Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 628101, 18 pages
http://dx.doi.org/10.1155/2015/628101

Research Article

A Novel Object Tracking Algorithm Based on Compressed Sensing and Entropy of Information

Ding Ma,[1] Zhezhou Yu,[2] JiKun Yu,[2] and Wei Pang[3]

[1] *College of Software, Jilin University, Changchun 130012, China*
[2] *College of Computer Science and Technology, Jilin University, Changchun 130012, China*
[3] *School of Natural and Computing Sciences, University of Aberdeen, Aberdeen AB24 3UE, UK*

Correspondence should be addressed to Zhezhou Yu; yuzz@jlu.edu.cn

Received 17 December 2014; Revised 24 May 2015; Accepted 31 May 2015

Academic Editor: Andrzej Swierniak

Object tracking has always been a hot research topic in the field of computer vision; its purpose is to track objects with specific characteristics or representation and estimate the information of objects such as their locations, sizes, and rotation angles in the current frame. Object tracking in complex scenes will usually encounter various sorts of challenges, such as location change, dimension change, illumination change, perception change, and occlusion. This paper proposed a novel object tracking algorithm based on compressed sensing and information entropy to address these challenges. First, objects are characterized by the Haar (Haar-like) and ORB features. Second, the dimensions of computation space of the Haar and ORB features are effectively reduced through compressed sensing. Then the above-mentioned features are fused based on information entropy. Finally, in the particle filter framework, an object location was obtained by selecting candidate object locations in the current frame from the local context neighboring the optimal locations in the last frame. Our extensive experimental results demonstrated that this method was able to effectively address the challenges of perception change, illumination change, and large area occlusion, which made it achieve better performance than existing approaches such as MIL and CT.

1. Introduction

Recognition and tracking of moving objects through computer vision technology have been widely applied in various fields such as food quality control [1], traffic flow monitoring [2], and illegal surveillance [3]. At present, the methods of detecting moving object mainly include model-based tracking [4], region-based tracking [5], and contour-based tracking [6]. Representative object tracking algorithms have been proposed in recent years by a series of studies [7–13]. Although object tracking algorithms have been studied for decades, many challenging problems are still to be solved. Many factors affect the performance of object tracking algorithms, including illumination change, occlusion change, and complex background. However, so far, to the best of our knowledge, no robust algorithm has been developed to effectively address all the challenges caused by the aforementioned factors. Therefore, the proposed tracking algorithm in this research attempts to partially solve the problems caused by these influencing factors.

Characterization of an object is an extremely important component for any type of object tracking algorithms. The overall object template is widely used in tracking [14–16]. Mei et al. [17] utilized sparse representation to overcome the object appearance changes. Besides template, many features have been used in tracking algorithms, including color histogram [8], histograms of oriented gradients (HOG) [18], Region Covariance descriptor [19–21], Haar-like features [22, 23], and ORB [24]. In addition, search strategies are also critical for tracking algorithms, and examples of search strategies include both definite and random methods [25]. Furthermore, due to high efficiency of calculation, particle filter is widely applied in object tracking algorithms [7, 26].

In this paper we propose a novel object tracking algorithm based on compressed sensing and information entropy. The rest of this paper is organized as follows: particle filter, ORB feature, Haar feature, sparsification, local area, and entropy of information are introduced in Section 2. The detailed description of the new object tracking algorithm is presented in Section 3. This is followed by the report of experimental results to validate the proposed algorithm in Section 4. Finally, Section 5 concludes the paper.

2. Related Work

2.1. Particle Filter. Particle filter [27] simulates the state space by using a certain number of particles. Each particle is given a weight through approximating probability density function, thereby resulting in the minimum variance estimation of system state.

There commonly exists the degeneracy problem in particle filter; that is, after several states all but one particle has significant weight [28]. A number of approaches have been used to avoid particle degeneracy [29]. One solution [30] is to generate several offspring for each particle, and the number of offspring of each particle is proportional to its weight. Thus those particles with higher weights will be more likely to be chosen, which avoids the degeneracy to a great extent.

When applying particle filter, the state transition model and observation model are defined first. Suppose t is the frame index of a video. $v_{t-1}^{(i)}$ is the velocity of the ith particle in frame t, $x_t^{(i)}$ is the position of the ith particle in frame t, $z_t^{(i)}$ is the observation of the ith particle in frame t, $w_t^{(i)}$ is the weight of the ith particle in frame t, and the problems are solved according to the process described below.

Step 1. When time $t = 1$, effective sampling of prior knowledge $p(x_1)$ is taken. Thus the sampling particle set $\{x_1^{(i)}\}_{i=1}^N$ is generated, where N is the number of particles.

Step 2. From $t = 2$, new particle set is generated by using the state transition function $x_t^{(i)} = f(x_{t-1}^{(i)}, v_{t-1}^{(i)})$, $f(x_{t-1}^{(i)}, v_{t-1}^{(i)}) = x_{t-1}^{(i)} + v_{t-1}^{(i)} + \text{rand}$, where rand is a random number matrix. The weight of particle is calculated and normalized by using the observation model $w_t^{(i)} = p(z_t^{(i)} \mid x_t^{(i)})$.

Step 3. The system output is obtained by calculating weighted average of particles' position. Then the system state at time t is estimated by $\hat{x}_t = \sum_{i=1}^N x_t^{(i)} w_t^{(i)}$, which is considered as the solution of the problem at time t.

Step 4. The original particle set $\{x_t^{(i)}, w_t^{(i)}\}_{i=1}^N$ is resampled to get new particle set.

Step 5. $t = t + 1$, return to Step 2 and continuously running until $t = T$.

As a rule of thumb, the particle number in this study is set as 200. The state transition function is set to be translation transformation in affine transformation. The prior probability model is assumed to be normal distribution. The mean and variance are obtained by using the vectors of ORB and Haar features.

2.2. Extraction of ORB Feature. The ORB feature [24] extracted for each sample image in the previous step is taken as the input characteristic of our tracking algorithm. The ORB feature is extracted according to the following equation:

$$\begin{aligned} &\text{feature} \\ &\quad = \text{ORB}\left(\text{image}, \text{scale\_factor}, n\text{levels}, \text{edge\_threhod}\right), \end{aligned} \tag{1}$$

where image stands for the image data, scale_factor stands for scale factor, *n*levels is the scale layers of the pyramid, and edge_threshold is the threshold of marginalization. In the implementation, the ORB class of OPENCV 2.3.1 (http://opencv.org/downloads.html) is called to extract the ORB feature. The parameter scale_factor is set to 1.2, *n*levels takes 1, and edge_threshold takes 31. After extracting the ORB feature from each image, the classifier is ready for training and application.

2.3. Extraction of the Haar Feature. The characteristic value of Haar-like feature equals the difference between the sum of pixel value in black filled area and that in white area. When extracting the Haar feature, only the following parameters are required: image (the image used to extract Haar feature), window size (the area of the search window), and rectangle number (Haar feature's model). The equation to extract the Haar features is given as follows:

$$\begin{aligned} &\text{feature} = \text{Haar}\left(\text{image}, \text{width}, \text{height}, \text{minNumRect},\right. \\ &\qquad \left.\text{maxNumRect}, M\right), \end{aligned} \tag{2}$$

where image is the image from which the Haar feature will be extracted, width and height are the width and height of the search window, respectively, minNumRect and maxNumRect are the minimum and maximum numbers of feature rectangles, respectively, and M is the number of times for extracting Haar features (as a rule of thumb, M is set as 50), namely, the dimension of vector.

In the current study, width and height are the width and height of the search area, respectively. minNumRect and maxNumRect are 2 and 4, respectively.

2.4. Sparsification. In this study, the ORB and Haar features are sparsified by using the following sparsification transition equation:

$$V = R * X, \tag{3}$$

where X stands for the vector which needs to be sparsified, V is the vector after sparsification, and R is the sparse matrix. Two methods are available for generating each element r_{ij} of R, which are shown in (4) and (5), respectively.

The Haar feature is processed by using the method given in [31], where sparsifying the Haar feature is through using (4) [31]. In (4), Achlioptas [32] proved that this type of matrix with $s = 2$ or 3 satisfied the Johnson-Lindenstrauss lemma,

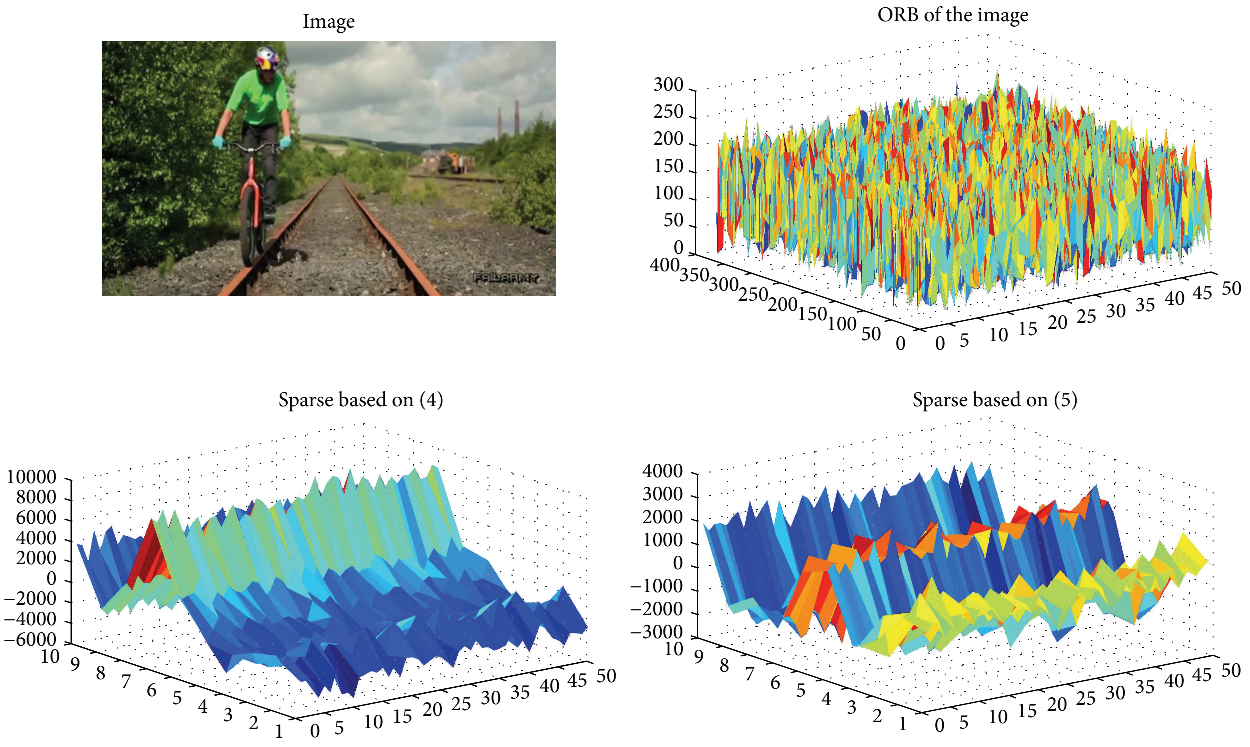


FIGURE 1: A comparison of results from (4) and (5) (image is taken from the Biker Video sequences [33]).

as shown in the following, where $s = M/4$ [31] and M is the column number of R:

$$r_{ij} = \sqrt{s} \times \begin{cases} 1, & \text{according to probability } \dfrac{1}{2s} \\ 0, & \text{according to probability } 1 - \dfrac{1}{s} \\ -1, & \text{accordingt o probability } \dfrac{1}{2s}. \end{cases} \quad (4)$$

Different from (4), the generated random matrix according to (5) has a higher sparsity. The ORB feature is sparsified by using (5), where k takes 3. When $k = 3$, it is very sparse where two-thirds of the computation can be avoided:

$$r_{ij} = \begin{cases} 1, & \text{according to probability } \dfrac{1}{2k} \\ 0, & \text{according to probability } 1 - \dfrac{1}{k} \\ -1, & \text{according to probability } \dfrac{1}{2k}. \end{cases} \quad (5)$$

Take Figure 1 as an example: *Image* is the experimental figure; ORB of the Image is the extraction of ORB feature dimensions on the *Image*. The dimension of the ORB feature is high, and the nonlinear variation is complex and thus is not conducive for the processing of classifier. The sparse result of the figure, "sparse based on (4)" is based on (4), and the dimension is 40 times reduced. At the same time, the difference features extracted from the image are retained, which greatly reduces the processing burden of the classifier. Consider the 3D map shown in Figure 1; we can see that, compared to the sparse result calculated from (5) (sparse based on (5)), results obtained from (4) have larger variance; after normalization the part dimensions of information accounted for the total information are larger and in aspect of describing the features of diversity is not as good as the results from (5). In comparison, the sparsification method based on (5) can achieve dimension reduction and at the same time reduce as much as possible the loss of difference between features. So we choose (5) in this research.

2.5. Local Context. The movement of an object being tracked is a continuous process from the previous frame to the next one. Thus, the object displacement difference between adjacent frames is necessary to be within a limited range. Take Figure 2 as an example: in the current frame, the object region is enclosed by a red rectangle. Then, in the next frame, the object region will naturally not exceed the limit of the blue rectangle, which is defined as local context in [34]. When generating particles, the coordinates of particles need to be placed within a limit. For those particles beyond the local context, we transfer the particles to the restricted region. Reasonably adjusting the size of the local context according to the motion speed of an object can increase the accuracy of the object location detection to a certain extent. In this study, the width and length of local context are twice as those of the original object region.

The red rectangle stands for object location in the previous frame; the green rectangles are those generated particles; the blue rectangle is the restricted local context.

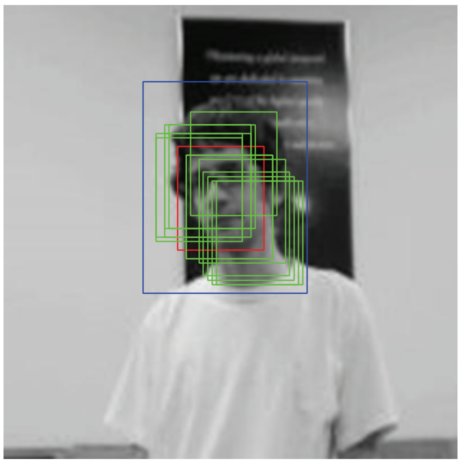


Figure 2: The illustration of local context (source taken from the David2 video sequences [33]).

2.6. Self-Adaptive Fusion Based on Entropy of Information. The concept of information entropy [35] is used to quantitatively measure the amount of information contained by data. The entropy of information is defined in the following:

$$H = -K\sum_{i=1}^{n} p_i \log p_i, \quad (6a)$$

where K is a positive constant. If $K = 1$, we get

$$H(X) = -\sum_{i=1}^{n} p(x^i) \log p(x^i), \quad (6b)$$

where $H(X)$ is the entropy of information for the feature (Haar or ORB feature), $p(x^i)$ is the probability of x^i, and x^i is the ith particle.

In feature fusion, the weight of a feature can be determined by the amount of information contained by data, namely, the entropy of information. After discretization and taking logarithm, (6a) and (6b) are transformed to the following equation:

$$H(x) = -\sum_{j=1}^{N} \left(p(x^j) \log_2^{p(x^j)} \right). \quad (7)$$

Then the observation probability value of Haar feature, namely, the weight of Haar feature, is expressed in

$$H(z_{\text{haar}}) = -\sum_{j=1}^{N} \left(p(z_{\text{haar}} \mid x^j) \log_2^{p(z_{\text{haar}} \mid x^j)} \right), \quad (8)$$

where $H(z_{\text{haar}})$ is the entropy of information for the Haar feature; $p(z_{\text{haar}} \mid x^j)$ is the conditional probability of the jth particle with observation value z_{haar}; N is the particle number.

The observation density of ORB feature, namely, the weight of ORB feature, is shown in

$$H(z_{\text{orb}}) = -\sum_{j=1}^{N} \left(p(z_{\text{orb}}^j \mid x^j) \log_2^{p(z_{\text{orb}}^j \mid x^j)} \right), \quad (9)$$

where $H(z_{\text{orb}})$ is the information entropy of ORB feature; $p(z_{\text{orb}} \mid x^j)$ is the probability of the jth particle with observation value z_{orb}; N is the particle number.

According to the definition of entropy of information, the fused weight of Haar feature is computed by the following equation:

$$\beta_{\text{haar}} = \frac{H(z_{\text{haar}})}{H(z_{\text{haar}}) + H(z_{\text{orb}})}. \quad (10)$$

The fused weight of ORB feature is obtained by

$$\beta_{\text{orb}} = \frac{H(z_{\text{orb}})}{H(z_{\text{haar}}) + H(z_{\text{orb}})}. \quad (11)$$

The illustration of fused weight of ORB feature and Haar feature is shown in Figure 3. In Figure 3, the number of different color squares represents different fused weights. Among them, the small red squares represent the Haar feature, while the small green squares stand for the ORB feature. As illustrated, fused weight of Haar feature accounts for a bigger proportion when the object is not being covered. However, when the object is covered, ORB feature dominates in the fusion.

In multiple feature fusion, the additive fusion and multiplicative fusion are the most widely used methods [36]. Additive fusion can relatively reduce system noise, while multiplicative fusion can increase weight identification ability but amplify the system noise. In the current study, two types of fusion are combined through adaptive adjustment of weight allocation by using

$$p(z_i \mid x) = \frac{1}{(1+\beta_{\text{haar}})(1+\beta_{\text{orb}})} [p(z_{\text{haar}} \mid x) p(z_{\text{orb}} \mid x) + \beta_{\text{haar}} p(z_{\text{haar}} \mid x) + \beta_{\text{orb}} p(z_{\text{orb}} \mid x) + \beta_{\text{haar}}\beta_{\text{orb}}], \quad (12)$$

where β_{haar} and β_{orb} are weights of the two features, respectively, and they are also called adaptive adjustment factors determined by the amount of entropy of information of the two features. According to (12), when β_{haar} approaches 0, namely, the entropy of information of Haar feature reaches minimum, (12) is equivalent to additive fusion, where only ORB feature plays a role in tracking. However, when β_{orb} approaches 0, namely, the entropy of information of ORB feature reaches minimum, (12) is equivalent to multiplicative fusion, where only Haar feature plays a role. Finally, $p(z_i \mid x)$ will be the particle weight w.

3. Algorithm Description (HOPEF: Haar-ORB-Particle-Entropy-Information)

See Figures 4 and 5

The description of our algorithm is shown as follows.

The Initial Condition. In the first frame $f(1)$, the location of tracking object is provided.

Initial Operation. The observation of the first frame is calculated as the estimated observation of the second frame.

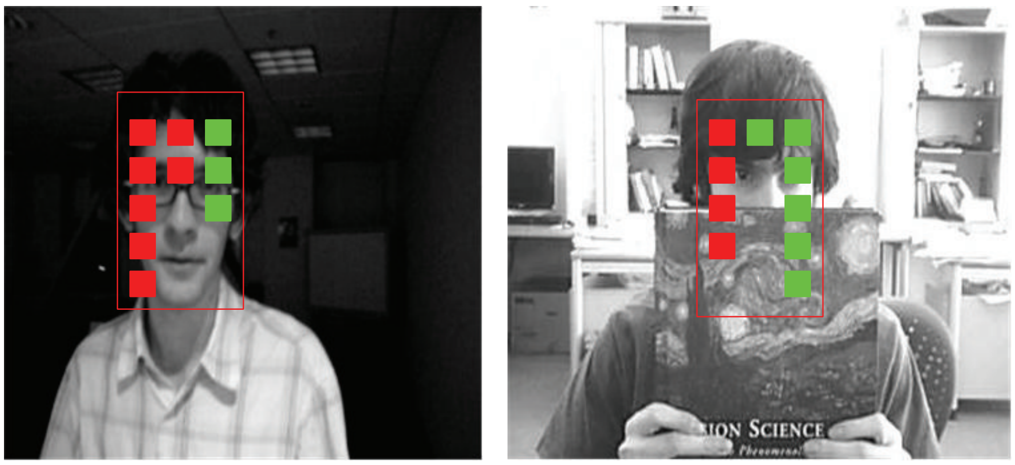


FIGURE 3: Illustration of fused weight of the ORB feature and Haar feature (video source: David video sequences and FaceOcc2 video sequences [33]).

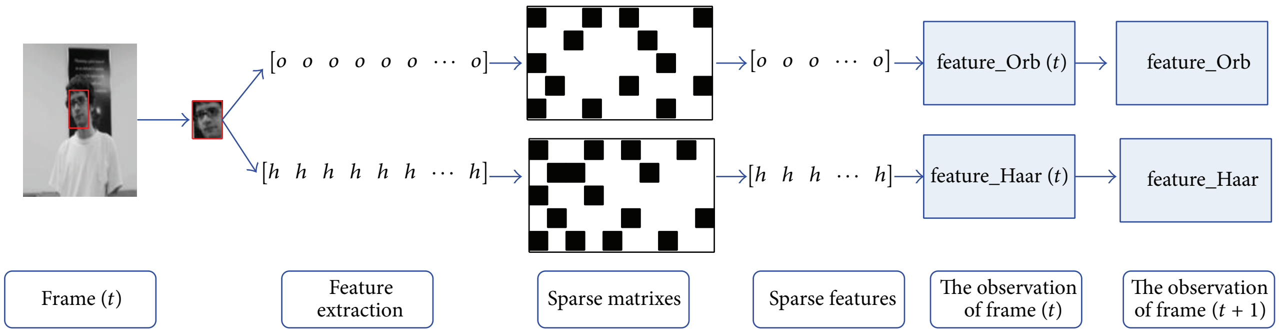


FIGURE 4: The process of updating observation estimation, frame = t.

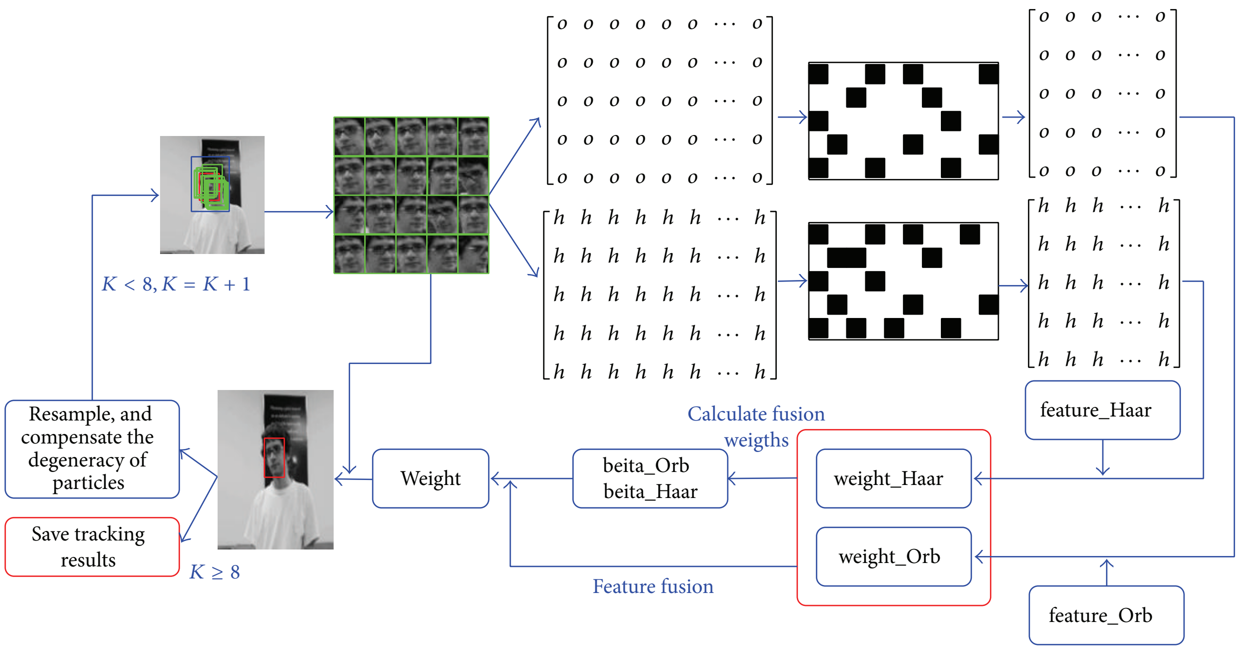


FIGURE 5: The process of estimating object results, frame = $t + 1$.

Tracking starts from the second frame ($t = 2, 3, \ldots, N$).

Input. Frame $f(t)$.

Step 1. In $f(t)$, particles are sampled. The vector of the Haar feature and ORB feature are extracted from each particle image to form the observation matrix. The prior probability of each particle is computed as weight according to normal distribution.

Step 2. The entropy of information of particle weight H_h and H_o is calculated.

Step 3. The fused weights *beita_h* and *beita_o* are computed.

Step 4. The location of the object is obtained (selecting candidate object locations in the current frames from the local areas neighbouring the optimal locations in the last frames) by calculating weighted average fusion of all particle locations.

Step 5. Resampling is conducted. The particle degeneracy is compensated as described in Section 2.1.

Step 6. Steps 1~5 are repeated for 8 times (based on the rule of thumb). The kth estimated object location is saved as the object location of the tth frame.

Step 7. Loc(t) is the estimated observation of $(t + 1)$th frame. The update rate is 0.2.

Step 8. If $t < N$, $t = t + 1$, return to Step 1 and continue running. If $t = N$, the tracking is finished. The tracking results and location of each frame are returned.

The corresponding algorithm flow chart is shown in Figure 6.

4. Experiments

To verify the performance of the algorithm, an OpenCV and MATLAB-based object tracking prototype framework have been developed according to the proposed algorithm. In this prototype framework, the ORB feature is extracted from the OpenCV library, while the Haar feature and adaptive fusion of entropy of information are implemented by MATLAB scripts. In this study, open video datasets are utilized as experimental data including twelve video sequences: Basketball, David2, dollar, Dudek, OccFace2 Freeman1, Mhyang, Sylvester, Gym, Jumping, Jogging, Trellis from [33], and EnterPaths1 from [37]. The algorithm is compared with other algorithms such as BSBT [38], SBT [39], CT [31], MIL [10], and BT [23] in terms of performance from the following three aspects: average pixel distance error, average overlap area, and success rate (the overlap of object tracking results with rectangle area > 50%). All experiments reported in this research were performed on a computer with an Intel i5 CPU (2.67 GHz basic frequency) and 4 GB memory. The parameters have been given in Sections 2 and 3.

Table 1: A comparison of the six algorithms on average pixel distance error.

	BSBT	SBT	CT	MIL	BT	HOPEF
Basketball	303.01	278.57	121.56	103.80	**71.56**	*16.82*
David2	101.85	142.60	15.67	11.42	*3.51*	**5.88**
Dollar	63.86	115.76	**17.91**	73.67	29.00	*15.02*
Dudek	78.52	204.26	31.93	32.88	**22.59**	*18.21*
OccFace2	110.14	38.47	**20.15**	23.04	*17.51*	22.27
Freeman1	185.76	157.49	**12.02**	19.31	119.11	*11.44*
Mhyang	63.80	63.66	22.65	36.10	**10.89**	*10.50*
Sylverter	55.31	104.01	*21.58*	46.05	52.93	**35.17**
Gym	102.38	55.23	31.05	66.28	**21.04**	*16.78*
Trellis	117.48	49.21	**17.39**	78.26	105.23	*12.32*
Jumping	185.17	115.68	47.16	41.75	**26.27**	*9.77*
EnterPaths1	23.49	21.14	114.99	169.69	*2.98*	**6.72**
Jogging	45.74	95.93	91.61	98.61	*11.66*	**14.47**

Table 2: A comparison of the six algorithms on average overlapping area.

	BSBT	SBT	CT	MIL	BT	HOPEF
Basketball	0.103	0.04	0.26	0.28	**0.40**	*0.70*
David2	0.50	0.21	0.46	0.60	*0.87*	**0.79**
Dollar	0.40	0.16	**0.78**	0.30	0.68	*0.82*
Dudek	0.62	0.41	0.74	0.75	*0.78*	*0.82*
OccFace2	0.40	0.65	**0.80**	0.76	*0.81*	0.76
Freeman1	0.11	0.18	*0.42*	0.29	0.25	*0.40*
Mhyang	0.56	0.55	0.59	0.44	**0.75**	*0.77*
Sylverter	0.63	0.40	*0.66*	0.39	**0.59**	0.50
Gym	0.49	0.67	0.67	0.35	**0.78**	*0.81*
Trellis	0.22	0.41	*0.90*	0.18	0.16	**0.89**
Jumping	0.13	0.09	0.08	0.07	**0.39**	*0.64*
EnterPaths1	**0.84**	0.33	0.26	0.004	*0.94*	0.82
Jogging	**0.61**	0.46	0.21	0.02	*0.73*	0.59

4.1. The General Tracking Results. The proposed algorithm, which we name HOPEF (Haar-ORB-Particle-Entropy- Information), is compared with other algorithms in analyzing ten videos sequences from three aspects: average pixel distance error, average overlap area, and success rate.

The comparison results are shown in Tables 1–3.

The data in Table 1 represent the average error, which is the average of distances between the tracking result and the actual center of the tracking object in all images. The data in Table 2 are evaluated in terms of average overlapping area, namely, the ratio of the overlapping area between the object tracking rectangle and the groundtruth_rect rectangle, in which each row represents the bounding window of the actual object in that frame, that is, (x, y, window-width, and window-height), where (x, y) is the left upper corner coordinates of the window. The data in Table 3 are average success rates. The success of tracking in one image is defined as the ratio of the overlapping area exceeding 0.5. The success rate is defined as the proportion of successfully tracked image

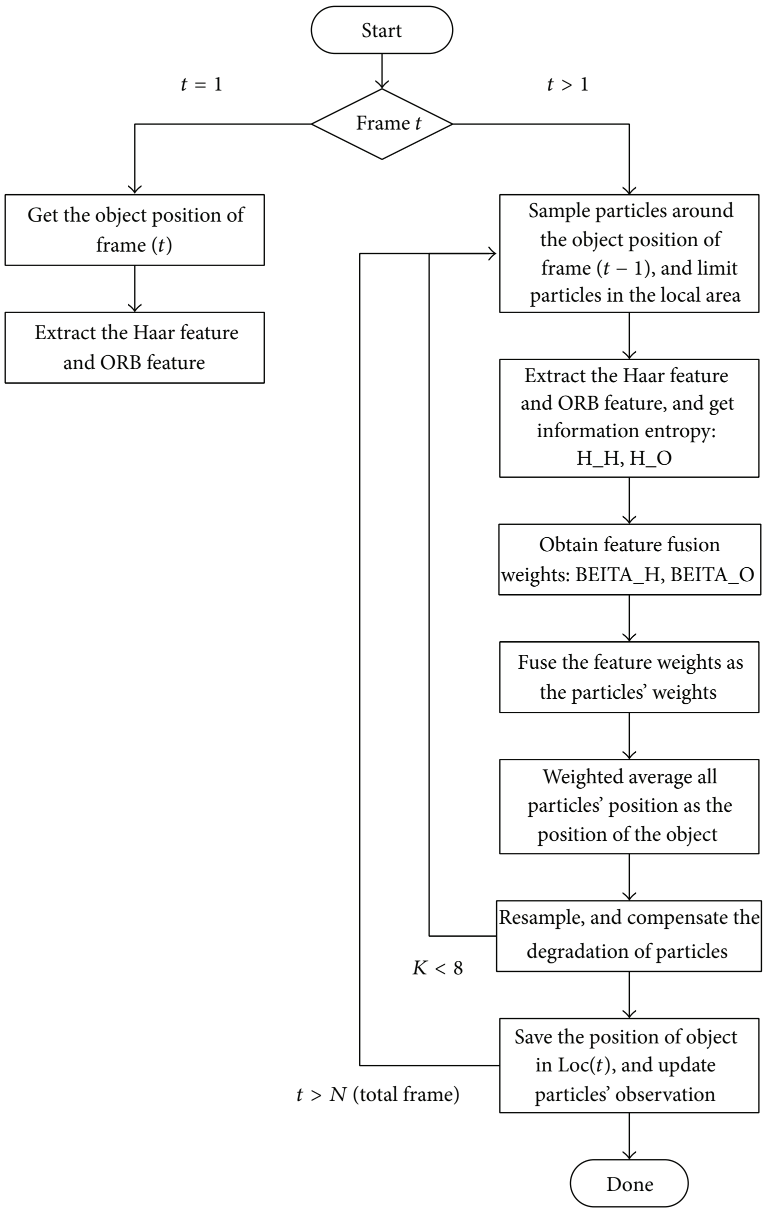

Start
t = 1
t > 1
Frame t
Get the object position of frame (t)
Extract the Haar feature and ORB feature
Sample particles around the object position of frame (t − 1), and limit particles in the local area
Extract the Haar feature and ORB feature, and get information entropy: H_H, H_O
Obtain feature fusion weights: BEITA_H, BEITA_O
Fuse the feature weights as the particles' weights
Weighted average all particles' position as the position of the object
Resample, and compensate the degradation of particles
K < 8
Save the position of object in Loc(t), and update particles' observation
t > N (total frame)
Done

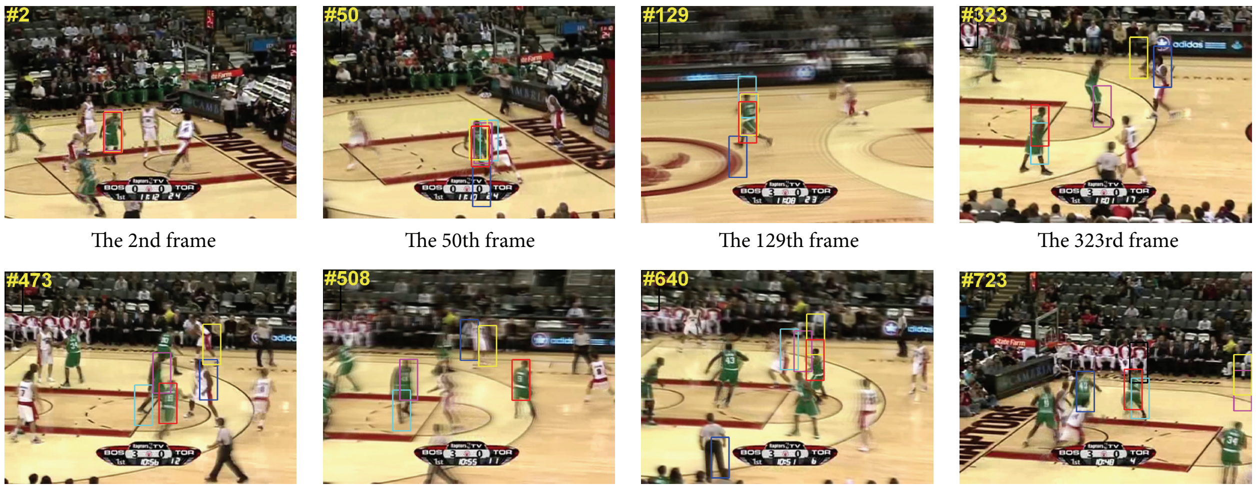
Figure 6: The flow chart of Haar-ORB-Entropy of Information algorithm (HOPEF).

among the total images. In Table 3, red color stands for optimal results, while blue stands for the suboptimal results.

From a further examination of all the tracking results in Tables 1~3 for all ten videos, we can see that our algorithm shows the minimum average error in 8 videos, the maximum average overlapping area in 7 videos, and the highest success rate in 9 videos, respectively. In general, our proposed HOPEF is more accurate in tracking the location of the object than other algorithms such as BSBT, SBT, BT, CT, and MIL.

4.2. The Analysis of Algorithm Performance on Key Frames. In this section, we analyze the performance of the six algorithms based on some key frames. First, the algorithms and their corresponding colors used in this section are listed in Table 4.

4.2.1. Basketball Video. Basketball video sequences describe an NBA basketball game where the tracking object is a basketball player. The foreground of the video contains the players from two teams, while the background is the audience in the stands. Besides the complex background, another tracking difficulty in this video is that the uniforms of the players. The performance of the six algorithms is shown in Figure 7. BSBT (black) and SBT (blue) lost the object in the 180th frame. MIL (purple) mistakenly tracked the wrong player who wore the same uniform from the same team. CT (yellow) lost the object in the 508th frame; meanwhile, BT (green) made the same mistake. However, our proposed algorithm was able to basically cover the object across all frames. The error curve of HOPEF (red) is found to lie in a



The 2nd frame The 50th frame The 129th frame The 323rd frame

The 473rd frame The 508th frame The 640th frame The 723rd frame

FIGURE 7: A performance comparison of the six algorithms in the 2nd, 50th, 129th, 323rd, 473rd, 508th, 640th, and 723rd frames in Basketball video sequence [33].

TABLE 3: A comparison of six algorithms on success rates.

	BSBT	SBT	CT	MIL	BT	HOPEF
Basketball	0.10	0.05	0.299	0.310	**0.417**	*0.903*
David2	0.529	0.233	0.227	**0.959**	0.935	*1*
Dollar	0.394	0.152	*1*	0.393	*1*	*1*
Dudek	0.788	0.499	**0.979**	0.951	0.924	*0.988*
OccFace2	0.417	0.674	*1*	**0.922**	0.899	0.913
Freeman1	0.129	0.187	*0.215*	0.018	**0.212**	**0.212**
Mhyang	0.657	0.646	0.782	0.314	**0.807**	*0.960*
Sylverter	**0.707**	0.439	*0.792*	0.473	0.687	0.523
Gym	0.62	0.84	0.87	0.27	**0.97**	*0.99*
Trellis	0.20	0.41	*1*	0.88	0.16	*1*
Jumping	0.18	0.1	0.07	0.02	**0.45**	*0.85*
EnterPaths1	**0.89**	0.29	0.29	0.005	*1*	*1*
Jogging	**0.78**	0.56	0.23	0.007	*0.95*	0.68

TABLE 4: The algorithms and corresponding colors.

Algorithms	SBT	BSBT	BT	MIL	CT	HOPEF
Color of rectangle	Blue	Black	Green	Purple	Yellow	Red

small area during all the time when referencing error chart frame by frame.

4.2.2. David2 Video Sequences. The background in David2 video sequence is a laboratory bench and walls, which are all fixed. The tracking difficulty in this video is that the dark plate painting on the wall has similar color with that of the object, which produces interference. The performance of the six algorithms is shown in Figure 8. In the 61st frame, influenced by the background dark painting plate, a large tracking deviation appeared and this error cumulated and affected the performance of CT (yellow) and MIL (purple) following this frame. During the entire tracking process, SBT (blue) and BSBT (black) lost object frequently. However, BT (green) and HOPEF (red) show a relatively smaller deviation in each frame, thereby avoiding losing the object.

4.2.3. Dollar Video Sequences. The tracking objects in the dollar video sequence are a pile of dollars on the surface of a table. The interference is all set on the foreground including 2 aspects. On one hand, one dollar bill was folded up. On the other hand, one pile of bills was divided into two piles, namely, splitting and merging of two stacks of money repeated during the entire video. The performance of the six algorithms is shown in Figure 8. In general, only BSBT algorithm (black) is not affected by the action of the dollar bill being folded up. However, the interleaving movement of the two piles of money made BSBT (black) mistakenly follows the wrong pile, thereby leading to the subsequent false tracking. However, although HOPEF (red) is influenced by the bill folding up action, it is not affected by alternating interference. In addition, HOPEF (red) shows the minimum deviation error among all the algorithms, which means that it is less affected by alternating inference. The frames in Figure 9 demonstrated that our HOPPEF (red) algorithm achieved the highest accuracy.

4.2.4. Dudek Video Sequences. The object in Dudek video sequence is the head of a moving person in the laboratory. In the process of shooting, the person took off his glasses and moved along the laboratory. In addition, both background and illumination changed. The performance of the six algorithms is shown in Figure 10. SBT (blue) lost the object from time to time. In the 250th frame, CT (yellow) and MIL (purple) show an upper right deviation and BSBT (black) and BT (green) show a lower left deviation, which results in the loose of the tracking object in the subsequent frames for all these four algorithms. Only our HOPEF (red) algorithm shows a relatively small deviation from these frames.

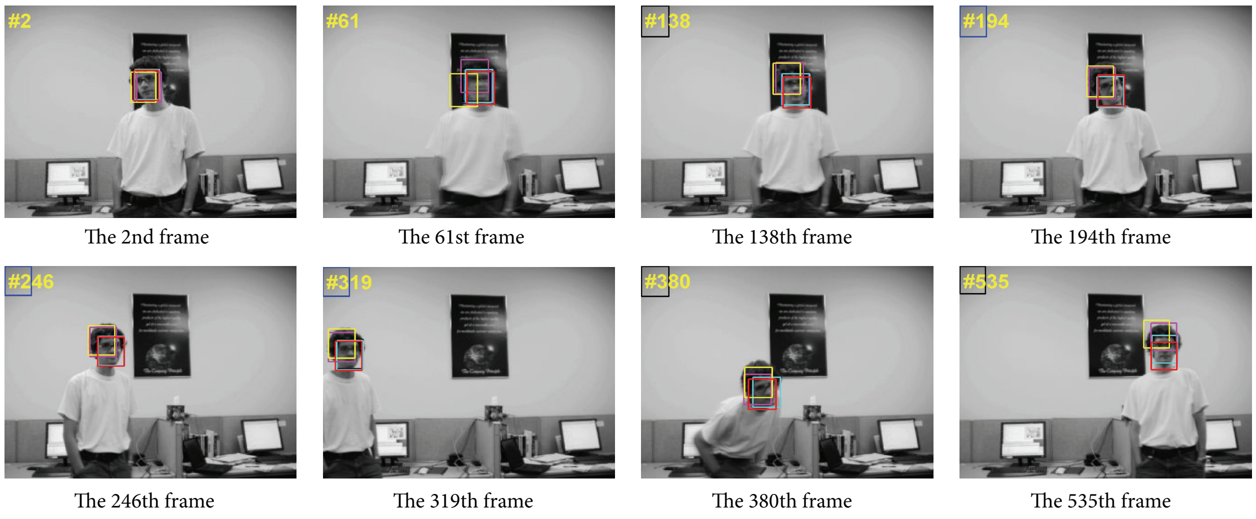


FIGURE 8: A performance comparison of the six algorithms in the 2nd, 61st, 138th, 194th, 246th, 319th, 380th, and 535th frames of David2 video sequences [33].

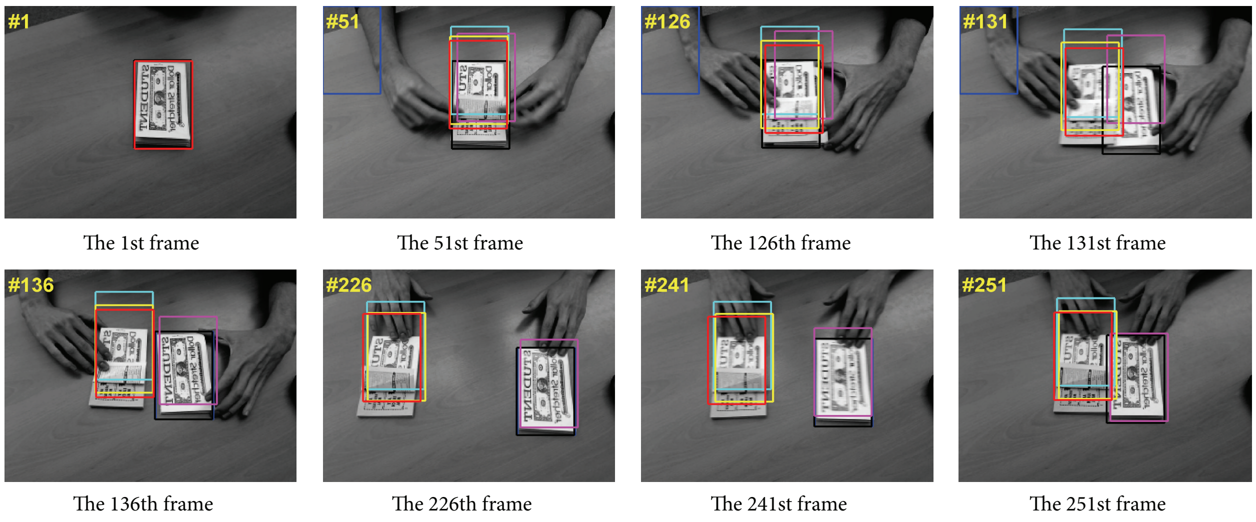


FIGURE 9: A performance comparison of the six algorithms in the 1st, 51st, 126th, 131st, 136th, 226th, 241st, and 251st frames in the dollar video [33].

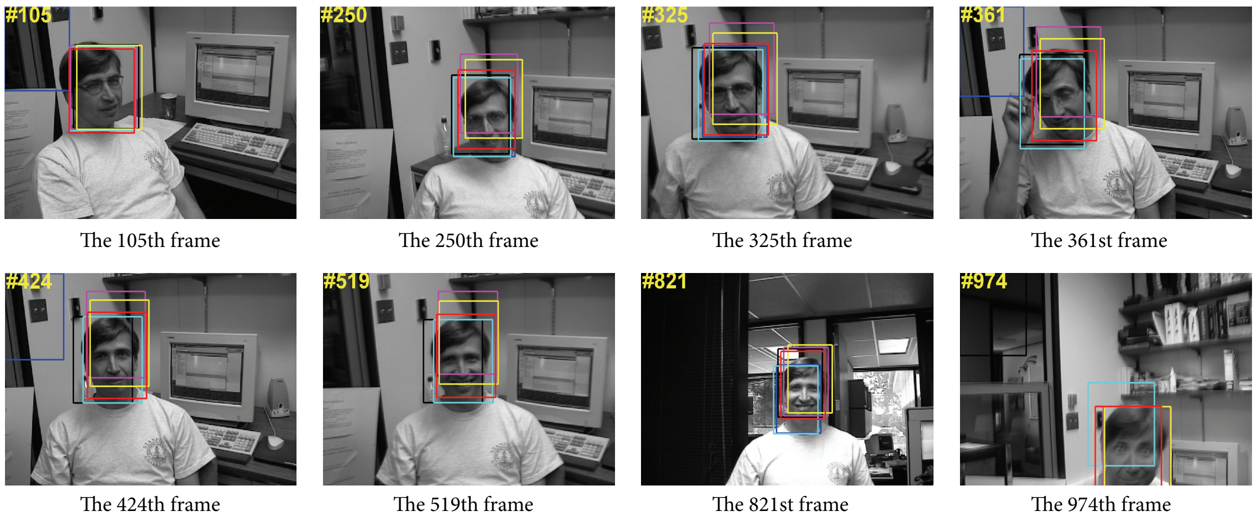


FIGURE 10: A performance comparison of the six algorithms in the 105th, 250th, 325th, 361st, 424th, 519th, 821st, and 974th frames in video Dudek sequences [33].

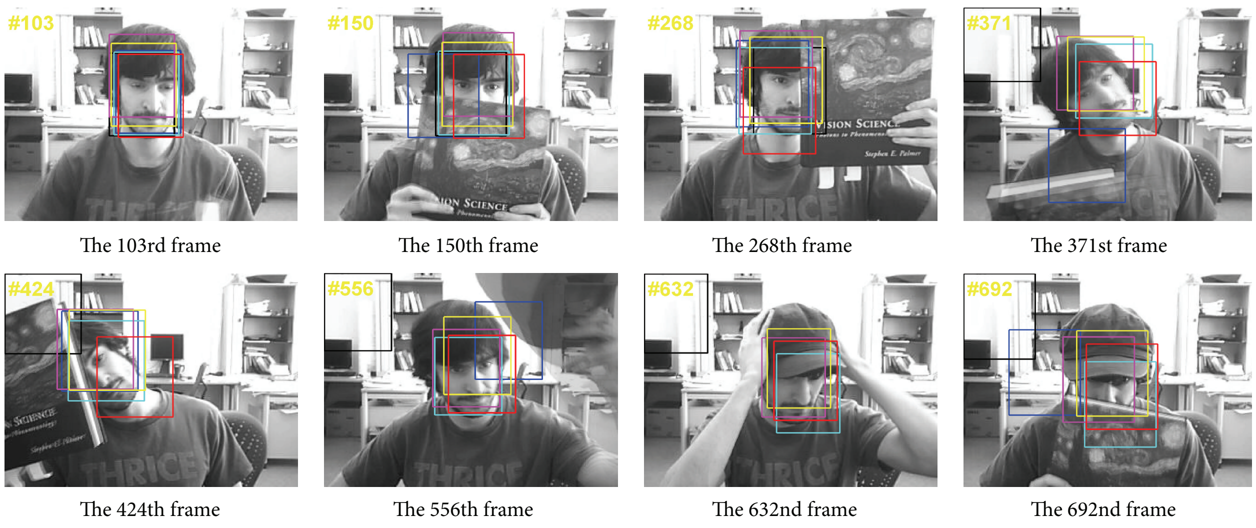


Figure 11: A comparison of the performance of the six algorithms in 103rd, 150th, 268th, 371st, 424th, 556th, 632nd, 692nd frames in video OccFace2 [33].

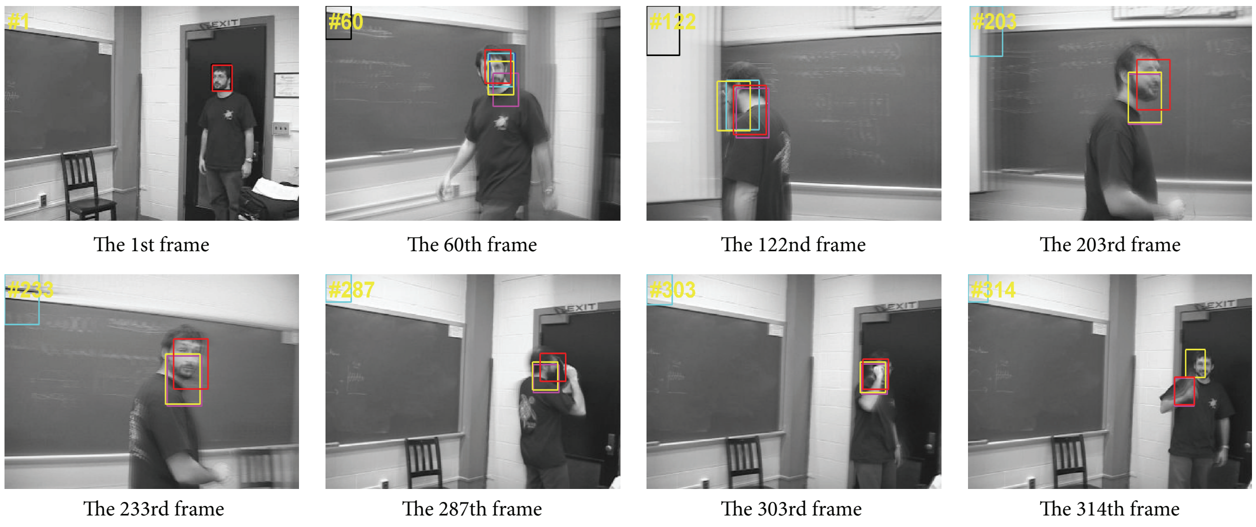


Figure 12: The performance comparison of the six algorithms in the first, 60th, 122nd, 203rd, 233rd, 287th, 303rd, and 314th frames in video Freeman [33].

4.2.5. OccFace2 Video Sequences. The purpose in video OccFace2 is also tracking a person's head. In the video, the person's head was covered by a book or a cap. The performance of the six algorithms is shown in Figure 11. After the 371st frame, BSBT (black) lost object all the time. Comparing the detection effect in the 371st and 556th frames, the influence of covering by book and cap shows a relatively large effect on SBT (blue) and a small effect on other algorithms. Therefore, the performance of CT (yellow), MIL (purple), BT (green), and HOPEF (red) is close in this video.

4.2.6. Freeman Video Sequences. The Freeman video records a person walking from right to left and from far to near. The person took off his glasses and looked around from time to time. The performance of the six algorithms is shown in Figure 12. Before the 303rd frame, MIL (purple), CT (yellow), and HOPEF (red) did not lose the object and basically covered the person's head. After the 303rd frame, only CT (yellow) can capture the object.

4.2.7. Mhyang Video Sequences. The person in the Mhyang video moved around inside the lens. The illumination and the head size slightly changed along with the movement. In the 138th frame, CT (yellow) shifted to the left, which influenced the detection in all the subsequent frames. In the 482nd frame, ML (purple) shifted to the lower left. After the 575th frame, BSBT (black) lost the object from time to time. BT (green) offset in the top right-hand corner of the object. However, the proposed algorithm in this study

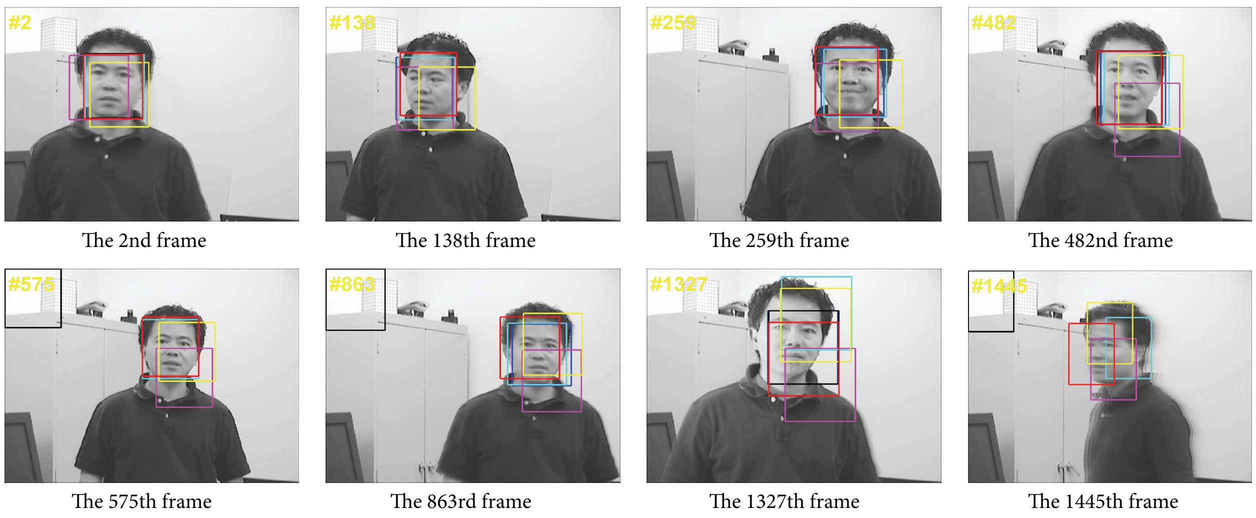


Figure 13: The performance comparison of the six algorithms in the 2nd, 138th, 259th, 482nd, 575th, 863rd, 1327th, and 1445th frames in video Mhyang [33].

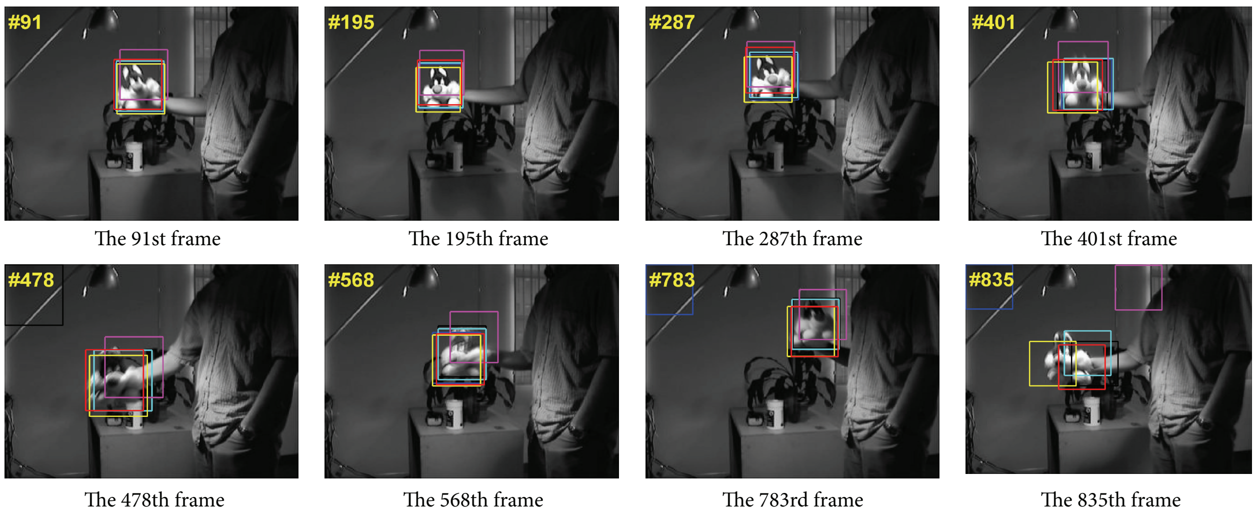


Figure 14: The performance comparison of the six algorithms in the 91st, 195th, 287th, 401st, 478th, 568th, 783rd, and 835th frames in Sylvester video [33].

HOPEF (red) continuously captured the object and showed the least deviation during the entire video (see Figure 13).

4.2.8. Sylvester Video Sequences. The tracking object in Sylvester video sequence is an animal doll with many edges randomly rotating under a lamp. The tracking difficulties of this video are the drastic change of the illumination and big rotation angle. In the 91st, 195th, 287th, and 401st frame, all algorithms could basically capture the object since there were only slight rotation of the animal doll and small illumination change. Compared to the previous ones, in the 478th frame, BSBT (black) lost the object; meanwhile, MIL (purple) began to shift. Between the 478th and 835th frames, the performance of CT (yellow), BT (green), and HOPEF (red) is close. After the 835th frame, the performance of HOPEF (red) is inferior to that of CT (yellow) (see Figure 14).

4.2.9. Gym Video Sequences. The tracking object in Gym video sequence is an athlete. This video proves that our algorithm is robust to pose and illumination changes (see Figure 15).

4.2.10. Trellis Video Sequences. In the video of Trellis, there is a man walking in a greatly varying environment. The extraction difficulties are moving vehicles in the background and the number of buildings surrounding the target. Moreover, when the tracking target slowly walked out from the dim room and the light became stronger, the tracker will face great challenge. According to the experimental result, it can be seen that the tracking effects are ideal with our tracking algorithm. The features used in the study are not sensitive to these factors (especially illumination change), and

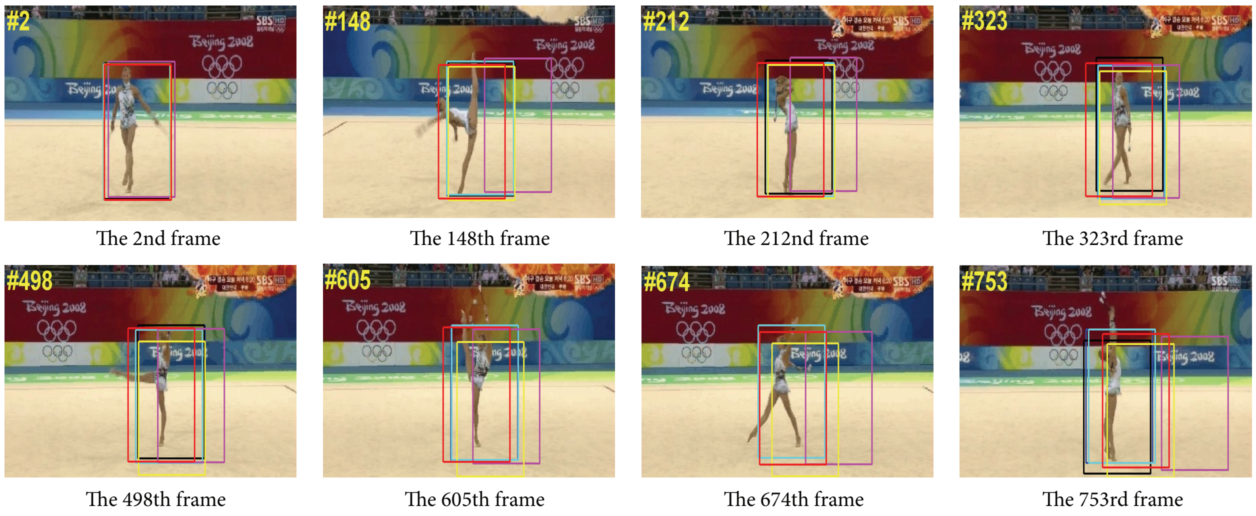


Figure 15: The performance comparison of the six algorithms in the 2nd, 148th, 212nd, 323rd, 498th, 605th, 674th, and 753rd frames in Gym video [33].

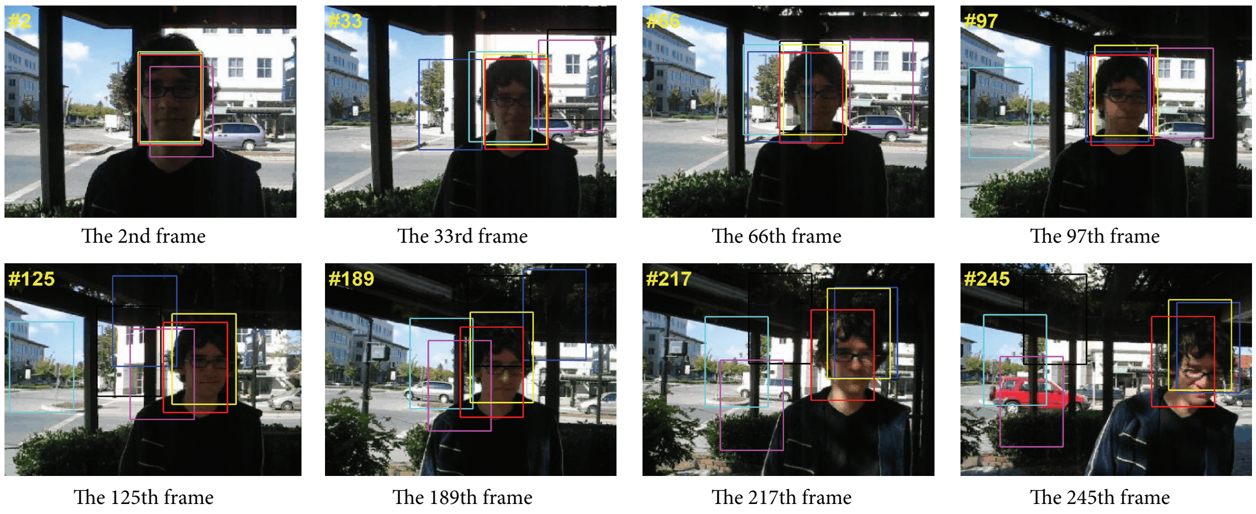


Figure 16: The performance comparison of the six algorithms in the 2nd, 33rd, 66th, 97th, 125th, 189th, 217th, and 245th frames in Trellis video [33].

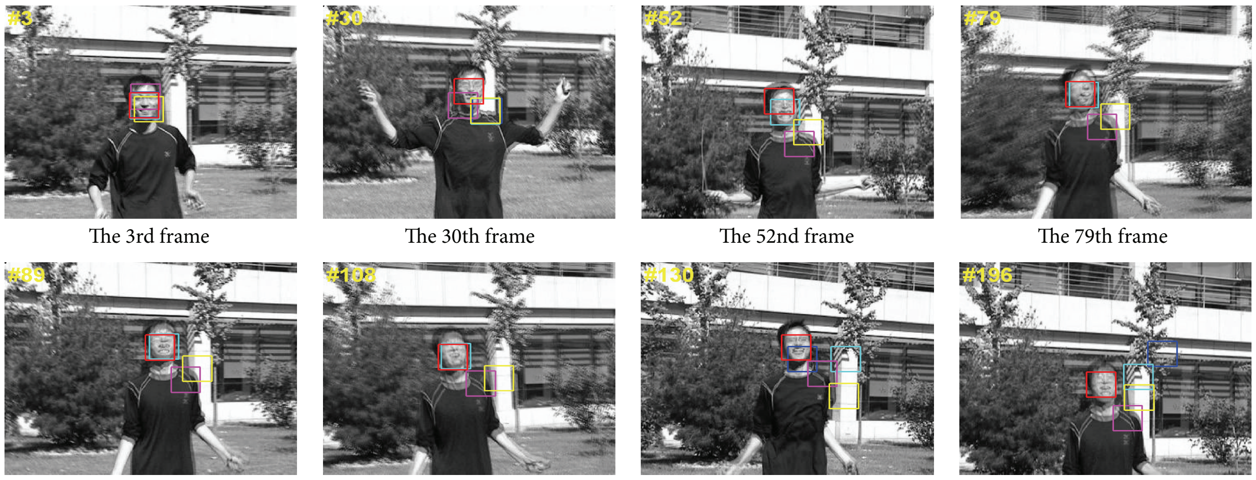
hence the tracking performance is good throughout the video (see Figure 16).

4.2.11. Jumping Video Sequences. In the video of Jumping, there is a man Jumping in a greatly varying environment. The extraction difficulties are motion blur. The performance of the six algorithms is shown in Figure 17. According to the experimental result, it can be seen that the tracking effects are ideal with our tracking algorithm.

4.2.12. Ambiguity in Detection. In this section we discuss how our approach deals with ambiguity in detection. The ORB feature is robust to object rotation, scaling, and noise. The Harr feature is generated statistically, which makes it resilient to object rotation. The combination of these two features provides a better way of describing objects and can reveal differences between multiple objects.

When tracking objects overlap, for instance, the first object is covered by the second, which makes it challenging when sampling image features from the first object as features from the second object may be sampled instead. To deal with this issue, when updating observations for the tracking object, we compare the similarity between the tracking object and other objects. If in the current frame there is a high similarity between the tracking object and a particular other object, we can conclude that the tracking object is covered by that particular object. In this way we can address the ambiguity issue to some extent (see Figure 18).

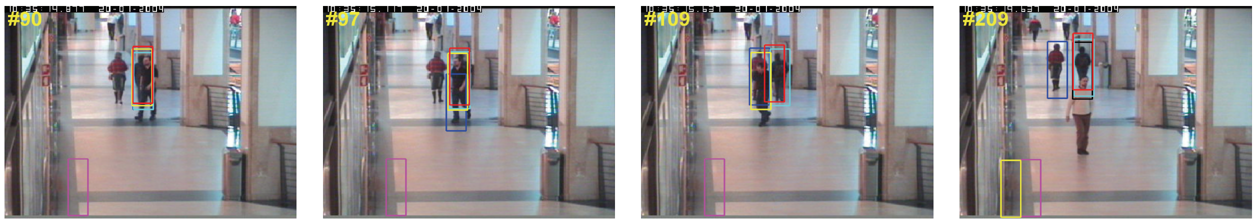
4.3. The Error Comparison in Each Frame. Figure 19 shows the tracking error comparison in Basketball, David2, dollar,



The 3rd frame The 30th frame The 52nd frame The 79th frame

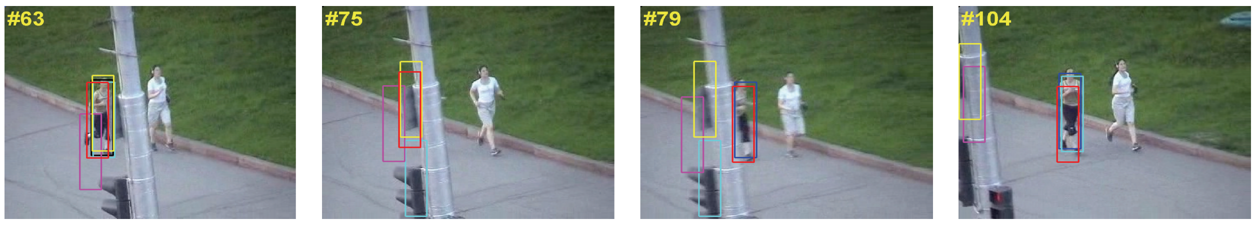
The 89th frame The 108th frame The 130th frame The 196th frame

Figure 17: The performance comparison of the six algorithms in the 3rd, 30th, 52nd, 79th, 89th, 108th, 130th, and 196th frames in Jumping video [33].



The 90th frame The 97th frame The 109th frame The 209th frame

(a) Video source taken from [37]



The 63rd frame The 75th frame The 79th frame The 104th frame

(b) Video source taken from [33]

Figure 18: Ambiguity in detection.

Dudek, FaceOcc2, Freeman1, Mhyang, Sylvester, Gym, Trellis, Singer1, man, Jumping, EnterPaths1, and Jogging video sequences by the six algorithms. In Figure 19, the red line stands for our proposed HOPREF algorithm. It is clear that the red position error curve always stays in the bottom. In addition, from the beginning to the end, the error fluctuation of HOPREF was relatively small, which demonstrated that its tracking performance is stable.

The performance of HOPEF and the other five models is compared from three aspects in this research. The results demonstrate that the proposed tracking method based on the prior probability of entropy of information of the ORB and Haar features is effective and robust.

4.4. Track for Two Objects. Our algorithm can also track for two objects as shown in Figure 20.

In Figure 20, the red rectangle shows the location of the first object, and the green rectangle shows the location of the second object. Our algorithm for tracking two objects can handle occlusion, fast motion, and change of view.

5. Conclusions

In this research we propose a novel object tracking method based on compressed sensing and entropy of information. First, this method adopts the Haar and ORB features to characterize the object. Second, the dimensions of computational

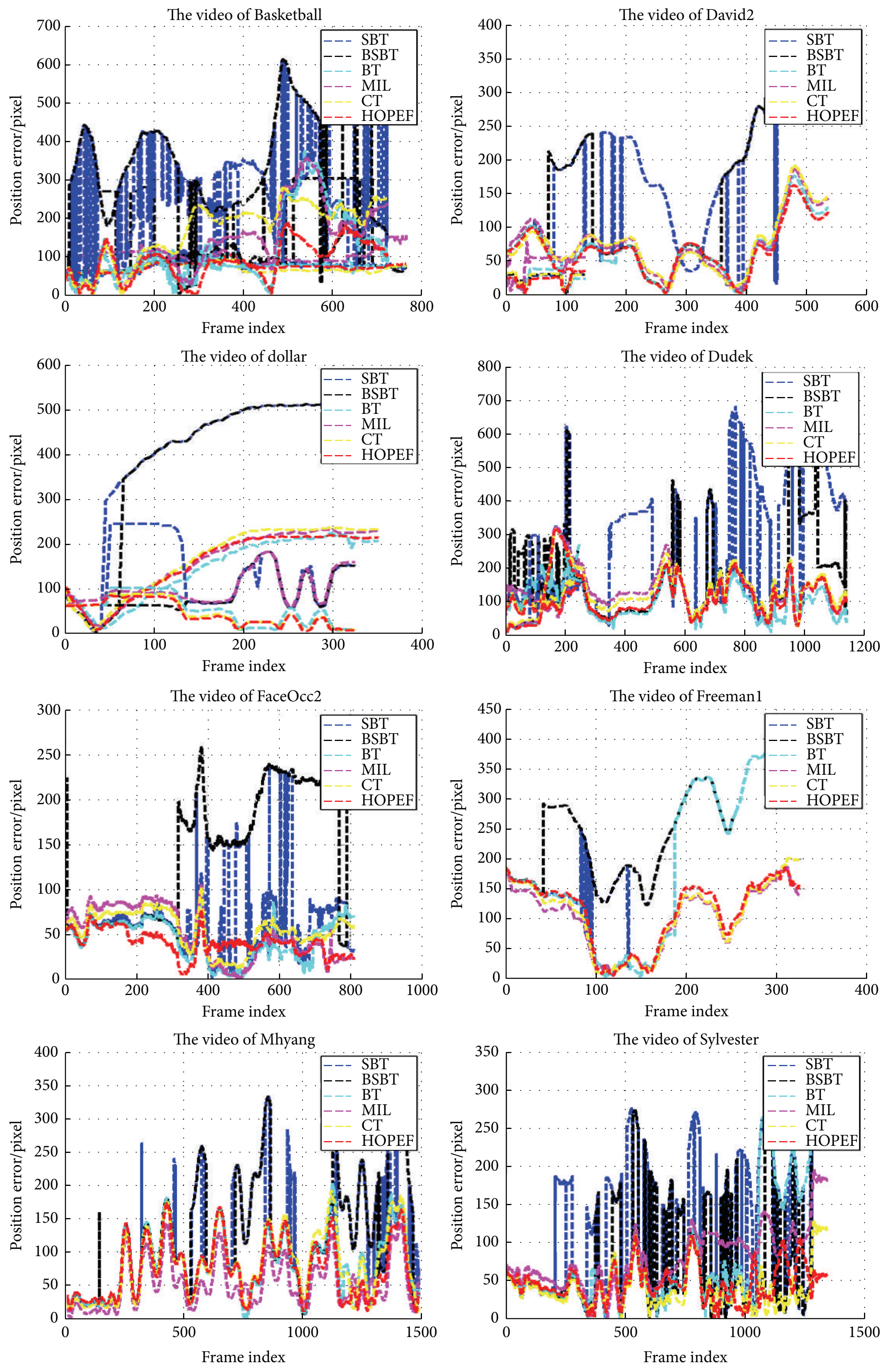


Figure 19: Continued.

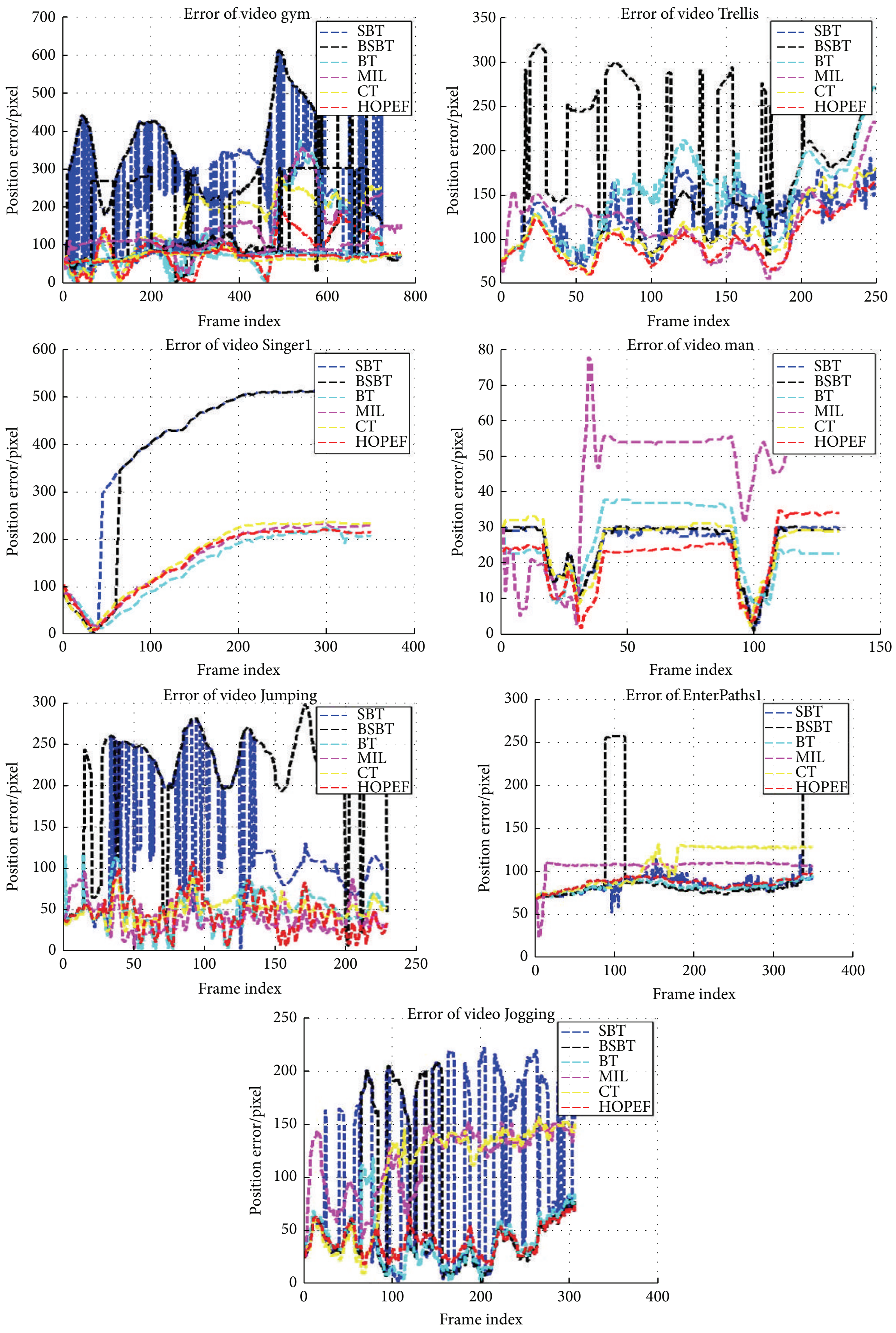
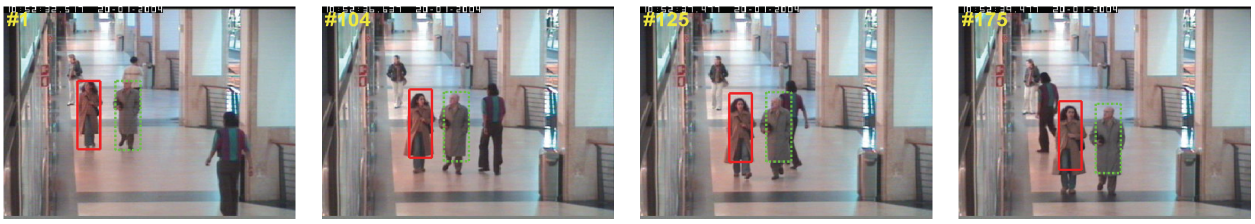
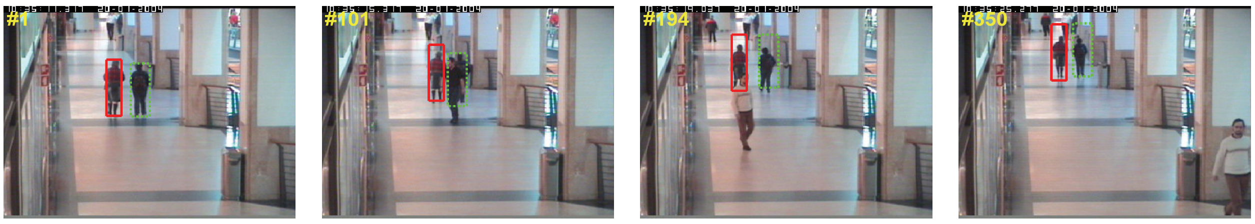


FIGURE 19: The tracking error comparisons in Basketball, David2, dollar, Dudek, FaceOcc2 Freeman1, Mhyang, Sylvester, Gym, Trellis, Singer1, man, Jumping, EnterPaths1, Jogging video sequences by all 6 algorithms.



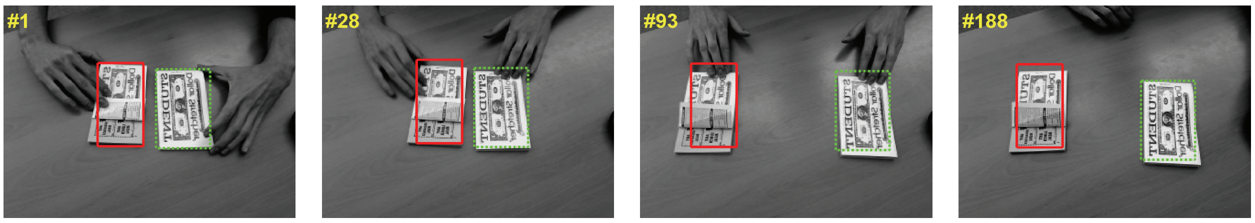
The 1st frame The 104th frame The 125th frame The 175th frame

(a) Video source taken from [37]



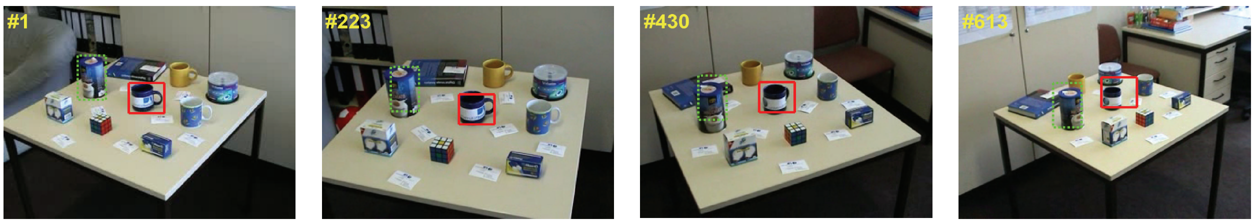
The 1st frame The 101st frame The 194th frame The 350th frame

(b) Video source taken from [37]



The 1st frame The 28th frame The 93rd frame The 188th frame

(c) Video source taken from [33]



The 1st frame The 223rd frame The 430th frame The 613rd frame

(d) Video source taken from [33]

FIGURE 20: Track for two objects.

space of the Haar and ORB features were effectively reduced through compressed sensing. Then the above-mentioned features were fused based on entropy of information. Finally, in the particle filter framework, the object location was obtained by selecting candidate object locations in the current frame from the local areas neighboring the optimal locations in the last frame. Experimental results demonstrated that this method was able to effectively address the challenges of perception change, illumination change, and large area collocation.

However, there is still room for improvement in our algorithm, which will be considered in the future work. First, the situation of losing track of the fast moving object still exists. A self-adaptive method needs to be designed to further improve the tracking performance. Second, according to the experimental results, the detection effect is not satisfactory when the size of tracking object changes dramatically due to the fixed size of identifying window in the entire tracking process. Furthermore, this algorithm cannot take multiple objects (more than two) into consideration, and we aim to address this issue in the next step of our research.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research is supported by (1) the Ph.D. Programs Foundation of Ministry of Education of China under Grant no.

20120061110045, (2) the Science and Technology Development Projects of Jilin Province of China under Grant no. 20150204007G X, and (3) the Key Laboratory for Symbol Computation and Knowledge Engineering of the National Education Ministry of China.

References

[1] D. Wu and D.-W. Sun, "Colour measurements by computer vision for food quality control—a review," *Trends in Food Science & Technology*, vol. 29, no. 1, pp. 5–20, 2013.

[2] M. F. Hashmi and A. G. Keskar, "Analysis and monitoring of a high density traffic flow at T-intersection using statistical computer vision based approach," in *Proceedings of the 12th International Conference on Intelligent Systems Design and Applications (ISDA '12)*, pp. 52–57, IEEE, Kochi, India, November 2012.

[3] M. Wei, C. Li, and D. Luo, "Illegal video surveillance on satellite," *Communications in Computer & Information Science*, vol. 321, pp. 219–226, 2012.

[4] G. Zhao, L. Chen, and G. Chen, "Large head movement tracking using scale invariant view-based appearance model," in *Proceedings of the International Conference on Advanced Concepts for Intelligent Vision Systems*, pp. 331–339, 2007.

[5] S. T. Birchfield and S. Rangarajan, "Spatiograms versus histograms for region-based tracking," in *Proceedings of the IEEE Computer Society Conference on Computer Vision & Pattern Recognition (CVPR '05)*, vol. 2, pp. 1158–1163, June 2005.

[6] A. Yilmaz, X. Li, and M. Shah, "Contour-based object tracking with occlusion handling in video acquired using mobile cameras," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 26, no. 11, pp. 1531–1536, 2004.

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

[8] X. Jia, H. Lu, and M.-H. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, pp. 1822–1829, June 2012.

[9] Y. Bai and M. Tang, "Robust tracking via weakly supervised ranking SVM," in *Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition (CVPR '12)*, pp. 1854–1861, Providence, RI, USA, June 2012.

[10] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '09)*, pp. 983–990, IEEE, Miami, Fla, USA, June 2009.

[11] D. W. Park, J. Kwon, and K. M. Lee, "Robust visual tracking using autoregressive hidden Markov model," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, pp. 1964–1971, IEEE, Providence, RI, USA, June 2012.

[12] R. He, B. Yang, N. Sang, Y. Yu, G. Bai, and J. Li, "Integral region-based covariance tracking with occlusion detection," *Multimedia Tools & Applications*, vol. 74, no. 6, pp. 2157–2178, 2015.

[13] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, "Robust visual tracking via structured multi-task sparse learning," *International Journal of Computer Vision*, vol. 101, no. 2, pp. 367–383, 2013.

[14] G. D. Hager and P. N. Belhumeur, "Efficient region tracking with parametric models of geometry and illumination," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 20, no. 10, pp. 1025–1039, 1998.

[15] I. Matthews, T. Ishikawa, and S. Baker, "The template update problem," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 26, no. 6, pp. 810–815, 2004.

[16] N. Alt, S. Hinterstoisser, and N. Navab, "Rapid selection of reliable templates for visual tracking," in *Proceedings of the IEEE Computer Society Conference on Computer Vision & Pattern Recognition (CVPR '10)*, pp. 1355–1362, June 2010.

[17] X. Mei, H. Ling, Y. Wu, E. Blasch, and L. Bai, "Minimum error bounded efficient $\ell 1$ tracker with occlusion detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '11)*, pp. 1257–1264, June 2011.

[18] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, pp. 886–893, IEEE, San Diego, Calif, USA, June 2005.

[19] O. Tuzel, F. Porikli, and P. Meer, "Region covariance: a fast descriptor for detection and classification," in *Proceedings of the European Conference on Computer Vision*, pp. 589–600, 2006.

[20] F. Porikli, O. Tuzel, and P. Meer, "Covariance tracking using model update based on lie algebra," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 728–735, IEEE, June 2006.

[21] Y. Wu, J. Cheng, J. Wang et al., "Real-time probabilistic covariance tracking with efficient model update," *IEEE Transactions on Image Processing*, vol. 21, no. 5, pp. 2824–2837, 2012.

[22] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[23] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting," in *Proceedings of the British Machine Vision Conference (BMVC '06)*, pp. 47–56, Edinburgh, UK, September 2006, Real-Time Tracking via On-line Boosting, Bmvc,.

[24] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: an efficient alternative to SIFT or SURF," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV '11)*, pp. 2564–2571, November 2011.

[25] I. Ben-Gal and E. Kagan, *Probabilistic Search for Tracking Targets: Theory and Modern Applications*, Wiley & Sons, 2012.

[26] P. Perez, C. Hue, J. Vermaak et al., "Color based probabilistic tracking," in *Proceedings of the European Conference on Computer Vision*, pp. 661–675, May 2002.

[27] E. Maggio and A. Cavallaro, "Hybrid particle filter and mean shift tracker with adaptive transition model," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '05)*, vol. 2, pp. II221–II224, March 2005.

[28] J. P. Kaipio and E. Somersalo, *Statistical and Computational Inverse Problems*, vol. 16, no. 2, Springer, Berlin, Germany, 2005.

[29] T. Li, S. Sun, T. P. Sattar, and J. M. Corchado, "Review: fight sample degeneracy and impoverishment in particle filters: a review of intelligent approaches," *Expert Systems with Applications*, vol. 41, no. 8, pp. 3944–3954, 2014.

[30] H. Nakagawa and H. Takada, "Numerical analysis of rating transition matrix depending on latent macro factor via nonlinear particle filter method," *Journal of Financial Engineering*, vol. 1, no. 3, 2014.

[31] K. Zhang, L. Zhang, and M.-H. Yang, “Real-time compressive tracking,” in *Computer Vision—ECCV 2012*, vol. 7574 of *Lecture Notes in Computer Science*, pp. 864–877, Springer, Berlin, Germany, 2012.

[32] D. Achlioptas, “Database-friendly random projections: johnson-lindenstrauss with binary coins,” *Journal of Computer & System Sciences*, vol. 66, no. 4, pp. 671–687, 2003.

[33] Visual tracker benchmark dataset, 2015, http://cvlab.hanyang.ac.kr/tracker_benchmark/datasets.html.

[34] K. Zhang, L. Zhang, Q. Liu, D. Zhang, and M.-H. Yang, “Fast visual tracking via dense spatio-temporal context learning,” in *Computer Vision—ECCV 2014*, vol. 8693 of *Lecture Notes in Computer Science*, pp. 127–141, Springer, Cham, Switzerland, 2014.

[35] C. E. Shannon and W. Weaver, “The mathematical theory of communication,” *M.D. Computing*, vol. 14, no. 3, pp. 3–55, 1949.

[36] C. Ó. Conaire, N. E. O’Connor, and A. Smeaton, “Thermo-visual feature fusion for object tracking using multiple spatiogram trackers,” *Machine Vision & Applications*, vol. 19, no. 5-6, pp. 483–494, 2008.

[37] CAVIAR Test Case Scenarios, May 2015, http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/.

[38] S. Stalder, H. Grabner, and L. Van Gool, “Beyond semi-supervised tracking: tracking should be as simple as detection, but not simpler than recognition,” in *Proceedings of the IEEE 12th International Conference on Computer Vision Workshops (ICCV ’09)*, pp. 1409–1416, October 2009.

[39] H. Grabner, C. Leistner, and H. Bischof, “Semi-supervised on-line boosting for robust tracking,” in *Computer Vision—ECCV 2008: Proceedings of the 10th European Conference on Computer Vision, Marseille, France, October 12–18, 2008, Part I*, vol. 5302 of *Lecture Notes in Computer Science*, pp. 234–247, Springer, Berlin, Germany, 2008.