A novel visual tracking system with adaptive incremental extreme learning machine

Zhihui Wang[1], Sook Yoon[2,3,*] and Dong Sun Park[4,5]
[1] National Key Laboratory, Hisense Company Limited, Qingdao, China
[e-mail: zhihuiwangjl@gmail.com]
[2] Department of Multimedia Engineering, Mokpo National University, Jeonnam, Korea
[3] Research Institute of Realistic Media and Technology, Mokpo National University, Jeonnam, Korea
[e-mail: syoon@mokpo.ac.kr]
[4] Division of Electronics Engineering, Chonbuk National University, Jeonju, South Korea
[5] IT Convergence Research Center, Chonbuk National University, Jeonju, South Korea
[e-mail: dspark@jbnu.ac.kr]
*Corresponding author: Sook Yoon

Received March 7, 2016; revised August 24, 2016; accepted December 4, 2016; published January 31, 2017

Abstract

This paper presents a novel discriminative visual tracking algorithm with an adaptive incremental extreme learning machine. The parameters for an adaptive incremental extreme learning machine are initialized at the first frame with a target that is manually assigned. At each frame, the training samples are collected and random Haar-like features are extracted. The proposed tracker updates the overall output weights for each frame, and the updated tracker is used to estimate the new location of the target in the next frame. The adaptive learning rate for the update of the overall output weights is estimated by using the confidence of the predicted target location at the current frame. Our experimental results indicate that the proposed tracker can manage various difficulties and can achieve better performance than other state-of-the-art trackers.

Keywords: Extreme learning machine, visual tracking, overall output weights, random Haar-like features, adaptive learning rate

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(No.NRF-2016R1D1A1B03936269)

1. Introduction

Object tracking has received widespread attention due to its wide range of applications in various industries, including the fields of video surveillance, medical imaging systems, intelligent user interfaces, etc [1, 2]. Numerous tracking algorithms that are suitable for specific targets under certain circumstances have been proposed [3–9]. All of these tracking methods are effective when operating under the given assumptions of their design during experimental simulations. In the process of tracking, the target objects always undergo various kinds of challenges and problems, such as occlusion, illumination variation, camera movement, and image blur, and it is still difficult to manage these challenges effectively and in a timely manner.

The online tracking algorithms must update their parameters to manage the aforementioned issues. Therefore, the strategy used to update the parameters is rather important for the online tracking stage. A visual tracking method with an online boosting feature selection has been used in Ref. [10]. Also, an IVT tracker can learn and incrementally update a low-dimensional eigenspace representation [11]. A TLD tracker explicitly decomposes the visual tracking task into three portions: tracking, learning, and detection [12]. In order to deal with the drifting problems, multiple instance learning has been adopted in the MIL tracker with random Haar-like features [13]. A locality sensitive histogram has been proposed and has used for visual tracking [14]. Most of the existing tracking algorithms use a fixed learning rate to update the parameters per frame. When the target object has a poor appearance, the parameter update may have a negative effect on the tracking results of the following frames. If the information of the target history is handled improperly, drifting problems are inevitable in these situations.

An extreme learning machine (ELM) is a rather powerful and efficient machine learning algorithm [15-18]. An ELM algorithm has been extensively applied to many research problems that require regression or pattern classification. Various improved algorithms have been proposed during the past decade [19–23], including OS-ELM, which can deal with sequentially arriving data [24]. The sample blocks are trained individually, and the output weights are updated without performing any repetitive computation. However, the contributions of these training samples are equal during the estimation of the output weights. The proposed adaptive incremental extreme learning machine (AI-ELM) uses a confidence strategy to update its overall output weights at every frame. The confidence is used to assign a searching area to collect training samples and to calculate the learning rate that is used to update the weights of the AI-ELM. Instead of updating the parameters using a constant learning rate, the proposed AI-ELM tracker uses a variable learning rate. This variable learning rate depends on the confidence value of the estimated target at the current frame. When the confidence is lower than the given threshold, the parameter update is cancelled at that frame. In this way, the parameters are updated adaptively based on the state of the target appearances.

The remainder of the paper is organized as follows. The proposed AI-ELM tracking system is described in Section 2. An experimental comparison of the proposed tracker against other state-of-the-art trackers is demonstrated in Section 3. Finally, the conclusions are drawn in Section 4.

2. The proposed tracking system

The proposed visual tracking system (AI-ELM tracker) implements an adaptive, incremental strategy to update the parameters of the extreme learning machine. We first briefly introduce a system overview of the proposed AI-ELM tracker in Subsection 2.1. Then, we explain the collection strategy for the training samples, the principle of the AI-ELM algorithm, the object location estimation method, and the adaptive parameter update for the AI-ELM tracker in Subsections 2.2, 2.3, 2.4 and 2.5.

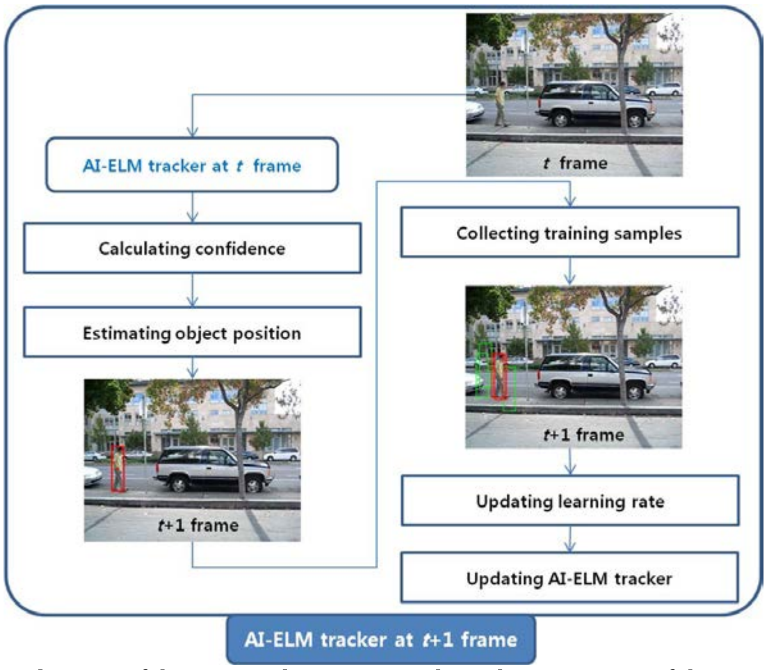

AI-ELM tracker at t frame
t frame
Calculating confidence
Collecting training samples
Estimating object position
t+1 frame
t+1 frame
Updating learning rate
Updating AI-ELM tracker
AI-ELM tracker at t+1 frame

Fig. 1. Basic flow diagram of the proposed AI-ELM tracker. The parameters of the AI-ELM tracker are initialized at the first frame, and are updated frame by frame using an adaptive learning rate. The red rectangles are positive training samples, and the green rectangles are negative training samples.

2.1. System overview

In the proposed tracking system, the tracking process is produced with the proposed AI-ELM tracker. After the target object is initialized in the first frame, the target's location on each frame is estimated in the searching region. On each frame, several target candidates are extracted from the searching region, and their confidences are calculated by the AI-ELM tracker. The target candidate with the best confidence is selected as the target's location on this frame. Only targets with the confidence higher than a given threshold are used to update the parameters of AI-ELM tracker to estimate the new location on the next frame.

As the basic flow diagram shows in **Fig. 1**, the target object position at frame $t + 1$ is estimated using the confidence values calculated by the AI-ELM tracker at frame t, while the center and the size of the target is selected manually at the first frame. Some positive and negative training samples, with their given numbers, are collected around the target object. The parameters of the AI-ELM tracker are initialized at the starting frame along with the information of the collected training samples. In the following procession, the parameters of the proposed tracking system are updated using the latest collected training through an adaptive learning rate per frame. The updated tracker is then used to estimate the location of the target in the search region around the current location at the next frame.

The details of the proposed visual tracking system are detailed next.

2.2. Collection of training samples

For each frame, positive and negative training samples are collected around the target object to update the parameters of the AI-ELM tracker. At frame t, some positive training samples $\aleph_t^+ = \{x_{i,t}^+\}_{i=1}^{n_1}$ and some negative training samples $\aleph_t^- = \{x_{i,t}^-\}_{i=1}^{n_2}$ are collected from the corresponding regions $\{x \mid \| l(x) - l(x_t) \| < r_1\}$ and $\{x \mid r_2 \leq \| l(x) - l(x_t) \| < r_3\}$, where $l(\cdot)$ is the location function; x_t is the center of the target object at frame t; n_1 and n_2 are the given numbers for the positive and negative training samples; and $\{r_1,\ r_2,\ r_3\}$ are region radii following the relation $r_1 < r_2 < r_3$. The sizes of all of the training samples are the same as those of the target object.

Then, random Haar-like features $\mathbf{h}(x) = [h_1(x), \cdots, h_K(x)]^\top$ [13, 25], where K is the amount of feature dimensions, are extracted from these training samples $x \in \left\{\aleph_t^+, \aleph_t^-\right\}$, and all the components of $\mathbf{h}(x)$ are normalized. (For the first frame, all of the components are normalized into $[-1, 1]$. For the following frames, these components are normalized based on the standardized scale of the first frame.) The corresponding outputs of the positive and the negative training samples are ones and zeros, and the extreme learning machine is trained using a regressive pattern. Compared with other image features, such as Histogram of Oriented Gradient (HOG), random Haar-like features is rather suitable for the tracker due to their relative simplicity and efficiency. Since the tracking algorithm is processed online, the features should be extracted fast enough to guarantee the efficiency of the tracking system, while keeping the high accuracy of tracking results.

2.3. Principle of adaptive incremental extreme learning machine (AI-ELM)

In a manner similar to that of the original ELM algorithm [15, 16], the hidden node parameters $\{(\mathbf{a}_i,\ b_i)\}_{i=1}^{L}$ of the AI-ELM are randomly selected before tracking, where $\mathbf{a}_i = \{a_{i,j}\}_{j=1}^{K}$ and L represents the number of hidden nodes. The ranges of the values for $a_{i,j}$ and b_i are $[-1, 1]$ and $[0, 1]$, correspondingly. At frame t, the hidden node output function of the ith additive hidden node of the training sample $x \in \left\{\aleph_t^+, \aleph_t^-\right\}$ is

$$G(\mathbf{a}_i, b_i, \mathbf{h}(x)) = g(\mathbf{a}_i \cdot \mathbf{h}(x) + b_i). \tag{1}$$

According to the theory of an extreme learning machine and to the Moore-Penrose pseudo inverse [26], the optimal output weights $\boldsymbol{\beta}_t = [\beta_{t,1}, \cdots, \beta_{t,L}]^\top$ are estimated as

$$\boldsymbol{\beta}_t = \underset{\boldsymbol{\beta} \in \mathbb{R}^L}{\arg\min} \| \mathbf{H}_t\boldsymbol{\beta} - \mathbf{T_t} \| = \mathbf{H}_t^\dagger \mathbf{T}_t, \tag{2}$$

where the hidden layer output matrix is

$$\mathbf{H}_t(\mathbf{a}_1,\cdots,\mathbf{a}_L,b_1,\cdots,b_L,\aleph^+,\aleph^-)$$
$$=\begin{bmatrix} G(\mathbf{a}_1,b_1,\mathbf{h}(x^+_{1,t})) & \cdots & G(\mathbf{a}_L,b_L,\mathbf{h}(x^+_{1,t})) \\ \vdots & \ddots & \vdots \\ G(\mathbf{a}_1,b_1,\mathbf{h}(x^+_{n_1,t})) & \cdots & G(\mathbf{a}_L,b_L,\mathbf{h}(x^+_{n_1,t})) \\ G(\mathbf{a}_1,b_1,\mathbf{h}(x^-_{1,t})) & \cdots & G(\mathbf{a}_L,b_L,\mathbf{h}(x^-_{1,t})) \\ \vdots & \ddots & \vdots \\ G(\mathbf{a}_1,b_1,\mathbf{h}(x^-_{n_2,t})) & \cdots & G(\mathbf{a}_L,b_L,\mathbf{h}(x^-_{n_2,t})) \end{bmatrix}_{(n_1+n_2)\times L},$$

and $\mathbf{T}=[1,\cdots,1,0,\cdots,0]^\top$ are the outputs of the positive and the negative training samples concatenated with n_1 ones and n_2 zeros. $\mathbf{H}_t^\dagger=(\mathbf{H}_t^T\mathbf{H}_t)^{-1}\mathbf{H}_t^T$is the Moore-Penrose generalized inverse of the hidden layer output matrix $\mathbf{H}_t$ when $\mathbf{H}_t$ is nonsingular. Otherwise, $\mathbf{H}_t^\dagger$ is estimated based on the method of singular value decomposition. The output of the AI-ELM is

$$P(x)=\sum_{i=1}^{L}B_{i,t}G(\mathbf{a}_i,b_i,x), \tag{3}$$

where $B_{i,t}$ is the ith element of the overall output weights at frame t, $\mathbf{B}_t$, which is updated with an adaptive learning rate η_t. Compared with the original ELM algorithm, overall output weights of the proposed AI-ELM tracker contain the historical information and are updated only with high confidences, which are explained in subsection 2.5. The output value is used to estimate the position of the object at the next frame and to calculate its confidence value and learning rate.

As mentioned above, the parameters of the proposed AI-ELM tracker are updated at each frame using the latest collected training samples. The updated tracker is then used to search for the target object in the next frame. Suppose that the overall output weights $\mathbf{B}_t$ are updated at frame t. Then, the new target location at frame $t+1$ can be searched exhaustively in the region $X_{t+1}^\gamma=\{x \mid \parallel l(x)-l(x_t)\parallel\leq\gamma\}$ around the previous target center x_t, where γ is the positive search radius. The confidences for the candidates $\left\{x|x\in X_{t+1}^\gamma\right\}$ are estimated by using Eq. (4).

The maximum confidence value of all the candidates in the search region, C_t, is used to calculate the learning rate to update the output weight B_{t+1} at frame $t+1$ and defined as

$$C_t=\max\left\{P(x)|x\in X_{t+1}^\gamma\right\} \tag{4}$$

And the location of the target object with this maximum confidence $l(x_{t+1})$ where $x_{t+1}=\arg\max_{x\in X_{t+1}^\gamma}P(x)$ is accepted as the new object location at frame $t+1$.

2.5. Parameter updating using adaptive learning rate

For visual tracking, the parameter update is a rather important process that is used to manage the variation in the appearance of the targets. The strategy used to update the parameters affects the tracking capabilities of the specific tracking method used. Instead of performing a parameter update using a constant learning rate, we use an adaptive learning rate to update the tracking system parameters during the online tracking process.

For the AI-ELM tracker, the tracking system parameters are updated at each frame, and the learning rate is estimated adaptively. These parameters are initialized at the first frame, and for the following frames, the learning rate of the parameter update is estimated according to the confidence value of the target, which is estimated by the tracker updated at the previous frame.

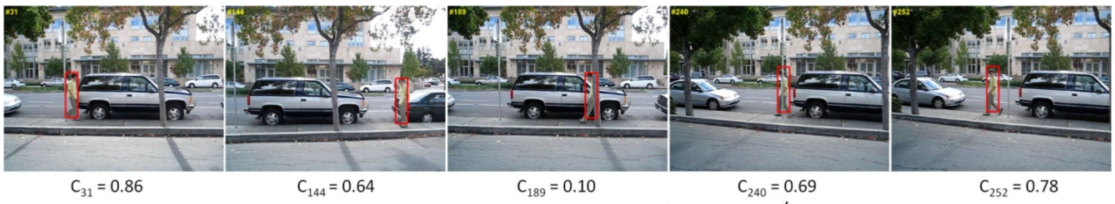


Fig. 2. Examples of the varied confidences. The confidence C_t at frame t is the maximum value of all candidates in the searching region, which is predicted by the proposed AI-ELM tracker.

Under the assumption that there cannot be a large variation of the target object for consecutive frames, the confidence value of the selected target at each frame can therefore reflect appearance of the target. Large variations in this respect may be caused by different reasons, including occlusions and illumination changes. Such conditions always have a negative impact causing a poorly estimated confidence for the selected target. As shown in **Fig. 2**, The confidence C_{31} of the target estimated at frame #31 is 0.86, which is a high value, since there are seldom any variations in the appearance of the pedestrian. When the pedestrian turns and walks in the opposite direction at frame #144, the confidence is reduced to a value of 0.64. At frame #189, the confidence is even worse and just reaches 0.1 since there is heavy occlusion. The confidence at frame #240 is 0.69 since there is light occlusion. Finally, the confidence is 0.78 at frame #252, which is higher than that of the other frames that have shape changes and occlusions.

Drifting problems are inevitable after the parameter is updated for these frames. These training samples at these frames with interferences resulting from internal or external factors have worse performance than that of other unaffected frames. Moreover, the confidence value can briefly reflect the degree of these influences. Therefore, the learning rates of the proposed AI-ELM tracker are limited at these frames based on the variation of the confidence values.

In this paper, the overall output weights $\mathbf{B}_t$ are updated using an adaptive learning rate η_t as follows

$$\mathbf{B}_t = (1-\eta_t)\mathbf{B}_{t-1} + \eta_t\boldsymbol{\beta}_t, \quad t = 1, 2, \cdots, \tag{5}$$

where $\mathbf{B}_0 = \mathbf{0}$, the learning rate $\eta_1 = 1$, and $\eta_t \ (t = 2, 3, \cdots)$ is calculated by

$$\eta_t = \begin{cases} 0, & C_t \le \theta; \\ \frac{s_1 + C_t}{s_1 + 1} \cdot s_2, & \theta < C_t \le 1; \\ s_2, & 1 < C_t; \end{cases} \tag{6}$$

where (s_1, s_2) are positive constant parameters, θ is the threshold for C_t, and the maximum confidence of all the candidates in the search region X_t^{γ} at frame t estimated by the previous updated tracker at frame $t-1$ is defined in subsection 2.4. Therefore, the learning rate is adjusted adaptively and steadily. **Fig. 3** shows illustrates the relationship between the learning rate η_t and the estimated confidence C_t.

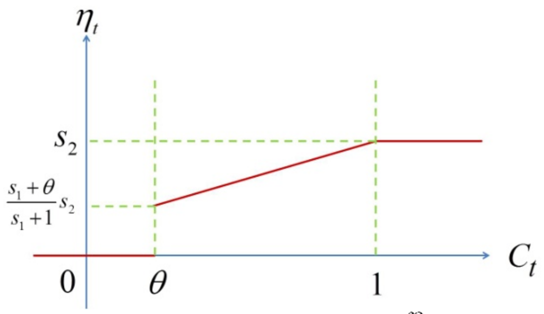


Fig. 3. The relationship between the learning rate η_t and the confidence C_t.

When $C_t \le \theta$, the overall output weights $\mathbf{B}_t$ are not updated and are assigned to be $\mathbf{B}_{t-1}$. The learning rate η_t linearly increases by C_t in the region $[\theta, 1]$. When $C_t > 1$, η_t is suppressed to a constant learning rate s_2. From Eq. (5), we obtain the relation between the overall output weights $\mathbf{B}_t$ and all of the calculated output weights $\{\boldsymbol{\beta}_i\}_{i=1}^{t}$ by

$$\mathbf{B}_t = \sum_{i=2}^{t-1}\left(\eta_i\boldsymbol{\beta}_i \prod_{j=i+1}^{t}(1-\eta_j)\right). \tag{7}$$

Therefore, the updated parameters contain sufficient information from these passed frames. The learning rate η_t is calculated as a tradeoff between the latest output weights $\boldsymbol{\beta}_t$ trained at frame t and the previous overall output weights $\mathbf{B}_{t-1}$, and the updated overall output weights $\mathbf{B}_t$ are used to search the new target location at the following frame $t+1$.

The pseudo-code for the proposed AI-ELM tracker is presented in Algorithm 1.

Algorithm 1 The proposed AI-ELM algorithm

```
Input: Given the the hidden node output function $G(\mathbf{a}, b, \mathbf{x})$, hidden node number $L$, and
    the testing sequence with $N$ frames, assign randomly hidden node parameters $\mathbf{a}_i$ and $b_i$,
    $i = 1, \cdots, L$, from the corresponding regions [-1,1] and [0,1].
 1: Select the center and size of the target object $x_1$ at the first frame manually;
 2: initialize the overall output weights $\mathbf{B}_0 = \mathbf{0}$, and $\eta_1 = 1$;
 3: for $t = 1$ to $N-1$ do
 4:     collect the training samples $\{\aleph_t^+, \aleph_t^-\}$ at frame $t$;
 5:     extract the random Haar-like features of $\{\aleph_t^+, \aleph_t^-\}$;
 6:     calculate the hidden layer output matrix $\mathbf{H}_t$;
 7:     assign the outputs $\mathbf{T}_t$ with ones for $\aleph_t^+$ and zeros for $\aleph_t^-$;
 8:     estimate the output weights $\boldsymbol{\beta}_t = \mathbf{H}_t^\dagger \mathbf{T}_t$;
 9:     update the overall output weights $\mathbf{B}_t = (1-\eta_t)\mathbf{B}_{t-1} + \eta_t\boldsymbol{\beta}_t$;
10:     calculate the confidences of all candidates $\left\{P(x) | x \in X_{t+1}^{\gamma}\right\}$ exhaustively;
Output: the new target location $x_{t+1}$ at frame $t+1$ with the maximum confidence;
11:     estimate $\eta_{t+1}$ with Eq. (6).
12: end for
```

3. Experiments

In this section, we compare the proposed AI-ELM tracker against five state-of-the-art trackers using seven publicly available video sequences that present various tracking challenges. For the comparison, we use the default parameters for the five trackers, L1APG [27], LOT [28], ORIA [29], IVT [11], and MIL [13]. Four test video sequences ($Cliffbar$, $DavidIndoor$, $Tiger2$, and $Twinnings$) are from the MIL tracker [13] using a 'png' format. The other three video sequences ($Basketball$, $DavidOutdoor$, and $FaceOcc2$) are from Tracker Benchmark v1.0 [30] using a 'jpeg' format. The other three video sequences (*PolarBear*, *Sphere*, and *Surfing*) are from VOT14 Benchmark [31] with `jpeg' format. All of the performance evaluations were carried out in MATLAB R2013a running on a PC with an Intel(R) Core(TM) i5-2500 CPU at 3.30 GHz and 16 GB RAM. In order to ensure consistency in the random number generation, we reset the random number generator in MATLAB to its initial state. The parameter settings are presented in Subsection 3.1, and the detailed results of the comparison are discussed in Subsections 3.2 and 3.3.

3.1. Parameter settings

The region radii r_1, r_2 and r_3 are selected based on the sizes of tracked objects and their traveling speeds. The searching radius r_3 should be selected based on traveling speeds of the tracked targets. When the target objects are larger, the region radius r_1 should be set to be larger than the assigned value. Furthermore, when the traveling speeds are faster than the tested samples, their region radii r_2 and r_3 should be set to be larger than the assigned values. Otherwise, the targets cannot be followed when the targets leave the searching regions. In our experiments, the region radii of the training sample collection are set to be $r_1 = 4,\ r_2 = 6 \text{ and } r_3 = 50$, respectively, for the given video sequences. All of the samples in the positive training sample region $\aleph_t^+$ are collected with number $n_1 = 45$, and a randomly selected $n_2 = 50$ negative training samples from region $\aleph_t^-$ at frame t. The numbers of positive and negative training samples are selected based on the overall performances of the AI-ELM tracker, and the numbers of training samples can be set to be larger to enhance the robustness of the tracking accuracies. The dimension of the extracted random Haar-like features is 150, and the number of hidden nodes of the extreme learning machine are 200. The dimension of the extracted features was investigated in the region of [100, 200] with step 10 and the number of hidden nodes is selected in the region of [100, 300] with step 20, based on their performances of the experiments. The positive constant for the learning rate η_t is set to $s_1 = 5,\ s_2 = 0.08$, and the threshold θ is set to 0.2. The parameters s_1 and s_2 are used to control the learning rate η_t, θ is the threshold for the estimated confidence C_t. When the confidence C_t is lower than the threshold θ, the situation of the tracked target is not good enough. In these cases, the targets are always suffer with heavy occlusions or deformations. Therefore, it is desirable not to update the parameters of AI-ELM tracker at these frames.

3.2. Quantitative analysis

In order to demonstrate the superiority of the proposed AI-ELM tracker, we compared the failure rates, the average frame tracking speeds, and the center location errors of the proposed tracker against five other state-of-the-art trackers. A failure frame is indicated when the intersection of R_t and R_g is less than half of their union, where R_t is the tracking bounding box and R_g is the ground truth bounding box. The center location errors are evaluated in terms of the maximum value, mean value, and the standard deviation.

A comparison of the failure rate (FR) and average frame per second (FPS) are presented in **Table 1**. The failure rates of the proposed AI-ELM tracker are lower than 30% on all seven testing video sequences, and even lower than 10% on four of these. For the other trackers, the failure rates are all higher than 20%, except only for the LOT tracker on $DavidOutdoor$ and $Polarbear$ which has a good result with failure rates of 3.9% and 4.5%. Compared to the other five trackers, the proposed AI-ELM tracker has the lowest failure rates on all of the testing sequences.

The average FPS of the proposed AI-ELM tracker reaches 28, which is faster than that of the other five trackers and is adequate enough to deal with the online tracking applications. The training time for the MIL tracker at each frame is linearly proportional to the dimension of the extracted random Haar-like features. However, the feature dimensions have little influence on the computing time of the output weights $\boldsymbol{\beta}_t$ at frame t. Since the extracting time of the random Haar-like features is rather short, the tracking time of the proposed AI-ELM tracker is increasingly slow as the feature dimension rises.

Table 1. The failure rate (FR) (%) and the average frame per second (FPS) comparison of the proposed AI-ELM tracker and other five state-of-the-art trackers.

Sequence	AI-ELM	L1APG	LOT	ORIA	IVT	MIL
Cliffbar	6.0	56.0	90.9	77.2	92.4	40.9
DavidIndoor	1.0	61.2	94.6	74.1	85.0	58.1
Tiger2	20.5	72.6	97.2	84.9	98.6	20.8
Twinnings	11.7	38.2	43.6	72.3	58.5	23.4
Basketball	29.1	71.1	41.2	98.4	98.6	71.6
DavidOutdoor	3.1	54.3	3.9	98.6	98.4	41.7
FaceOcc2	1.1	20.5	56.4	28.5	63.8	30.1
PolarBear	26.1	41.2	4.5	41.5	53.6	26.4
Sphere	10.4	0	66.6	93.5	97.5	86.1
Surfing	2.1	0	28.0	14.5	11.7	0
Average FPS	28	1	1	10	17	5

[1]Red fonts indicate the best performance, while blue fonts indicate the second best results.

Tables 2 and **3** show a comparison of the center location errors of the proposed AI-ELM tracker and of the other five state-of-the-art trackers. As shown in **Table 3**, the MIL tracker has good performances on the sequence $Twinnings$ and the last four sequences, and is not good enough on the other sequences. The IVT tracker works well only on the sequence $Surfing$. Although the center location error of the IVT tracker is quite small on the sequence $DavidIndoor$, the size of the tracking rectangle of the IVT is also decreased to be quite small. Based on the definition of failure frame at the beginning of Section 3.2, if the size of the tracking rectangle is much smaller than the target's size, the tracking result is regarded to be failed and not acceptable. Similarly to the MIL and IVT trackers, the LiAPG, LOT and ORIA trackers work well only on few sequences, which has been shown in **Tables 2** and **3**.

For the proposed AI-ELM tracker, the mean values are all lower than 15, and the standard deviations are all lower than 12. For the video sequences $Cliffbar$, $Tiger2$, $Basketball$, and $DavidOutdoor$, the AI-ELM tracker has the best overall performances when considering all six trackers in terms of the maximum value, the mean value, and the standard deviation. For the other six testing sequences, the proposed AI-ELM tracker has comparable performances to that of the best tracking results from the other five trackers. Therefore, we can conclude that our tracker outperforms the other five trackers when using these testing video sequences.

Table 2. Center location error (in pixels) comparison of the proposed AI-ELM tracker and the other five state-of-the-art trackers (1).

Sequence	AI-ELM			L1APG			LOT		
	Max	Mean	Std	Max	Mean	Std	Max	Mean	Std
Cliffbar	17.5	7.4	3.7	125.1	28.6	30.4	136.2	37.0	31.1
DavidIndoor	31.9	14.2	7.5	173.1	44.9	52.5	194.4	62.9	51.7
Tiger2	58.6	7.7	7.8	129.9	35.8	31.4	96.2	43.8	21.7
Twinnings	22.2	10.8	5.5	25.1	9.5	6.4	80.7	18.2	22.3
Basketball	47.8	13.9	11.2	359.0	137.7	114.5	342.6	69.6	101.8
DavidOutdoor	42.0	7.8	4.9	299.2	89.7	107.4	35.1	9.5	5.3
FaceOcc2	28.4	9.5	5.0	47.3	12.9	11.4	54.8	17.4	12.7
PolarBear	27.4	10.9	5.3	42.5	11.4	9.0	28.1	7.8	5.1
Sphere	56.2	14.3	9.4	21.6	9.3	5.0	163.3	60.5	61.0
Surfing	10.2	4.1	2.2	5.2	1.7	0.7	23.3	5.7	3.5

Table 3. Center location error (in pixels) comparison of the proposed AI- ELM tracker and the other five state-of-the-art trackers (2).

Sequence	ORIA			IVT			MIL		
	Max	Mean	Std	Max	Mean	Std	Max	Mean	Std
Cliffbar	97.7	28.7	24.9	141.3	61.8	35.0	39.7	11.1	8.3
DavidIndoor	89.9	25.9	25.6	20.7	6.4	3.3	71.5	32.7	16.3
Tiger2	115.4	42.8	26.4	134.7	65.1	24.7	54.5	10.1	6.9
Twinnings	182.1	54.6	67.6	129.4	41.9	45.9	26.3	10.9	6.7
Basketball	250.4	151.6	58.9	210.7	98.3	50.9	326.2	100.1	87.3
DavidOutdoor	417.7	176.3	132.6	403.7	225.8	130.0	128.5	32.3	39.3
FaceOcc2	18.6	6.4	4.1	173.1	65.0	45.2	36.9	20.1	8.5
PolarBear	52.8	17.9	14.4	59.0	21.1	15.7	35.4	11.6	7.0
Sphere	216.7	112.2	47.6	262.5	135.4	40.7	57.6	40.3	14.9
Surfing	18.8	4.5	3.7	9.5	2.8	2.0	8.0	2.6	1.6

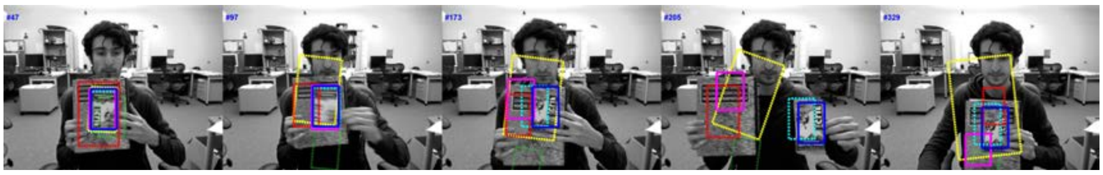
3.3. Qualitative analysis

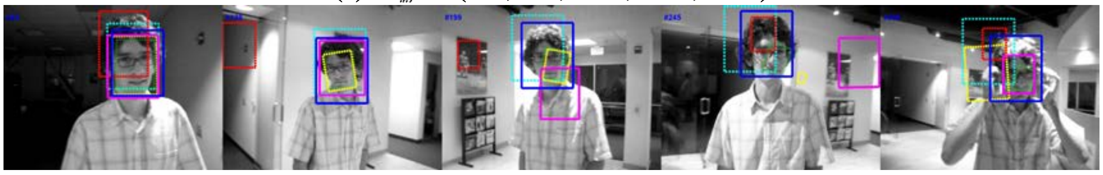
The screenshots of the proposed AI-ELM tracker and of the other five state-of-the-art trackers with these seven test video sequences are shown in **Fig. 4**, **Fig. 5**, and **Fig. 6**. The tracking results of all of the trackers are highlighted with rectangles of different colors and different line styles. For all of the screenshots of the seven test video sequences, the AI-ELM tracker presents rather accurate tracking results without serious drifting problems. The AI-ELM tracker has better processing capacity on these testing sequences when compared to the other five trackers.

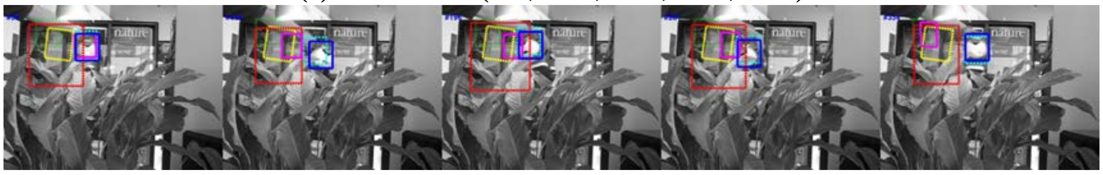
For the *Cliffbar* and *Twinnings* sequences, the target objects suffer from the influence of a texture that is similar to the local background. Moreover, the scale of the targets varied during the video sequences. For the *Cliffbar* sequence, the AI-ELM tracker successfully tracks the target in all five screenshots. The MIL tracker works well at frame #47 and #97, and suffers from heavy drifting problems. The other four trackers are seriously impacted by the background. For the *Twinnings* sequence, the AI-ELM tracker, the MIL tracker, the LOT tracker, and the L1APG tracker can follow the target in the majority of the situations. However, the rectangles of the latter two trackers have different degrees of offsets, and the IVT and ORIA trackers even lose the target at frame #390 and #419.

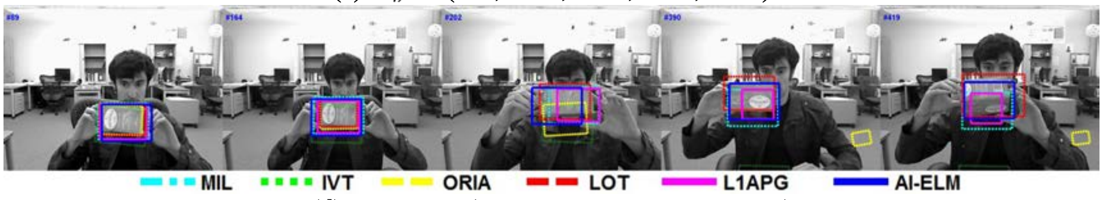
For the $DavidIndoor$, $Tiger2$, $DavidOutdoor$, and $FaceOcc2$ sequences, the sequences are comprised of rotations, scale variations, and heavy occlusion. The LOT and ORIA trackers have terrible performance on the $DavidIndoor$ and $Tiger2$ sequences and suffer from serious scale problems. The IVT tracker always loses the targets in all of these four sequences. As Shown in **Fig. 4**, the IVT tracker has small location error on the sequence $DavidOutdoor$ (#199, #245 and #395), which seems that the tracking accuracy is acceptable. However, the tracking rectangle is changed to be quite small than the object, the tracking result is regarded as failure for this case. The L1APG tracker can follow the targets at the beginning, but becomes ineffective later. The MIL tracker can basically track the targets and with light drifting issues, and the proposed AI-ELM works much better than all these five trackers on these four sequences.

The basketball player that is tracked in the $Basketball$ sequence is easily confused with other players, even with the same uniform. The proposed AI-ELM tracker can deal with these challenges effectively and achieves better performance, compared to the other five trackers for these test video sequences. At the frames #585 and #720, the other five trackers lose the targets with rather large center location errors, and the AI-ELM tracker achieves better performances than the other five trackers in this experiment.


(a) $Cliffbar$ (#47, #97, #173, #205, #329)

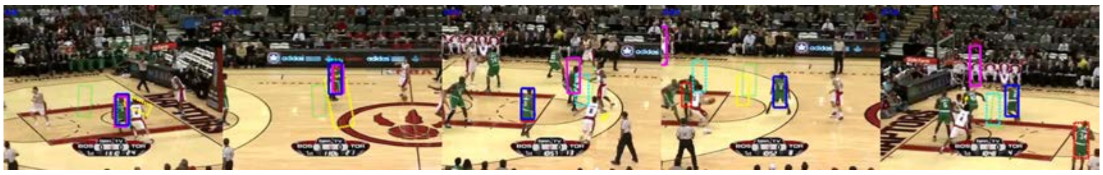

(b) $DavidIndoor$ (#63, #128, #199, #245, #395)

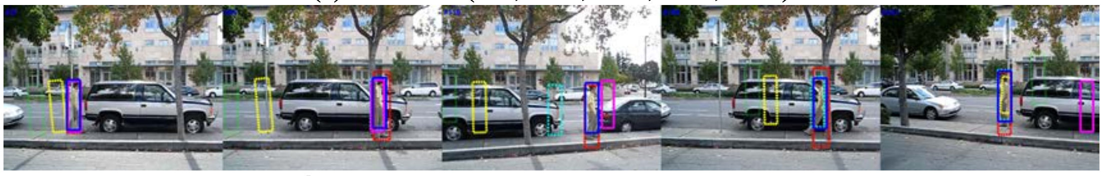

(c) $Tiger2$ (#58, #144, #196, #251, #334)

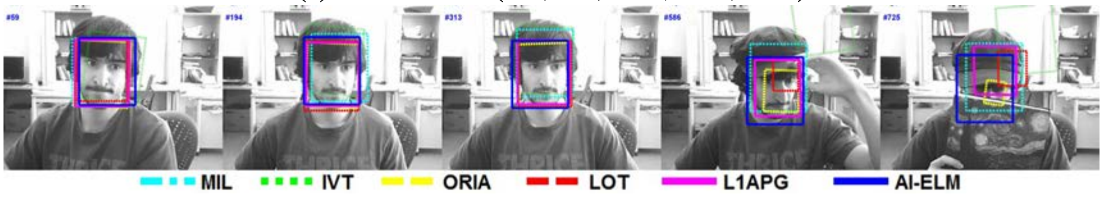


(d) $Twinnings$ (#89, #164, #202, #390, #419)

Fig. 4. Screenshots of tracking results (1). The video sequences are *Cliffbar, DavidIndoor, Tiger2, Twinnings,* from top to bottom. The rectangles of tracking results are best viewed in color and line style.

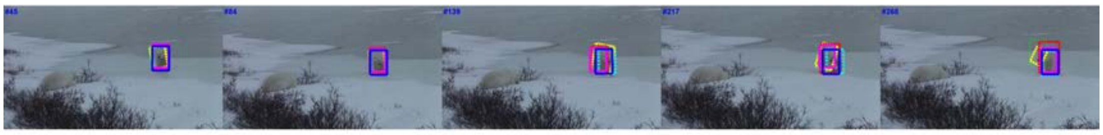

(a) *Basketball* (#46, #191, #450, #585, #720)

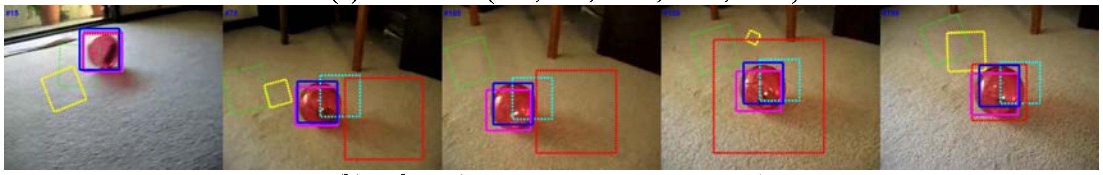

(b) *DavidOutdoor* (#27, #86, #119, #190, #243)

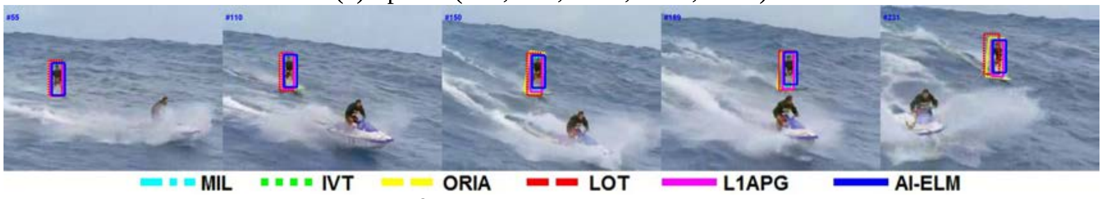


(c) *FaceOcc2* (#59, #194, #313, #586, #725)

Fig. 5. Screenshots of tracking results (2). The video sequences are *Basketball*, *DavidOutdoor*, and *FaceOcc2*, from top to bottom. The rectangles of tracking results are best viewed in color and line style.


(a) *PolarBear* (#45; #84; #139; #217; #268)


(b) Sphere (#15; #78; #105; #139; #190)



(c) *Surfing* (#55; #110; #150; #189; #231)

Fig. 6. Screenshots of tracking results (3). The video sequences are *PolarBear*, *Sphere*, and *Surfing*, from top to bottom. The rectangles of tracking results are best viewed in color and line style.

For sequence *PolarBear*, *Sphere*, and *Surfing*, the proposed AI-ELM tracker always has the second best performances. The L1APG, LOT, and MIL trackers have rather perfect performances on one or two video sequences. However, these three trackers cannot follow the targets well on other testing sequences. In other words, there performances are not robust enough on all of the testing sequences.

For some sequences, such as Tiger2 and FaceOcc2, the proposed AI-ELM tracker has better performances than other trackers. When the tracked target is occluded, the estimated confidence C_t always has low values. Therefore, the parameters of the AI-ELM tracker are not updated for these cases, which are controlled by the learning rate in Eq. (6). The AI-ELM tracker can avoid the undesirable updates of its parameters, and the target can be followed by the tracker when the target clearly appears again. Therefore, the proposed AI-ELM tracker achieves better performances than the other five trackers in our experiments.

4. Conclusion and future work

In this paper, we have introduced a novel visual tracking algorithm that uses an adaptive, incremental extreme learning machine. For the online tracking process, some positive and some negative training samples were collected per frame, and random Haar-like features of these training samples are extracted and normalized to preform the tracking system parameter update. The adaptive, incremental extreme learning machine updates the overall output weights for each frame using an adaptive learning rate, and the overall output weights contain the information from the previous frames, with the contribution of each frame determined by the estimated confidence of the target.

Our experimental results indicate that the proposed visual tracking system is able to manage challenges, such as occlusion and variations in the appearance. When compared to five other state-of-the-art algorithms, the proposed tracking system shows better performance on the test video sequences.

There are several interesting avenues for future work. First, a higher power training sample collection and feature extracting methods can be adopted to improve the effect and the efficiency of the tracking system. Here we have used a heuristic strategy to define the learning rate, and more rational and robust strategies can be added to this tracking system to enhance the tracking performance. Finally, the adaptive incremental extreme learning machine can be implemented for use applications outside of online data processing for visual tracking.

References

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys*, vol. 38, no. 4, 2006. Article (CrossRef Link)

[2] H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song, "Recent advances and trends in visual tracking: A review," *Neurocomputing*, vol. 74, no. 18, pp. 3823-3831, 2011. Article (CrossRef Link)

[3] Y. Su, Q. Zhao, L. Zhao, and D. Gu, "Abrupt motion tracking using a visual saliency embedded particle filter," *Pattern Recognition*, vol. 47, no. 5, pp. 1826-1834, 2014. Article (CrossRef Link)

[4] C. H. Hsia, Y. J. Liou, and J. S. Chiang, "Directional Prediction CamShift algorithm based on Adaptive Search Pattern for moving object tracking," *Journal of Real-Time Image Processing*, DOI 10.1007/s11554-013-0382-x, 2015. Article (CrossRef Link)

[5] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," *Computer Vision-ECCV 2008*, pp. 234-247: Springer, 2008. Article (CrossRef Link)

[6] C. Migniot, and F. Ababsa, "Hybrid 3DC2D human tracking in a top view," *Journal of Real-Time Image Processing*, Vol 11, issue 4, pp. 769-784, 2015. Article (CrossRef Link)

[7] T. Bai, and Y. F. Li, "Robust visual tracking with structured sparse representation appearance model," *Pattern Recognition*, vol. 45, no. 6, pp. 2390-2404, 2012. Article (CrossRef Link)

[8] R. V. Babu, S. Suresh, and A. Makur, "Online adaptive radial basis function networks for robust object tracking," *Computer Vision and Image Understanding*, vol. 114, no. 3, pp. 297-310, 2010.

Article (CrossRef Link)

[9] D. Wang, H. Lu, and M.-H. Yang, "Least Soft-thresold Squares Tracking," in *Proc. of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, Portland, Oregon, USA, 2013. Article (CrossRef Link)

[10] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting," *British Machine Vision Conference*, pp. 47-56, 2006. Article (CrossRef Link)

[11] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 125-141, 2008. Article (CrossRef Link)

[12] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409-1422, 2010. Article (CrossRef Link)

[13] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1619-1632, 2011. Article (CrossRef Link)

[14] S. He, Q. Yang, R. W. Lau, J. Wang, and M.-H. Yang, "Visual tracking via locality sensitive histograms," in *Proc. of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2427-2434, Portland, Oregon, USA, 2013. Article (CrossRef Link)

[15] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, pp. 489-501, 2006. Article (CrossRef Link)

[16] G.-B. Huang, Q.-Y. Zhu and C.-K. Siew, "Extreme Learning Machine: A New Learning Scheme of Feedforward Neural Networks," in *Proc. of the International Joint Conference on Neural Networks (IJCNN 2004)*, Budapest,Hungary, pp. 25-29, Jul. 2004. Article (CrossRef Link)

[17] G. Huang, G.-B. Huang, S. Song, K. You, "Trends in Extreme Learning Machines: A Review," *Neural Networks*, vol. 61, pp. 32-48, 2015. Article (CrossRef Link)

[18] L. L. C. Kasun, H. Zhou, G. -B. Huang, C. M. Vong, "Extreme Learning Machines," *IEEE Intelligent System*, vol. 28, no. 6, pp. 30-59, 2013. Article (CrossRef Link)

[19] G.-B. Huang, L. Chen, "Convex incremental extreme learning machine," *Neurocomputing*, vol. 70, pp.3056-3062, 2007. Article (CrossRef Link)

[20] M.-B. Li, G.-B. Huang, P. Saratchandran, N. Sundararajan, "Fully complex extreme learning machine," *Neurocomputing*, vol. 68, pp. 306C314, 2005. Article (CrossRef Link)

[21] Y. Lan, Y. C. Soh, and G.-B. Huang, "Two-stage extreme learning machine for regression," *Neurocomputing*, vol. 73, pp. 3028C3038, 2010. Article (CrossRef Link)

[22] G. Huang, S. Song, J. N. D. Gupta, C. Wu, "Semi-supervised and Unsupervised Extreme Learning Machines," *IEEE transactions on cybernetics*, vol. 44, no. 12, 2405-2417, 2014. Article (CrossRef Link)

[23] L. L. C. Kasun, H. Zhou, G.-B. Huang, and C. M. Vong, "Representational Learning with Extreme Learning Machine for Big Data," *IEEE Intelligent System*, vol. 28, no. 6, pp. 31-34, 2013. Article (CrossRef Link)

[24] N. Y. Liang, G.-B. Huang, P. Saratchandran, N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Transactions on Neural Networks*, vol. 17, no. 6, pp. 1411-1423, 2006. Article (CrossRef Link)

[25] R. Collins, Y. Liu, M. Leordeanu, "Online selection of discriminative tracking features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, pp. 1631-1643, 2005. Article (CrossRef Link)

[26] C. R. Rao and S. K. Mitra, "Generalized inverse of matrices and its applications," *New York: Wiley*, pp. 601-620, 1971. Article (CrossRef Link)

[27] C. Bao, Y. Wu, H. Ling, and H. Ji, "Real time robust L1 tracker using accelerated proximal gradient approach," in *Proc. of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence*, RI, USA, 2012. Article (CrossRef Link)

[28] S. Oron, A. Bar-Hillel, D. Levi, and S. Avidan, "Locally orderless tracking," in *Proc. of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence*, RI, USA, 2012. Article (CrossRef Link)

[29] Y. Wu, B. Shen, and H. Ling, “Online robust image alignment via iterative convex optimization,” in *Proc. of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence,* RI, USA, 2012. Article (CrossRef Link)
[30] Y. Wu, J. Lim, and M.-H. Yang, “Online object tracking: A benchmark,” in *Proc. of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2411-2418, Portland, Oregon, USA, 2013. Article (CrossRef Link)
[31] M. Kristan, R. Pugfelder, A. Leonardis, A., et al., “The visual object tracking vot2014 challenge results,” in *Proc. of the Computer Vision-ECCV 2014Workshops*, pp. 191-217, Springer, Zurich, Switzerland, 2014. Article (CrossRef Link)

Zhihui Wang received the B.Sc. degree in applied mathematics from Qufu Normal University, China, M.Sc. degree in applied mathematics from China Jiliang University, and the Ph.D. degree in electronics and information engineering from Chonbuk National University, Korea. His research interests include artificial intelligence, pattern recognition, and visual tracking. Email: zhihuiwangjl@gmail.com

Sook Yoon received the B.S., M.S., and Ph.D. degrees in engineering from Chonbuk National University, Jeonbuk, Korea, in 1993, 1995, and 2003, respectively. Until June 2006, she conducted her postdoctoral research work in EECS at the University of California, Berkeley. She is presently an associate professor at Department of Multimedia Engineering, Mokpo National University, Jeonnam, Korea. Her current research interests include image processing, pattern recognition, machine learning, and multimedia computing. Email: syoon@mokpo.ac.kr

Dong Sun Park received the B.S. degree from Korea University, Seoul, Korea, in 1979, and the M.S. and Ph.D. degrees from the University of Missouri, Columbia, in 1984 and 1990, respectively. He is currently a Professor at Division of Electronics Engineering, Chonbuk National University, Jeonbuk, Korea. His current research interests include image processing, pattern recognition, computer vision, artificial intelligence and deep learning. Email: dspark@jbnu.ac.kr