30 and ♯36) and to right (♯48). Only the CPH algorithm performs well on this sequence, which indicates the effectiveness of CPH

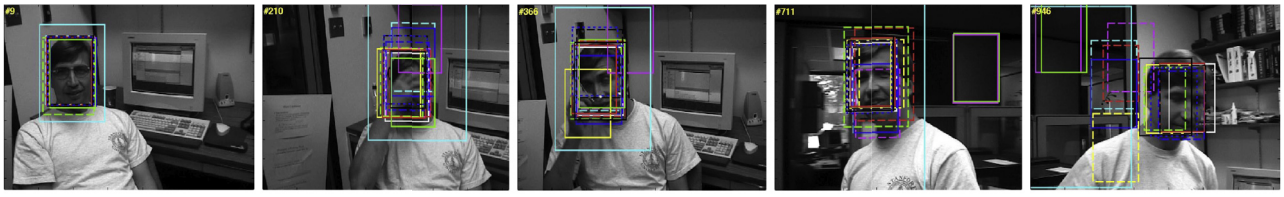


(a) Tracking results of the *FaceOcc1* sequence

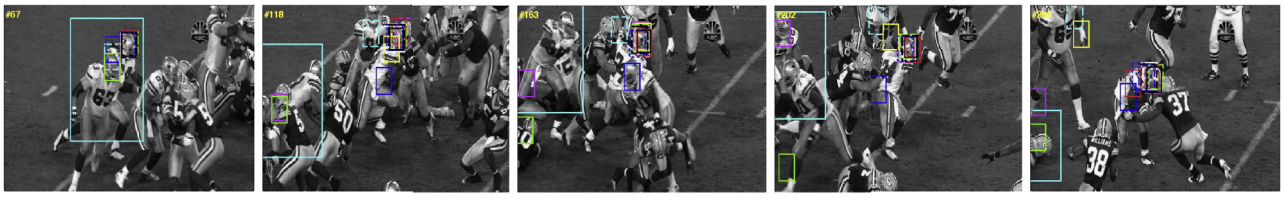


(b) Tracking results of the *FaceOcc2* sequence

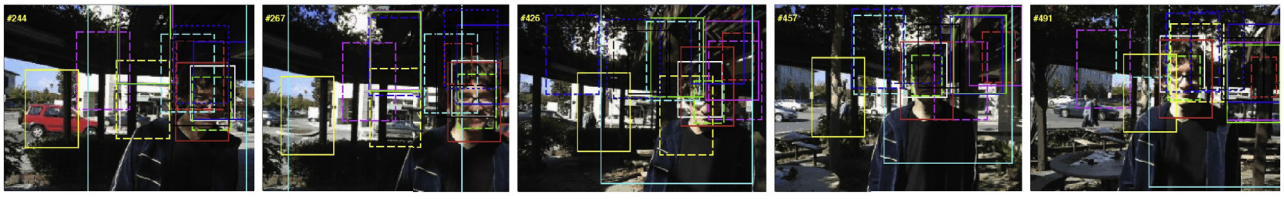
Fig. 8. Screenshots of some sample tracking results when there is severe occlusion.



(a) Tracking results of the *dudek* sequence



(b) Tracking results of the *Football* sequence



(c) Tracking results of the *Trellis* sequence

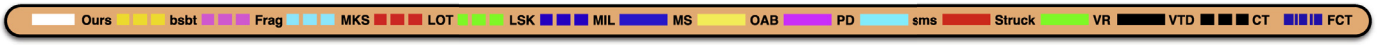
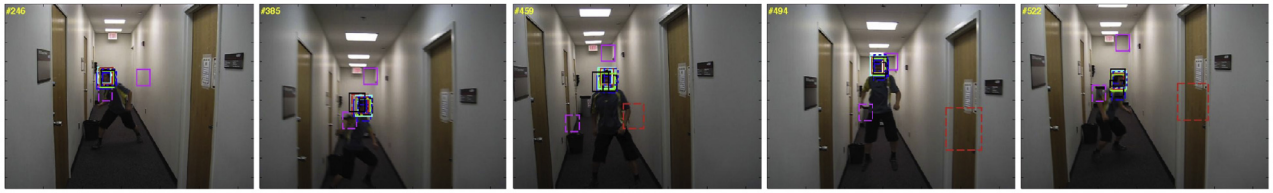


Fig. 9. Screenshots of some sample tracking results with background clutter.

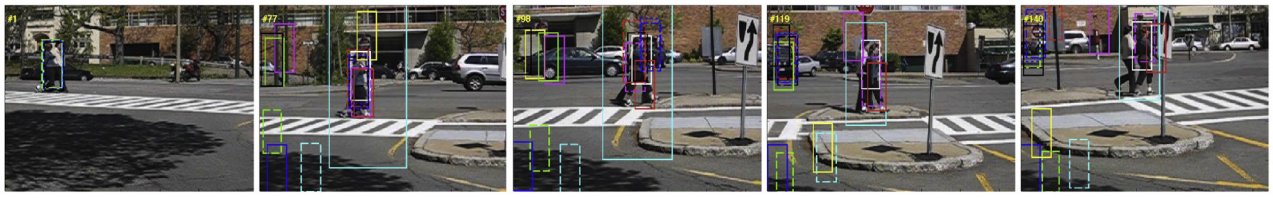
binary feature. Second-best performance came from VTD and CT, due to its multiple observation models and compressive features respectively, nevertheless, the two trackers are nearly lost from the target. In the *Shaking* sequence shown in Fig. 7(b), the object undergoes large illumination change because of background light. The VTD, LSK, CPH, and Struck perform well on this sequence with lower tracking errors than other methods. In Fig. 7(c), the target object in the *Soccer* sequence undergoes shape deformation (♯209, ♯270), occlusion(♯112) and chaotic background. The VR, CPH, and VTD methods have better performances than other methods. In the *Sylvester* sequence shown in Fig. 7(d), the object undergoes large pose and illumination change. The FCT, VTD, CT, MIL, Struck, and MKS methods perform well on this sequence. The SMS, MS, VR, LSK, and PD do not perform well on this sequence because the feature used in these methods are less effective for large scale pose variations. Due to the online learning foreground weight map, the proposed methods can cope with huge pose and illumination change well.

3.5.2. Occlusion

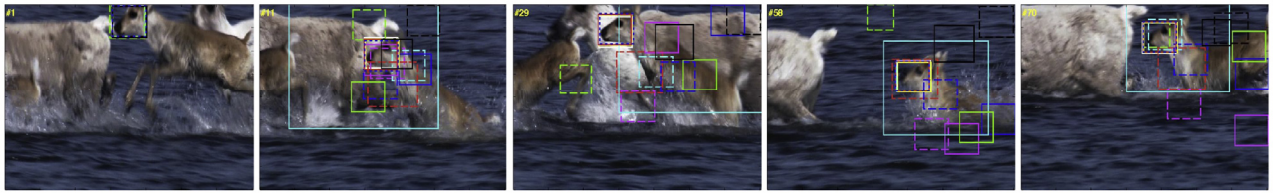
The target object in the *FaceOcc1* sequence in Fig. 8(a) undergoes part occlusion. The CPH, BSBT, Frag, SMS, VR, LOT, OAB, and MKS methods work well on this sequence. In the *FaceOcc2* sequence (Fig. 8(b)), the target undergoes pose variation and occlusion. The CPH, BSBT, Frag, LSK, OAB, PD, Struck, VTD, FCT, and MIL algorithms have good performance, while the MS, MKS,



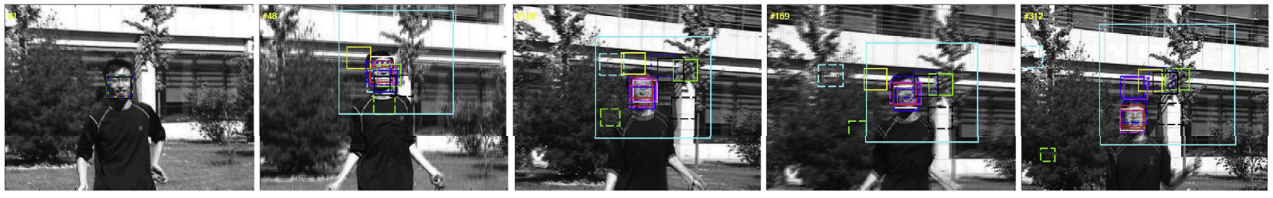
(a) Tracking results of the *boy* sequence



(b) Tracking results of the *couple* sequence



(c) Tracking results of the *deer* sequence



(d) Tracking results of the *Jumping* sequence

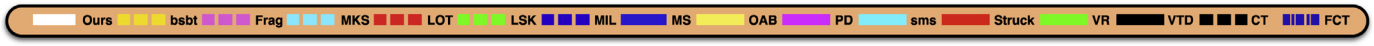


Fig. 10. Screenshots of some sample tracking results when there are fast motion and motion blur.

PD, VR, and SMS fail to track the target object stably. MKS lost the target since frame ♯6 and MS make errors when the occlusion occurs from the book (♯137). The OAB and MIL methods work well on this sequence as the most discriminative Haar-like features they used for object representation can handle pose variation and part occlusion effectively.

3.5.3. Background clutters

The surrounding background of the tracking object in the *dudek* sequence changes in illumination and context(Fig. 9(a)). Beyond that, the face undergoes pose change and occlusion (♯210). The Struck, FCT, CT, and CPH algorithms perform well on this sequence. The *Football* sequence (Fig. 9(b)) is a sport video. The object moves fast and the background undergoes significant and quick change. Only CPH, FCT, LSK, VTD, and CT work well on this sequence. FCT performs well in these sequences as it extracts discriminative scale invariant features for the most positive sample (i.e., the target object) online with classifier update for foreground and background separation. This is the benefit of compressive sensing based tracking, which is also reflected from the good performance of the proposed CPH algorithm. The background of the *Trellis* sequence (Fig. 9(c)) is clutter and the object undergoes illumination change continually. Only the Struck, LSK, and the proposed CPH algorithms perform well on this sequence. CPH has great performance in background cluster, which is mainly due to the effective online foreground learning method for promoting the confidence evaluation.

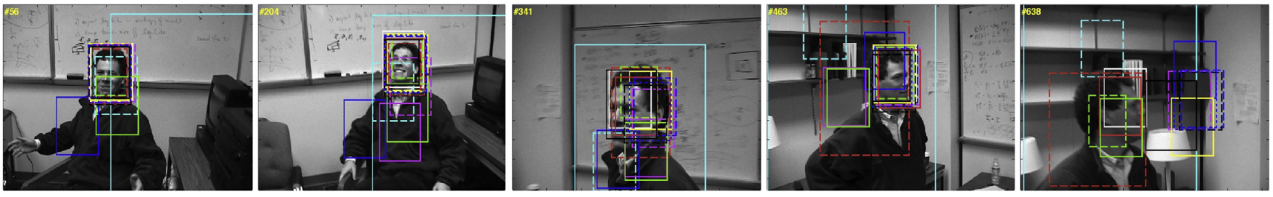
3.5.4. Fast motion and motion blur

The target object in the *boy* sequence (Fig. 10(a)) and *Jumping* sequence (Fig. 10(d)) both undergo continuous abrupt movements. Only FCT, Struck, and CPH perform well on the two sequences. PD, Frag, and LOT begin to drift post frame ♯246 of the *boy* sequence. Besides CPH, FCT, Struck, and Frag, all other trackers drift at frame ♯246. The object in the *couple* sequence Fig. 10(b) moves fast and undergoes an out-of-plane rotation. Only the CPH and Frag algorithms perform well on this sequence. The images of the *deer* sequence (Fig. 10(c)) are blurry due to fast motion of the deer. The Struck, FCT, OAB, and CPH algorithms work well in this sequence. When abrupt motion and motion blur occur in these sequences, most algorithms fail to track the target objects well, the proposed CPH algorithms outperform most of the other methods in most metrics (accuracy and success rate). Thanks to the proposed visual attention knowledge, the model drift problem can be handled effectively.

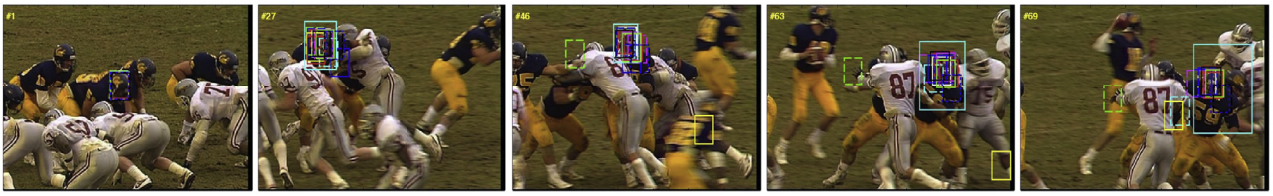
3.5.5. In-plane rotation and out-of-plane rotation

The target object in the *david2* sequence (Fig. 11(a)) and *FleetFace* sequence (Fig. 11(b)) both undergo the in-plane rotation and the out-plane rotation. The in-plane rotation in the *david2*

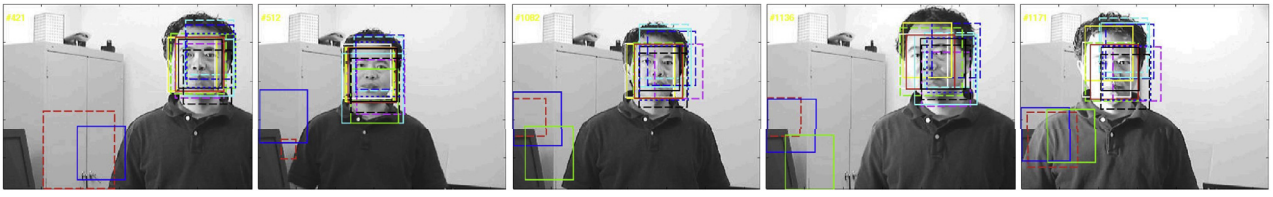
(a) Tracking results of the *david2* sequence



(b) Tracking results of the *FleetFace* sequence



(c) Tracking results of the *Football1* sequence



(d) Tracking results of the *Mhyang* sequence

Fig. 11. Screenshots of some sample tracking results when there is in-plane rotation and out-of-plane rotation.

sequence is big while the out-of-plane rotation in the *FleetFace* sequence is big. VTD, Struck, CPH, and FCT work well on the *david2* sequence, but VTD and FCT fail in the *FleetFace* sequence. The LSK performs well on the *FleetFace* sequence but fails on the *david2* sequence. The *Football1* sequence contains objects undergoing in-plane rotation, out-of-plane rotation, and abrupt motion. The VTD, LSK, Struck, FCT, and CPH algorithms perform better than others. Out-of-plane rotation occurs in *Mhyang* sequence; many algorithms perform well except LOT, MS, and VR.

3.5.6. The speed analysis

The running speeds of the proposed tracker and 15 comparative advanced trackers are present in Table 1. We can see that FCT and CT lead the speed list with the frame rate of 76.42 fps and 64.40 fps. However, the their performances of DP, OS, CLE are not competitive. While MIL and MS achieve higher frame rate than our tracker, our algorithm performs pretty adequately at 20.24 fps with highest DP and lowest CLE. As a discriminative model tracker, our method has a real-time frame rate thanks to the binary feature representation by perceptual hashing.

4. Conclusions

In this paper, we propose a fast robust tracking algorithm with an appearance model based on compressive hashing feature that preserves the structure of original image space but with small size. In addition, we introduce a visual attention knowledge that the object, namely foreground, should be always located in the center of the weight map, to handle the model drift problem. Experimental results show that the proposed tracking approach performs favorably when compared to a number of recent state-of-the-art algorithms. The online foreground learning for rectification reduces the ambiguous effects of the choice of the initial bounding box, and may be applied to many trackers in the future.

Acknowledgment

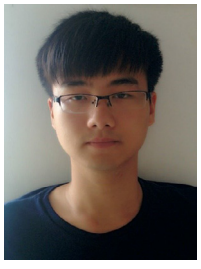
This research is supported by the National Natural Science Foundation of China (NSFC) under grant No. 41401525, the Natural Science Foundation of Guangdong Province under grant No. 2014A030313209, and CCF-Tencent Open Fund under Grant tIAGR20150114.

References

[1] C. Bao, Y. Wu, H. Ling, H. Ji, Real time robust l1 tracker using accelerated proximal gradient approach, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2012, pp. 1830–1837.
[2] J. Kwon, K.M. Lee, Tracking by sampling trackers, in: Proceedings of ICCV, IEEE, 2011, pp. 1195–1202.

[3] B. Liu, J. Huang, L. Yang, C. Kulikowsk, Robust tracking using local sparse appearance model and k-selection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2011, pp. 1313–1320.
[4] J. Kwon, K.M. Lee, Visual tracking decomposition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2010, pp. 1269–1276.
[5] L. Sevilla-Lara, E. Learned-Miller, Distribution fields for tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2012, pp. 1910–1917.
[6] B. Babenko, M.-H. Yang, S. Belongie, Robust object tracking with online multiple instance learning, IEEE Trans. Pattern Anal. Mach. Intell. 33 (8) (2011) 1619–1632.
[7] T.B. Dinh, N. Vo, G. Medioni, Context tracker: exploring supporters and distracters in unconstrained environments, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2011, pp. 1177–1184.
[8] S. Hare, A. Saffari, P.H. Torr, Struck: structured output tracking with kernels, in: Proceedings of ICCV, IEEE, 2011, pp. 263–270.
[9] Z. Kalal, K. Mikolajczyk, J. Matas, Tracking-learning-detection, IEEE Trans. Pattern Anal. Mach. Intell. 34 (7) (2012) 1409–1422.
[10] J.F. Henriques, R. Caseiro, P. Martins, J. Batista, Exploiting the circulant structure of tracking-by-detection with kernels, in: Proceedings of ECCV, Springer, 2012, pp. 702–715.
[11] K. Zhang, L. Zhang, M.-H. Yang, Real-time compressive tracking, in: Proceedings of ECCV, Springer, 2012, pp. 864–877.
[12] Y. Wu, J. Lim, M.-H. Yang, Online object tracking: a benchmark, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2013, pp. 2411–2418.
[13] S. He, Q. Yang, R.W. Lau, J. Wang, M.-H. Yang, Visual tracking via locality sensitive histograms, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2013, pp. 2427–2434.
[14] D. Wang, H. Lu, M.-H. Yang, Least soft-threshold squares tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2013, pp. 2371–2378.
[15] Q. Bai, Z. Wu, S. Sclaroff, M. Betke, C. Monnier, Randomized ensemble tracking, in: Proceedings of ICCV, IEEE, 2013, pp. 2040–2047.
[16] J. Zhang, S. Ma, S. Sclaroff, MEEM: robust tracking via multiple experts using entropy minimization, in: Proceedings of ECCV, Springer, 2014, pp. 188–203.
[17] S. Oron, A. Bar-Hille, S. Avidan, Extended Lucas–Kanade tracking, in: Proceedings of ECCV, Springer, 2014, pp. 142–156.
[18] J. Kwon, K.M. Lee, Tracking of a non-rigid object via patch-based dynamic appearance modeling and adaptive basin hopping monte carlo sampling, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009, pp. 1208–1215.
[19] L. Cehovin, M. Kristan, A. Leonardis, An adaptive coupled-layer visual model for robust visual tracking, in: Proceedings of ICCV, IEEE, 2011, pp. 1363–1370.
[20] A. Adam, E. Rivlin, I. Shimshoni, Robust fragments-based tracking using the integral histogram, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1, IEEE, 2006, pp. 798–805.
[21] D. Wang, H. Lu, M.-H. Yang, Online object tracking with sparse prototypes, IEEE Trans. Image Process. 22 (1) (2013) 314–325.
[22] X. Mei, H. Ling, Y. Wu, E.P. Blasch, L. Bai, Efficient minimum error bounded particle resampling l1 tracker with occlusion detection, IEEE Trans. Image Process. 22 (7) (2013) 2661–2675.
[23] M. Danelljan, F.S. Khan, M. Felsberg, J.v.d. Weijer, Adaptive color attributes for real-time visual tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2014, pp. 1090–1097.
[24] J. Wang, H. Wang, Y. Yan, Robust visual tracking by metric learning with weighted histogram representations, Neurocomputing 153 (2015) 77–88.
[25] J. Wang, S. Kumar, S.F. Chang, Semi-supervised hashing for scalable image retrieval, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 3424–3431.
[26] Y. Gong, S. Lazebnik, A. Gordo, F. Perronnin, Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval, IEEE Trans. Pattern Anal. Mach. Intell. 35 (12) (2012) 2916–2929.
[27] Y. Liu, F. Wu, Y. Yang, Y. Zhuang, A.G. Hauptmann, Spline regression hashing for fast image search., IEEE Trans. Image Process 21 (10) (2012) 4480–4491. A Publication of the IEEE Signal Processing Society
[28] Z. Liu, H. Li, W. Zhou, R. Zhao, Q. Tian, Contextual hashing for large-scale image search, IEEE Trans. Image Process. 23 (4) (2014) 1606–1614.
[29] M. Norouzi, A. Punjani, D.J. Fleet, Fast exact search in hamming space with multi-index hashing, IEEE Trans. Pattern Anal. Mach. Intell. 36 (6) (2013) 1.
[30] C. Ma, C. Liu, F. Peng, Two dimensional ensemble hashing for visual tracking, Neurocomputing 171 (2016) 1387–1400.
[31] M. Fei, J. Li, H. Liu, Visual tracking based on improved foreground detection and perceptual hashing, Neurocomputing 152 (2015) 413–428.
[32] E.J. Candès, T. Tao, Decoding by linear programming, IEEE Trans. Inf. Theory 34 (4) (2004) 435–443.
[33] K. Zhang, L. Zhang, M.H. Yang, Fast compressive tracking, IEEE Trans. Pattern Anal. Mach. Intell. 36 (10) (2014) 1.
[34] S. Ji, Y. Xue, L. Carin, Bayesian compressive sensing, IEEE Trans. Signal Process. 56 (6) (2008) 2346–2356, doi:10.1109/TSP.2007.914345.
[35] Z. Zivkovic, Improved adaptive Gaussian mixture model for background subtraction, in: Proceedings of the 17th International Conference on Pattern Recognition, 2, 2004, pp. 28–31Vol.2, doi:10.1109/ICPR.2004.1333992.
[36] Z. Li, J. Yang, L. Chen, Compressive perceptual hashing tracking with online foreground learning., in: Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO), 2015.
[37] V. Sulić, J. Pers, M. Kristan, S. Kovacic, Dimensionality reduction for distributed vision systems using random projection, in: Proceedings of the 20th International Conference on Pattern Recognition, 2010, pp. 380–383, doi:10.1109/ICPR.2010.101.
[38] E. Candes, T. Tao, Decoding by linear programming, IEEE Trans. Inf. Theory 51 (12) (2005) 4203–4215, doi:10.1109/TIT.2005.858979.
[39] E. Candes, T. Tao, Near-optimal signal recovery from random projections: universal encoding strategies? IEEE Trans. Inf. Theory 52 (12) (2006) 5406–5425, doi:10.1109/TIT.2006.885507.
[40] Y. Hua, B. Xiao, D. Feng, B. Yu, Bounded LSH for similarity search in peer-to-peer file systems, in: Proceedings of the 37th International Conference on Parallel Processing, 2008, pp. 644–651, doi:10.1109/ICPP.2008.25.
[41] B. Babenko, M.-H. Yang, S. Belongie, Visual tracking with online multiple instance learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009, pp. 983–990.
[42] S. Avidan, D. Levi, A. Bar-Hillel, S. Oron, Locally orderless tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 1940–1947.
[43] H. Grabner, M. Grabner, H. Bischof, Real-time tracking via on-line boosting., in: Proceedings of BMVC, 2006, pp. 47–56.
[44] D. Comaniciu, V. Ramesh, P. Meer, S. Member, S. Member, Kernel-based object tracking, in: Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence, 2003, pp. 564–577.
[45] R.T. Collins, Mean-shift blob tracking through scale space, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2, 2003, pp. 234–240.
[46] S. Stalder, H. Grabner, L. Van Gool, Beyond semi-supervised tracking: tracking should be as simple as detection, but not simpler than recognition, in: Proceedings of the 12th International Conference on Computer Vision Workshops, 2009, pp. 1409–1416.
[47] R.T. Collins, Y. Liu, M. Leordeanu, Online selection of discriminative tracking features., IEEE Trans. Pattern Anal. Mach. Intell. 27 (2005).

Long Chen received the B.Sc. degree in communication engineering and the Ph.D. degree in signal and information processing from Wuhan University, Wuhan, China, in 2007 and in 2013, respectively. From October 2010 to November 2012, he was co-trained Ph.D. Student at National University of Singapore. From 2008 to 2013, he was in charge of environmental perception system for autonomous vehicle SmartV-II with the Intelligent Vehicle Group, Wuhan University. He is currently an Associate Professor with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China. His areas of interest include computer vision, point cloud processing and unmanned autonomous vehicles.



Zheng Li received the B.Eng. degree in software engineering from School of Data and Computer Science, Sun Yat-sen University, China in 2016. He will pursue Ph.D. Degree from the Computer Science and Engineering Department, Hong Kong University of Science and Technology, since 2016. His research interests include visual tracking, transfer learning, dialog systems and reinforcement learning.

Jianfei Yang received the B.Eng. degree in software engineering from the school of Data and Computer Science, Sun Yat-sen University, China in 2016. He is now a Ph.D. student at the school of Electrical and Electronic Engineering, Nanyang Technology University. He interned in DJI as a vision engineer in July, 2014, during which he did research on machine vision as well as the visual navigation of Unmanned Aerial Vehicle (UAV) and participated in designing Chinese robot competition namely Robomaster. His research interests include visual tracking, object recognition, positioning of Unmanned Ground Vehicle (UGV) and control theory.