An improved compressive tracking based on pixel-wise learner

Ting Chen\textsuperscript{a,b,*}, Hichem Sahli\textsuperscript{a,b,c}, Yanning Zhang\textsuperscript{a}, Tao Yang\textsuperscript{a,d}

\textsuperscript{a}School of Computer Science and Engineering, Northwestern Polytechnical University, Xi’an 710072, China
\textsuperscript{b}Dept. Electronics and Informatics, AVSP Lab, Vrije Universiteit Brussel, Brussels, Belgium
\textsuperscript{c}Interuniversity Microelectronics Center, Leuven, Belgium
\textsuperscript{d}Research & Development Institute of Northwestern Polytechnical University in Shenzhen, China

Abstract. This work extends upon state-of-the-art multi-scale tracking based on compressive sensing (CT) by increasing the overall tracking accuracy. A pixel-wise classification stage is incorporated in CT-based tracker to obtain relatively stable appearance model, by distinguishing object pixels from the background. Additionally, we identify potential distracting regions which are used in a feedback strategy to handle occlusion and avoid drifting toward nearby regions with similar appearances. We evaluate our approach on several benchmark datasets to demonstrate its effectiveness with respect to state-of-the-art tracking algorithms.

Keywords: Compressive tracking, Random projection, Pixel-wise learner, Feedback strategy.

*Ting Chen, chentingnwpu@mail.nwpu.edu.cn

1 Introduction

Visual object tracking is a fundamental task in computer vision, and it has a wide variety of applications, such as video surveillance, motion analysis, robot perception and human computer interaction\textsuperscript{1,2}. Although visual object tracking has been intensively investigated in the past decade, it still a challenging task due to object’s appearance variations over time due to occlusions, illumination variation, and cluttered environments\textsuperscript{3}.

Object tracking algorithms are largely based on good object appearance models\textsuperscript{4-7}. Tracking can be formulated in two different ways: generative and discriminative. Generative tracking learns an appearance model to represent the target in a particular feature space, and then use it to search for the image region with the best matching score\textsuperscript{8}. Such models are typically based on color representations\textsuperscript{9}, templates\textsuperscript{4}, subspace models\textsuperscript{10}, sparse representations\textsuperscript{5}, etc. These models do not consider background information to model object appearance. Discriminative tracking considers visual tracking as a binary classification problem to define the boundary between an object and the
Examples of such models are: support vector tracker (SVT), multiple instance learning (MIL), a semi-supervised tracker (SemiB), tracking learning detection (TLD), and the struck tracker (STRUCK).

Recently, compressive tracking (CT) algorithms, based on compressive sensing theory, have been proposed. Compressive tracking firstly extracts multi-scale image features, and then discriminates patches through a trained Bayes classifier. Lastly, the classifier is updated through an online learning strategy. In CT, the appearance model is generative as the object can be well represented using features extracted in the compressive domain. It is also discriminative because it uses the extracted features to separate the target from the Background. CT performs well in terms of accuracy, robustness, and speed, however suffer from drifting problems caused by background interference. Indeed CT uses a fixed-size tracking box which leads to target loss when the target changes significantly in scale. Moreover, as the classifier parameters are updated with a constant learning rate, the appearance model could be updated with noisy and potentially misaligned examples when the target is occluded for a certain period.

To alleviate these problems, several extensions to the CT algorithm have been proposed. Teng et al. implemented a feedback mechanism to reduce tracking drift and performed multi-scale tracking via a point-tracking process to correct the final location of the target. Chen et al. proposed an adaptive compressive sensing for appearance modelling using a weighted random projection and Fisher discriminant analysis to evaluate each random feature. Wu et al. proposed multi-scale tracking based on compressive sensing (MSCT), using a modified adaptive random measurement matrix that is updated on the basis of the initial matrix and a scale value. As feedback strategy, a second order transition model (target velocity model) has been proposed to estimate the current position and scale status.
Despite the good results obtained by these extensions, unstable and non-deterministic tracking results may be caused by the randomness of the used non-adaptive projection. To obtain a relatively stable appearance model, connected components of an object, based on maximally stable extremal regions (MSERs), are used by Gao et al.\textsuperscript{23} to extract image features through sparse random projection. Although this approach demonstrated impressive performance, it does not solve the problem of object’s scale variation and occlusions.

In this paper, in consideration of the aforementioned limitations of the previous CT methods, we augment the multi-scale tracking based on compressive sensing of Wu et al.\textsuperscript{22} to account for the likelihood of projecting, in the image feature space, stable object components. A pixel-wise learner is used to distinguish the object pixels from background pixels. Moreover, a feedback strategy based on the pixel-wise learner is employed to handle occlusions and avoid the drifting toward nearby regions with similar appearance. Numerous experiments using challenging sequences and comparisons to state-of-the-art algorithms indicated that our tracker has an improved performance in terms of precision, stability and adaptation.

The paper is organized as follows, in Section 2 we briefly introduce the CT and MSCT trackers. In Section 3 we present our proposed tracking algorithm. In Section 4 we extensively evaluate our approach to demonstrate its effectiveness compared to a variety state-of-the-art trackers. Finally, Section 5 concludes the paper.

2 Multi-scale tracking based compressive sensing

In this section, we briefly introduce the original compressive tracker (CT)\textsuperscript{16} and the Multi-scale tracking based compressive sensing (MSCT)\textsuperscript{22}, which are the base of our proposed tracking scheme.
2.1 The main components of the CT algorithm

The main components of the compressive tracking algorithm are (1) naïve Bayes classifier update, and (2) target detection:\(^{16}\):

(1) **Naïve Bayes classifier update:** CT samples some positive samples, \(D^+\), near the current target location, \(l_t^*\), and negative samples, \(D^-\) far away from the object center:

\[
D^+ = \{ z | \| l(z) - l_t^* \| < \alpha \},
\]

\[
D^- = \{ z | \beta < \| l(z) - l_t^* \| < \zeta \},
\]

where \(\alpha < \beta < \zeta\) are search radiusses. To represent the sample \(z \in \mathbb{R}^{w \times h}\), CT uses a set of rectangle features and extracts the features with low dimensionality using a very sparse measurement matrix \(R\):

\[
v = Ru
\]

with \(u \in \mathbb{R}^m\) \((m = (w \times h)^2)\) the high-dimensional image feature, formed by concatenating the convolved target images (represented as column vectors) with rectangle filters; \(v \in \mathbb{R}^n\) the lower-dimensional compressive features with \(n \ll m\). Each element \(v_i\) in the low-dimensional feature \(v\) is a linear combination of spatially distributed rectangle features at different scales as illustrated in Figure 1.

\[v_i = \sum_j r_{ij} x_j\]

**Fig 1** Graphical representation of compressing a high-dimensional vector \(x\) to a low-dimensional vector \(v\)^{16}. 

---

\(^{16}\) Babenko et al. [10] adopt the generalized Haar-like features for object detection with demonstrated success [10], [42], [43]. There often exists a large set of Haar-like features. To represent the sample \(z\) efficiently, a small subset of Haar-like features are compressively sensed with a very sparse measurement matrix \(R\).
The elements of $R$ satisfy:

$$r_{ij} = \sqrt{s} \times \begin{cases} 
1 & \text{with probability } \frac{1}{2s} \\
0 & \text{with probability } 1 - \frac{1}{s} \\
-1 & \text{with probability } \frac{1}{2s}
\end{cases} \quad (3)$$

with $s = n/2$. In the original CT$^{16}$, the random matrix $R$ is computed only once off-line and remains fixed throughout the tracking process. The computational load is very light, it requires to store the nonzero entries in $R$ and the positions $(c, r)$ and sizes $(w, h)$ of rectangle filters in an input image corresponding to the nonzero entries in each row of $R$ (see Figure 1). Then, $v$ is computed by using $R$ to sparsely measure the rectangular features which are computed using the integral image method. See$^{16}$ for the detailed implementation.

A simple Bayesian model is used to construct a classifier based on the positive and negative sample features. The compressive sensing algorithm assumes that all the lower-dimensional samples of the target are independent of each other:

$$H(v) = \log \left( \frac{\prod_{k=1}^{n} p(v_k|y = 1)p(y = 1)}{\prod_{k=1}^{n} p(v_k|y = 0)p(y = 0)} \right) = \sum_{k=1}^{n} \log \left( \frac{p(v_k|y = 1)}{p(v_k|y = 0)} \right). \quad (4)$$

The parameters of the naïve Bayes classifier are incrementally updated according to:

$$\mu_k^1 \leftarrow \lambda \mu_k^1 + (1 - \lambda) \mu_k^1$$
$$\sigma_k^1 \leftarrow \sqrt{\lambda (\sigma_k^1)^2 + (1 - \lambda)(\sigma_1^1)^2 + \lambda(1 - \lambda)(\mu_k^1 - \mu_1^1)^2} \quad (5)$$

where $(\sigma^1, \sigma_k^1, \mu^1, \mu_k^1)$ are the four parameters of the classifier’s Gaussian conditional distribu-
tion, and $\lambda > 0$ is the update rate.

(2) Target detection: The candidate region corresponding to the maximum $H(v)$ is regarded as the tracking target location:

$$l_t^* = l(\arg \max_v H(v))$$  \hspace{1cm} (6)

The overall performance, in terms of both speed and tracking accuracy, of the CT algorithm was significantly improved by the fast compressive tracking (FCT) presented in\textsuperscript{17}. While the CT samples in a fixed rectangular region in single pixel steps, the FCT improves upon this by introducing a coarse to fine sampling method.

2.2 MSCT Tracker

In the above described compressive tracking, the random matrix $R$ is computed once and remain fixed during tracking. To deal with the scale variation of the tracking box Wu et al.\textsuperscript{22} proposed a multi-scale tracking based compressive sensing (MSCT) in which the tracking box is changing alongside the target’s size, hence the random matrix $R$ is also changed accordingly. In their approach Eq.(3) is used to initialize the random matrix $R_0$ in the first frame. The modified random matrix , $R_s$, is constructed as follows: a relevant nonzero element $r_0$ in $R_0$ with a rectangle of $(c, r, w, h)$, will become $(\text{round}(s \cdot c), \text{round}(s \cdot r), \text{round}(s \cdot w), \text{round}(s \cdot h))$ in $R_s$, for scale $s$. Here, $(c, r)$ represent the center of the rectangle area, and $(w, h)$ its width and height, respectively.

The MSCT algorithm of Wu et al.\textsuperscript{22} combines the above described modified random projection matrix $R_s$ with a bootstrap filter framework\textsuperscript{24}, in which the velocity information of the moving target, both in terms of position and scale, is used in the sampling process of candidate samples for target detection. The state vector of the bootstrap filter is defined as $x = (x_p, y_p, s_p)$, with $(x_p, y_p)$
denoting the location of the image patch, and $s_p$ its scale (in both $x$ and $y$ directions). The temporal evolution of the state is modeled as a second-order autoregressive model:

$$x_t(i) = 2x_{t-1}(i) - x_{t-2}(i) + \omega_t$$  \hspace{1cm} \text{(7)}

with $x_t(i)$ the state vector of the $i$-th particle at time $t$, and $\omega_t$ a white noise.

The observation model of the bootstrap filter is based on the naïve Bayesian classifier response (Eq.(4)) for estimating the importance weight and to re-sample the particles:

$$p(z_t|x_t(i)) \propto \exp(H(v^i))$$  \hspace{1cm} \text{(8)}

with $v^i$ obtained by the adaptive random matrix $R_s$, and $H(v^i)$ represents the classifier response of the $i$-th particle, as given by Eq.(4). The weight of each particle is obtained by:

$$q(i) = \frac{p(z_t|x_t(i))}{\sum_{j=1}^{N_s} p(z_t|x_t(j))}$$  \hspace{1cm} \text{(9)}

with $N_s$ the number of particles. Compared to the original CT tracking, which keeps the samples with the maximum classifier response as base location for sampling in the next frame, in the MSCT algorithm the re-sampled particles, of the bootstrap filter, will contribute together to find the most possible samples in the next frame. Implementation details of the bootstrap filter framework can be found in\textsuperscript{22}.

To account for illumination variations and occlusion, Wu \textit{et al.}\textsuperscript{22} implemented a feedback strategy by adjusting the learning rate in Eq.(5). In their approach, for each frame $k$, they evaluate the maximum classifier response, $C_k$, among all particles and compare it to a high and low thresh-
olds along with a tolerance of the $C_k$’s decrease, $\Delta C_k$. Based on the values of $C_k$ and $\Delta C_k$, the authors defined rules to decide whether to update the classifier with $\lambda_0$ (a constant learning rate set at initialization), or not update the classifier at frame $k$.

In our proposed tracking we adopt Wu’s et al.\textsuperscript{22} strategy of updating the random matrix based on the object’s current scale value, and use the bootstrap filter framework to follow object variations. However, we augment the model by using a pixel-wise learner to distinguish object pixels from background pixels to handle occlusions and avoid drifting toward nearby regions with similar appearance.

3 Proposed algorithm

The main components of our tracking algorithm are depicted in Figure 2. We base our tracking algorithm on two primary requirements. First, considering that the CT trackers use pixels confined in rectangular areas to select the samples for estimating the rectangle features and use them for modeling the object’s appearance, distinguishing between object pixels and background is useful to realize adaptive appearance model. Second, to reduce the risk of drifting towards regions with
similar appearance at a future frames, distracting regions should be detected and not used for model update. More precisely, to obtain a relatively stable appearance model, we propose projecting stable components of image features into data-independent random basis based on a pixel-wise object likelihood. To further reduce the risk of drifting, we propose a distractor-aware feedback strategy to handle the appearance variations.

3.1 Stable feature projection for adaptive appearance model

From Eq.(2) and Eq.(3), we observe that unstable tracking results come from both the spars random projection matrix $R$ and the high-dimensional image feature representation $u$. If $R$ is fixed, and $u$ is represented by a stable high image feature space, then a relatively more stable random projection feature $v$ can be obtained to model the target appearance. Based on this analysis, Gao et al. proposed applying maximally stable extremal regions (MSERs) to the feature description to obtain stable image regions. In their model, with the fusion of random projection and MSERs of the image feature space, the target appearance has been modeled as $v_i = \sum_i r_{ij} m_j$, with $r_{ij}$ the elements of the random matrix $R$, and $m_j$ the location and size of a MSER that is matched with an inscribed rectangle. The vector $v$ models the target appearance with random basis, and the feature vector $m = [m_1, \cdots, m_n]^T$ bridges the gap between the randomness and the stability of non-adaptive projection.

Different from, in our approach, inspired by the work of Possegger et al., we propose using the pixel-wise object likelihood to obtain stable object pixels to estimate $v$. In our implementation we use the modified random matrix $R_s$ as defined in Section 2.2. Let $r_{kl}$ a non-zero element in $R_s$ and its associated rectangle filter $R_f = (c, r, w, h)$ in the image. We define an importance factor of $R_f$ based on the percentage of object pixels contained within $R_f$, and use this importance factor as...
weight for \( r_{kl} \):

\[
\bar{r}_{kl} = \bar{\omega}_{kl} \times r_{kl} \tag{10}
\]

\[
\bar{\omega}_{kl} = \frac{\sum_{p \in R_f} P(p \in \mathcal{O}|b_p)}{w \times h} \tag{11}
\]

with \( r_{kl} \) as defined in Eq.(3), and \( P(p \in \mathcal{O}|b_p) \) the object likelihood at location \( p \), defined as in\(^9\):

\[
P(p \in \mathcal{O}|b_p) = \begin{cases} 
\frac{\mathcal{H}_I(b_p)}{\mathcal{H}_O(b_p)+\mathcal{H}_S(b_p)} & \text{if } I(p) \in I(O \cup S) \\
0.5 & \text{otherwise}
\end{cases} \tag{12}
\]

with \( \mathcal{O} \) the object region, estimated from the first frame using the initial bounding box around the target object, and \( S \) its surrounding region, \( \mathcal{H}_X(b_p) \) denotes the \( b \)-th bin of the non-normalised histogram \( \mathcal{H}_X \), computed over the region \( X \), and \( b_p \) is the bin \( b \) assigned to the color component of \( I(p) \). This model allows distinguishing object and background pixels. To adapt the representation to object appearance and illumination changes, the model is updated using a linear interpolation with a learning rate \( \eta \):

\[
P_{1:t}(p \in \mathcal{O}|b_p) = \eta P(p \in \mathcal{O}|b_p) + (1 - \eta) P_{1:t-1}(p \in \mathcal{O}|b_p). \tag{13}
\]

More details on the estimation of \( P(p \in \mathcal{O}|b_p) \) and object segmentation via thresholding of \( P(p \in \mathcal{O}|b_p) \) can be found in\(^9\).

Although, using Eq.(10), the random matrix changes frequently, the additional computation load is small because the nonzero entries are very sparse. The process is illustrated in Figure 3. We illustrate in Figure 4(c) the weighting of the rectangle filters \( R_f = (c, r, w, h) \) associated to
non-zero elements of the random matrix, and compare them to the rectangle filters used in the CT tracker of\textsuperscript{16} in Figure 4(a), and the ones used by Gao \textit{et al.}\textsuperscript{23} according to the MSERs in Figure 4(b). As it can be seen, some rectangle filters used in CT are background areas, hence the classifier may become inaccurate due to non-object features. Using MSERs the number of rectangle filters is reduced, however still some features do not belong to the object. In our approach using the pixel-wise learner, we can 'segment' the object and hence give more weight to pixels belonging to the object, compared to background pixels, for estimating the object appearance.

**Fig 3** The online features selection mechanism based on the weighted random matrix.

**Fig 4** Visual representation of feature projection by random matrix. (a) Rectangle filters extracted by CT. (b) Rectangle filters using MSERs. (c) Weighted rectangle filters using the proposed approach; (d) The object likelihood (object segmentation). The green rectangles indicate features with high weight, while the red rectangles indicate features with low weight.
3.2 Target location based on the bootstrap filter framework

In our implementation, we adopt the the bootstrap filter of\textsuperscript{22}, described in Section 2. In our framework we define the state vector of the bootstrap filter as $x = (l_p, w_p, h_p, s_p)$, where $l_p$ represents the location of the image patch, $w_p$ and $h_p$ its width and height, respectively, and $s_p$ the scale in both the $x$ and $y$ directions. For each frame $t$, we consider $N_s$ particles $\{x_t(i) : i = 1, \cdots, N_s\}$, with importance weights, $q(i)$ given by Eq.(9). The particle $x_t(K)$ which obtains the maximum weight $q(K)$ is selected for defining the new object location $l_t^*$ at frame $t$. As in\textsuperscript{22}, in order to avoid degeneracy, the prior samples are re-sampled to generate an un-weighted particle set according to their importance weights.

Particles with high weights exhibit similar appearance as the target, and hence are considered as distractors to be discarded when sampling for object appearance model update. We define the set of distractors as:

$$D = \{x_t(d) = (l_p^d, w_p^d, h_p^d, s_p^d) | q(d) \geq \kappa q(K); d \in [1, N_s], d \neq K\}$$

with $\kappa \in [0, 1]$.

3.3 Observation model with feedback strategy

In the original compressive tracking the classifier parameters are updated with a constant rate, $\lambda$, as indicated in Eq.(5), which can produce tracking drift if the target is occluded for a long period, or it undertakes illumination changes. To prevent this, in our tracker, a feedback strategy is proposed to evaluate the current state of the object to (1) adapt the learning rate of Eq.(5), and (2) select effective positive and negative samples to update the appearance model.
The learning rate is set according to the target’s changes in scale. Let $\varphi_t = s_t/s_{t-1}$, the ratio of the target’s scale at frame $t$ and frame $t-1$ (at each frame $k$ the target scale $s_k$ is provided by the particle with high weight as indicated in Section 3.2), and $\varrho_t = |\varphi_t - \frac{1}{t-1} \sum_{i=2}^{t-1} \varphi_i|$. We also define two thresholds, $\tau_h$ and $\tau_l$, and a learning rate $\lambda_0$ set at the initial frame. The learning rate at frame $t$ is set as:

$$
\lambda = \begin{cases} 
\lambda_0, & \text{if } \varrho_t < \tau_l \\
\frac{\tau_h - \varrho_t}{\tau_h} \cdot \lambda_0, & \text{if } \tau_l < \varrho_t < \tau_h \\
0, & \text{if } \varrho_t > \tau_h 
\end{cases}
$$

(15)

For the sampling of the positive $D^+$ and negative $D^-$ sample sets, we make use of the detected distractor areas $D_d$ of Section 3.2 and define:

$$D^+ = \{z | ||l(z) - l^*_t|| < \alpha\},$$

$$D^- = D^{\beta,\zeta} \setminus D,$$

$$D^{\beta,\zeta} = \{z | \beta < ||l(z) - l^*_t|| < \zeta\},$$

(16)

with $\alpha < \beta < \zeta$ are search radiuses set at the initialization, and $D$ as defined in Eq.(14). Based on this search strategy, samples which are very similarly to the target samples are avoided when updating the classifier model, as illustrated in Figure 5.

3.4 Analysis and discussion

Our proposed tracker is different from the recent ones based on random matrix projection. The most significant difference is the proposed pixel-wise learner to obtain stable object pixels used with the random matrix. Different from 23 which constructs stable object feature template at the
Fig 5 Search areas for positive and negative samples. (a) Positive samples defined by \( \alpha \). (b) Negative samples defined by \( \beta \), and \( \zeta \), excluding a distractor illustrated in blue.

First frame, our proposed algorithm adaptively updates the stable object pixels according to the object’s appearance and deformation. Another significant difference lies in the fact that our approach employs a feedback strategy by considering the changes in scale of the objects to estimate the learning rate for updating the observation model. This strategy is simpler than the classifiers updating schemes of^{22} and^{25}. The main steps of our proposed algorithm are detailed in Algorithm 1.

**Algorithm 1** Sketch of the proposed PLCT algorithm

**Input:** The \( t \)-th image frame. The object is annotated in the first frame, \( x_1 = (l_1, s_1) \), with location and scale.

**Output:** Tracking location \( l_t \) and new samples \( \{ x_t(i) | i = 1, \ldots, N^s \} \).

1: if \( t=0 \) then First frame
2: Initialize the pixel-wise learner to get stable components in the tracking box using Eq.(12).
3: Initialize the random measure matrix \( R_0 \) using Eq.(3) and Eq.(10).
4: Sample the positive and negative patches by Eq.(16).
5: Compute positive and negative features by Eq.(2) to initialize the classifier using Eq.(4).
6: Initialize \( N^s \) particles \( \{ x_1(i) | i = 1, \ldots, N^s \} \), with importance weights, \( q_1(i) \).
7: else
8: Predict new particles \( \{ x_t(i) | i = 1, \ldots, N^s \} \) using Eq.(7).
9: Compute each feature vector of \( x_t(i) \) by Eq.(2) and classify each feature vector by Eq.(8).
10: Re-sample each particle according to the weight \( q_t(i) \) of Eq.(9).
11: Locate the object position \( l_t^* \) and detect the distractors \( D \).
12: Update the pixel-wise learner using the new object location \( l_t^* \) by Eq.(13), and select the new stable components by Eq.(12) to update the random matrix \( R_t \) using Eq.(10).
13: Compute the feature vectors from the new sampled positive and negative patches.
14: Update the classifier incrementally using the learning rate \( \lambda \) as defined in Eq.(15).
15: end if
Table 1 Used benchmark sequences.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Dataset</th>
<th>IV</th>
<th>SV</th>
<th>OCC</th>
<th>RT</th>
<th>FM</th>
<th>BC</th>
<th>LR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basketball</td>
<td>28</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
</tr>
<tr>
<td>Biker</td>
<td>28</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>√</td>
</tr>
<tr>
<td>Bolt</td>
<td>28</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Boy</td>
<td>28</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Couponbook</td>
<td>28</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Davidindoor</td>
<td>28</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>×</td>
</tr>
<tr>
<td>Deer</td>
<td>28</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>×</td>
</tr>
<tr>
<td>Football</td>
<td>28</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>MountainBike</td>
<td>28</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>×</td>
</tr>
<tr>
<td>Occludedface</td>
<td>28</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Shaking</td>
<td>28</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Skating</td>
<td>28</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Subway</td>
<td>28</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Sylvester</td>
<td>28</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>×</td>
</tr>
<tr>
<td>Tiger2</td>
<td>28</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>EgTest01</td>
<td>27</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>×</td>
</tr>
<tr>
<td>EgTest02</td>
<td>27</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>EgTest03</td>
<td>27</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>EgTest04</td>
<td>27</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Hockey</td>
<td>26</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
</tbody>
</table>

4 Experiment

In this section, the proposed algorithm, denoted hereafter as PLCT, is compared to 3 CT-based algorithms and other 8 state-of-the-art tracking methods using 20 challenging sequences obtained from 1,2,3. Table 1 lists the different sequences along with their main challenging situations, including Illumination Variation (IV), Scale Variation (SV), Occlusion (OCC), Rotation (RT), Fast Motion (FM), Background Clutter (BC) and Low resolution (LR).

4.1 Experimental setup

In our experiments, the size of the random matrix $R$ is set as $n = 200$. For the pixel-wise learner, we adopt the same parameters setting of 9 with a learning rate $\eta = 0.8$. For generating the positive
samples of the naïve Bayes classifier model, the search radius is set as $\alpha = 4$, to generated 45 positive samples. For the negative samples, the inner radius is set to $\beta = 8$ and the outer radius $\zeta = 30$, and we randomly select 50 negative samples. The initial learning rate for updating the classifier model was set as $\lambda_0 = 0.9$ and the pre-defined thresholds defined as $\tau_h = 0.8$ and $\tau_l = 0.5$. The number of particles (samples) is set to $N^s = 200$ and the initial scale of each particle is set to $s = 1$. To define the set of distractors $D$, we set $\kappa = 0.5$ in Eq. (14).

4.2 Evaluation criteria

For the quantitative evaluation of the tracking performance, we use the success rate score ($SR$) and the average center location error ($CLE$)\(^{14}\). $SR$ is defined as $SR = \frac{\text{Area}(B_T \cap B_{GT})}{\text{Area}(B_T \cup B_{GT})}$, with $B_T$ the tracking bounding box and $B_{GT}$ the ground truth bounding box. The tracking is successful if $SR$ is larger than a threshold $T_{SR}$; here we defined $T_{SR} = 0.5$. $CLE$ is defined as $CLE = ||l^* - L_{GT}||$, with $l^*$ the center (location) of the tracking bounding box, and $L_{GT}$ the center location of the ground truth bounding box.

4.3 Comparison to CT-based approaches

In the following we compare the proposed PLCT algorithm to three CT-based tracking approaches, namely the original CT\(^{16}\), our previous dynamic compressive tracker (DCT)\(^{29}\) and the multi-scale tracker based on compressive sensing (MSCT)\(^{22}\). We make use of the code provided by the authors of\(^{16,22}\), and adopt the same parameter setting as used by the authors in their original publication.

4.3.1 Overall performance

The success plot, based on $SR$ metric, and the precision plots, based on the $CLE$ metric, are given in Figure 6 for the considered CT-based trackers with 4 selected sequences Basketball, EgTest01,
4.3 Comparison to CT-based approaches

In the following we compare the proposed PLCT algorithm to three CT-based tracking approaches, namely the original CT, our previous dynamic compressive tracker (DCT) and the multi-scale tracker based on compressive sensing (MSCT). We made use of the code provided by the authors of the original CT and MSCT. The same parameter setting as used by the authors in their original publication was adopted.

4.3.1 Overall performance

The precision plots, based on the CLE metric, and the success plot, based on SR metric, are shown in Figure 6 for the considered CT-based trackers. The results show that our proposed PLCT algorithm achieved the best performance. Moreover, it produces relatively more stable result compared to the other three CT-based trackers, according to both the precision and the success rate criterions. For the reference threshold of 20 pixels from the center location error, the overall precision of PLCT is the highest among all trackers. For the overlap success rate, both MSCT and PLCT produce the highest rate thanks to the multi-scale random matrix, which can adaptively change the scale of the tracking bounding box.

4.3.2 The benefits of the pixel-wise learner and feedback strategy

Figure 7 shows snapshots of tracking results of the considered trackers with the 4 selected sequences. As can be observed, due to the fixed tracking box, both CT and DCT trackers lose the targets under partial occlusion, background clutter and low-grain background. In Figure 7(a), the players are playing within a yellow playground. The CT tracker drifts to the playground and then selects a distractor as tracked target after frame #283. The DCT tracker lose the target after frame #649 because the target is occluded by a distractor, while the MSCT and PLCT perform well with...
The help of the used transition model and feedback strategy. The tracking box of the MSCT is smaller than the one of PLCT compared to the ground truth after frame #736. This is due to the fact that MSCT zooms out the tracking box to select more stable features from the target, while the pixel-wise learner strategy of PLCT helps keeping the target size. The same situation can be observed in the Hockey sequence of Figure 7(d), where the target is occluded and cluttered by the background and other distractors.

The cars in the EgTest01 and EgTest02 sequences (shown in Figure 7(b) and Figure 7(c), respectively) are very small and their colors are very similar to the background. In Figure 7(b), the CT tracker drifts the tracking box to the shadow, this is mainly due to the fixed size of the random matrix. This also occurred in the EgTest02 sequence at frame #300 (Figure 7(c)). Since the CT and DCT trackers update the classifier model at a fixed learning rate, they fail tracking the right object after frame #40 because of the background clutter and distractors. However, considering the scale
and the feedback strategy of both MSCT and PLCT the tracking are more efficient.

4.4 Comparison to State-of-art Tracking Approaches

In addition to the above comparison to CT-based trackers, in the following the proposed PLCT algorithm is compared to eight state-of-the-art tracking methods: the fragment tracker (FRAG)\(^4\), the incremental visual tracker (IVT)\(^9\), the $\ell_1$ tracker (L1T)\(^5\), the MIL\(^14\), the online AdaBoost tracker (OAB)\(^30\), the SemiB\(^15\) tracker, the STRUCK tracker\(^7\) and TLD\(^6\). For the considered trackers, we use the publicly available source code or the one provided directly by the authors. For a fair comparison, the same parameter setting as used by the authors in their original publication is adopted. All the trackers are implemented in MATLAB and run on an Intel Core 2.40 GHz CPU with 32GB RAM. Tables 2 and Tables 3 list the results achieve in terms of $SR$ and $CLE$, respectively. In Table 2, we also give the average processing time (over all the sequences) in terms of frames per second (FPS). As it can be observed, our proposed PLCT tracker is able to accurately track the right targets in all considered sequences; moreover, it has similar accuracy to the best and second-best performing methods.

4.4.1 Pose and illumination change

In the *Biker* sequence shown in Figure 8(a), the target undergoes abrupt motion and pose variation. In the *David indoor* sequence shown in Figure 8(b), the target appearance changes, due to illumination and pose variation when the person walks out of the dark room. In the *Mountain bike* sequence (Figure 8(c)), target undergoes rotation and pose variation. In the *Shaking* sequence shown in Figure 8(d), the pose variation of the target is rapid. In Figure 8(e), the target in the *Skating* sequence undergoes occlusion at frame #165, shape deformation at frame #280 and severe illumi-
nation change at frame #383. In the *Sylvester* sequence, shown in Figure 8(f), the target undergoes a large number of pose and illumination changes. The IVT and L1T trackers, which are based on a generative model, are less effective in addressing the target’s appearance change. The TLD approach works well as it maintains a detector that uses Haar-like features during tracking. The MIL algorithm uses multiple samples to construct an observation model, which accounts for the target’s

Fig 8 Snapshots of tracking results under pose and illumination changes.
appearance variations, hence its tracking performance is good. The STRUCK method drifts away from the target when due to the rapidly pose variation. For these five sequences, the compressive sensing trackers are robust to pose and illumination changes, because the object appearance could be effectively modelled by random projections. Compared to the CT and DCT trackers, the MSCT and PLCT trackers improve the performance based on the scale adaptive tracking box.

4.4.2 Background clutter

In the Bolt sequence, several objects appear in the same scene with rapid appearance change due to shape deformation and fast motion, as shown in 9(a). The target in the Coupon book sequence undergoes significant appearance change at frame #60, and then another book appears in the scene, as shown in 9(b). In the Deer sequence shown in Figure 9(c), the deer undergoes fast motion and

![Tracking results of the Bolt sequence](image1)

![Tracking results of the Coupon book sequence](image2)

![Tracking results of the Deer sequence](image3)

![Tracking results of the Football sequence](image4)

**Fig 9** Snapshots of tracking results under background clutter.
background clutter. The deer is jumping in the river together with other deers. All the deers are similar and the tracked area has similar texture as the background. The target in the Football1 sequence undergoes background clutter (shown in the 9(d)), it is difficult for most trackers to effectively distinguish the target from the background clutters. The L1T and IVT trackers suffer from the drifting problem due to the fact that these trackers ignore the discriminative information between the target and the background. The SemiB tracker lose the target because it is relied on the casual information from the first frame while lose other useful information. The MIL, CT, DCT and MSCT adopt a fixed search rate to select the positive and negative samples to update the classifier, their discrimination ability are thus highly decreased when the negative samples include distractor samples. However, our proposed PLCT tracker effectively distinguishes the object from its surrounding background clutters because it makes full use of the adaptive appearance modeling thanks to the pixel-wise learner and the discriminative classification information based on the feedback strategy.

4.4.3 Occlusion

In the Basketball sequence shown in Figure 10(a), the target moving on the playground is associated with challenging situations, such as occlusion in frame #283, similar object distraction in frame #463 and background clutter in frame #495. In the Hockey sequence, the target undergoes pose variation, occlusion, and shape deformation, as shown in Figure 10(b). The target in the Occluded face sequence in Figure 10(c) undergoes in-plane pose variation and occlusion. In the Subway sequence shown in Figure 10(d), the target undergoes heavy occlusion at frame #44 and background clutter at frame #84. The TLD lose tracking when the target is occluded by the background, while the tracking box is re-located to the target with the help of the online semi-supervised PN learning
procedure. The classifier models of the OAB, MIL, CT and DCT are updated online without efficient determination of whether or not the target is occluded. The features from the occluded object replace the original target feature when the target is occluded for an extended period. Similar to the MSCT tracker, the classifier parameters in PLCT are adaptively update based on the feedback strategy. Once the performance tracking decline PLCT stops updating the classifier parameters.

Fig 10 Snapshots of tracking results under occlusion.
4.4.4 Low resolution and low contrast

In airborne-based object tracking scenarios, the target’s pose and background change gradually. Figure 11 shows tracking snapshots in airborne-based object tracking scenes, where the moving targets are very similar to the background, as illustrated in Figure 11(a)-(d). In the EgTest01 sequence, the CT tracker drifts the tracking box to the shadow due to illumination changes in frame #37. The TLD tracker drifts the tracking box, in frame #144, to another vehicle, which is very similar to the tracked target; the same happens to the the MIL tracker in frame #240. For the EgTest02 sequence, the SemiB, OAB and L1T trackers drift away to other vehicles after frame #405. Most of the trackers lose the target in the EgTest03 sequence due to the color similarity of

![Fig 11 Snapshots of tracking results under low resolution and low contrast.](image-url)
the target and background. In the EgTest04 sequence, the CT and DCT trackers drift the tracking box to the shadow due to inefficient feature selection. However, for all sequences the proposed PLCT tracker, keeps track of the target object.

In summary, the experiments described above show that the proposed PLCT tracker is able to accurately track the right target in all the considered sequences. It extracts discriminative stable object features based on the pixel-wise learner, and updates separately the foreground and background. Moreover, the feedback mechanism used in the proposed PLCT is able to avoid the drifting under short-term occlusion.

Table 2 Success Rate (SR) score(%). The best and the second best performing methods are shown in Bold and Italic, respectively.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Frag</th>
<th>IVT</th>
<th>LIT</th>
<th>MIL</th>
<th>OAB</th>
<th>SemiB</th>
<th>STRUCK</th>
<th>TLD</th>
<th>CT</th>
<th>DCT</th>
<th>MSCT</th>
<th>PLCT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basketball</td>
<td>72.21</td>
<td>33.52</td>
<td>39.15</td>
<td>48.73</td>
<td>17.24</td>
<td>39.06</td>
<td>41.33</td>
<td>58.95</td>
<td>22.46</td>
<td>79.87</td>
<td>80.24</td>
<td>81.68</td>
</tr>
<tr>
<td>Biker</td>
<td>3.00</td>
<td>10.00</td>
<td>3.00</td>
<td>66.00</td>
<td>39.00</td>
<td>39.00</td>
<td>52.00</td>
<td>84.00</td>
<td>75.52</td>
<td>76.59</td>
<td>78.24</td>
<td></td>
</tr>
<tr>
<td>Bolt</td>
<td>41.00</td>
<td>17.00</td>
<td>2.00</td>
<td>92.00</td>
<td>20.00</td>
<td>18.00</td>
<td>78.00</td>
<td>30.00</td>
<td>90.00</td>
<td>89.13</td>
<td>88.49</td>
<td>89.38</td>
</tr>
<tr>
<td>Boy</td>
<td>56.34</td>
<td>36.52</td>
<td>48.85</td>
<td>65.14</td>
<td>77.13</td>
<td>72.52</td>
<td>79.37</td>
<td>80.00</td>
<td>40.09</td>
<td>57.78</td>
<td>62.39</td>
<td>70.33</td>
</tr>
<tr>
<td>Coupon book</td>
<td>26.00</td>
<td>98.00</td>
<td>16.00</td>
<td>98.00</td>
<td>98.00</td>
<td>23.00</td>
<td>98.00</td>
<td>41.00</td>
<td>97.00</td>
<td>98.13</td>
<td>98.26</td>
<td>98.27</td>
</tr>
<tr>
<td>David indoor</td>
<td>8.00</td>
<td>98.00</td>
<td>83.00</td>
<td>71.00</td>
<td>32.00</td>
<td>46.00</td>
<td>93.00</td>
<td>98.00</td>
<td>94.00</td>
<td>96.31</td>
<td>97.73</td>
<td>97.79</td>
</tr>
<tr>
<td>Deer</td>
<td>23.65</td>
<td>29.69</td>
<td>46.98</td>
<td>80.36</td>
<td>45.26</td>
<td>65.23</td>
<td>85.36</td>
<td>79.17</td>
<td>63.59</td>
<td>76.69</td>
<td>70.84</td>
<td>79.29</td>
</tr>
<tr>
<td>EgTest01</td>
<td>73.12</td>
<td>52.75</td>
<td>67.35</td>
<td>76.88</td>
<td>46.39</td>
<td>69.49</td>
<td>70.28</td>
<td>65.46</td>
<td>52.79</td>
<td>73.98</td>
<td>75.94</td>
<td>80.89</td>
</tr>
<tr>
<td>EgTest02</td>
<td>55.34</td>
<td>40.85</td>
<td>75.08</td>
<td>73.48</td>
<td>20.66</td>
<td>68.74</td>
<td>80.67</td>
<td>61.24</td>
<td>52.23</td>
<td>72.47</td>
<td>70.06</td>
<td>76.19</td>
</tr>
<tr>
<td>EgTest03</td>
<td>64.88</td>
<td>44.35</td>
<td>63.74</td>
<td>76.43</td>
<td>50.41</td>
<td>71.57</td>
<td>79.41</td>
<td>72.78</td>
<td>63.47</td>
<td>64.73</td>
<td>70.47</td>
<td>80.41</td>
</tr>
<tr>
<td>EgTest04</td>
<td>75.25</td>
<td>49.47</td>
<td>73.51</td>
<td>68.07</td>
<td>60.68</td>
<td>64.74</td>
<td>80.44</td>
<td>84.67</td>
<td>58.13</td>
<td>66.17</td>
<td>71.83</td>
<td>74.36</td>
</tr>
<tr>
<td>Football</td>
<td>26.00</td>
<td>64.00</td>
<td>35.00</td>
<td>77.00</td>
<td>31.00</td>
<td>17.00</td>
<td>62.00</td>
<td>59.00</td>
<td>74.00</td>
<td>59.98</td>
<td>72.66</td>
<td>74.21</td>
</tr>
<tr>
<td>Hockey</td>
<td>72.37</td>
<td>54.65</td>
<td>66.64</td>
<td>65.51</td>
<td>35.82</td>
<td>58.24</td>
<td>63.84</td>
<td>66.05</td>
<td>38.37</td>
<td>73.53</td>
<td>68.43</td>
<td>75.18</td>
</tr>
<tr>
<td>MountainBike</td>
<td>24.72</td>
<td>57.07</td>
<td>54.95</td>
<td>61.52</td>
<td>30.64</td>
<td>29.72</td>
<td>66.16</td>
<td>70.75</td>
<td>26.45</td>
<td>32.16</td>
<td>64.35</td>
<td>65.18</td>
</tr>
<tr>
<td>Occluded face</td>
<td>54.00</td>
<td>96.00</td>
<td>96.00</td>
<td>97.00</td>
<td>49.00</td>
<td>41.00</td>
<td>97.00</td>
<td>87.00</td>
<td>99.00</td>
<td>98.35</td>
<td>98.49</td>
<td>98.37</td>
</tr>
<tr>
<td>Shaking</td>
<td>25.00</td>
<td>1.00</td>
<td>3.00</td>
<td>83.00</td>
<td>39.00</td>
<td>30.00</td>
<td>48.00</td>
<td>55.00</td>
<td>80.00</td>
<td>78.63</td>
<td>71.07</td>
<td>73.34</td>
</tr>
<tr>
<td>Skating</td>
<td>11.00</td>
<td>8.00</td>
<td>65.00</td>
<td>21.00</td>
<td>68.00</td>
<td>39.00</td>
<td>84.00</td>
<td>37.00</td>
<td>83.00</td>
<td>64.93</td>
<td>73.72</td>
<td>73.58</td>
</tr>
<tr>
<td>Subway</td>
<td>64.14</td>
<td>31.15</td>
<td>49.54</td>
<td>80.75</td>
<td>21.04</td>
<td>59.75</td>
<td>81.62</td>
<td>73.42</td>
<td>51.34</td>
<td>70.77</td>
<td>75.41</td>
<td>81.14</td>
</tr>
<tr>
<td>Sylvester</td>
<td>34.00</td>
<td>45.00</td>
<td>40.00</td>
<td>77.00</td>
<td>65.00</td>
<td>66.00</td>
<td>80.00</td>
<td>89.00</td>
<td>69.00</td>
<td>72.74</td>
<td>62.64</td>
<td>71.42</td>
</tr>
<tr>
<td>Tiger2</td>
<td>12.00</td>
<td>19.00</td>
<td>11.00</td>
<td>44.00</td>
<td>36.00</td>
<td>16.00</td>
<td>62.00</td>
<td>48.00</td>
<td>60.82</td>
<td>86.68</td>
<td>68.23</td>
<td></td>
</tr>
<tr>
<td>Average SR</td>
<td>41.10</td>
<td>44.30</td>
<td>46.99</td>
<td>67.89</td>
<td>45.46</td>
<td>46.70</td>
<td>73.47</td>
<td>64.23</td>
<td>64.35</td>
<td>73.13</td>
<td>73.82</td>
<td>79.37</td>
</tr>
<tr>
<td>Average FPS</td>
<td>3.86</td>
<td>9.33</td>
<td>0.45</td>
<td>10.59</td>
<td>8.61</td>
<td>6.78</td>
<td>5.42</td>
<td>10.63</td>
<td>38.27</td>
<td>12.14</td>
<td>25.69</td>
<td>16.26</td>
</tr>
</tbody>
</table>
Table 3 Average Center Location Errors (in pixels). The best and the second best performing methods are shown in **Bold** and *Italic*, respectively.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Frag</th>
<th>IVT</th>
<th>L1T</th>
<th>MIL</th>
<th>OAB</th>
<th>SemiB</th>
<th>STRUCK</th>
<th>TLD</th>
<th>CT</th>
<th>DCT</th>
<th>MSCT</th>
<th>PLCT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basketball</td>
<td>25.45</td>
<td>87.65</td>
<td>75.13</td>
<td>102.08</td>
<td>114.39</td>
<td>94.98</td>
<td>55.27</td>
<td>17.20</td>
<td>59.71</td>
<td>23.36</td>
<td><strong>16.65</strong></td>
<td><strong>14.77</strong></td>
</tr>
<tr>
<td>Biker</td>
<td>107.00</td>
<td>111.00</td>
<td>89.00</td>
<td>44.00</td>
<td><strong>8.00</strong></td>
<td>227.00</td>
<td>102.00</td>
<td>48.00</td>
<td>286.00</td>
<td>9.00</td>
<td>15.50</td>
<td>16.03</td>
</tr>
<tr>
<td>Bolt</td>
<td>44.00</td>
<td>60.00</td>
<td>261.00</td>
<td><strong>8.00</strong></td>
<td>227.00</td>
<td>102.00</td>
<td>48.00</td>
<td>286.00</td>
<td>9.00</td>
<td>15.50</td>
<td>16.03</td>
<td></td>
</tr>
<tr>
<td>Boy</td>
<td>36.64</td>
<td>91.02</td>
<td>47.46</td>
<td>14.99</td>
<td>23.87</td>
<td>86.33</td>
<td><strong>5.91</strong></td>
<td>16.29</td>
<td>33.03</td>
<td>21.46</td>
<td>20.76</td>
<td><strong>16.22</strong></td>
</tr>
<tr>
<td>Couponbook</td>
<td>62.00</td>
<td><strong>4.00</strong></td>
<td>75.00</td>
<td>9.00</td>
<td>74.00</td>
<td><strong>6.00</strong></td>
<td>81.00</td>
<td>7.00</td>
<td>10.28</td>
<td>13.72</td>
<td>9.67</td>
<td></td>
</tr>
<tr>
<td>Davidindoordoor</td>
<td>73.00</td>
<td><strong>6.00</strong></td>
<td>17.00</td>
<td>19.00</td>
<td>57.00</td>
<td>37.00</td>
<td>64.00</td>
<td><strong>8.00</strong></td>
<td>14.00</td>
<td>14.29</td>
<td>10.34</td>
<td></td>
</tr>
<tr>
<td>Deer</td>
<td>100.00</td>
<td>207.00</td>
<td>122.00</td>
<td>32.00</td>
<td>62.00</td>
<td>26.00</td>
<td>19.00</td>
<td>125.00</td>
<td><strong>16.00</strong></td>
<td>22.34</td>
<td>17.08</td>
<td><strong>15.37</strong></td>
</tr>
<tr>
<td>EgTest01</td>
<td>6.75</td>
<td>8.16</td>
<td>6.64</td>
<td>3.28</td>
<td>5.34</td>
<td>9.19</td>
<td><strong>3.22</strong></td>
<td>34.43</td>
<td>12.29</td>
<td>6.04</td>
<td>4.39</td>
<td><strong>2.19</strong></td>
</tr>
<tr>
<td>EgTest02</td>
<td>19.29</td>
<td>15.39</td>
<td>12.57</td>
<td>14.92</td>
<td>38.34</td>
<td>16.43</td>
<td><strong>4.87</strong></td>
<td>12.32</td>
<td>19.91</td>
<td>6.39</td>
<td>9.02</td>
<td>5.34</td>
</tr>
<tr>
<td>EgTest03</td>
<td>20.29</td>
<td>28.43</td>
<td>39.35</td>
<td>16.39</td>
<td>22.46</td>
<td>21.34</td>
<td><strong>12.72</strong></td>
<td><strong>9.83</strong></td>
<td>23.29</td>
<td>19.06</td>
<td>20.19</td>
<td>13.42</td>
</tr>
<tr>
<td>EgTest04</td>
<td>19.38</td>
<td>23.43</td>
<td>21.29</td>
<td>19.72</td>
<td>8.26</td>
<td>10.43</td>
<td><strong>4.72</strong></td>
<td><strong>3.46</strong></td>
<td>15.74</td>
<td>11.26</td>
<td>11.27</td>
<td>6.67</td>
</tr>
<tr>
<td>Football</td>
<td>144.00</td>
<td>10.00</td>
<td>39.00</td>
<td>13.00</td>
<td>37.00</td>
<td>58.00</td>
<td>26.00</td>
<td>15.00</td>
<td>14.00</td>
<td>23.43</td>
<td>18.92</td>
<td><strong>12.23</strong></td>
</tr>
<tr>
<td>Hockey</td>
<td>22.19</td>
<td>56.38</td>
<td>18.39</td>
<td>66.04</td>
<td>42.43</td>
<td>37.39</td>
<td><strong>13.22</strong></td>
<td>23.34</td>
<td>32.55</td>
<td>36.16</td>
<td>18.04</td>
<td></td>
</tr>
<tr>
<td>MountainBike</td>
<td>76.22</td>
<td>48.38</td>
<td>52.91</td>
<td><strong>19.42</strong></td>
<td>49.29</td>
<td>79.06</td>
<td>38.90</td>
<td><strong>22.46</strong></td>
<td>96.03</td>
<td>62.43</td>
<td>32.72</td>
<td>27.84</td>
</tr>
<tr>
<td>Occludedface</td>
<td>57.00</td>
<td>14.00</td>
<td>17.00</td>
<td>17.00</td>
<td>36.00</td>
<td>39.00</td>
<td>15.00</td>
<td>24.00</td>
<td>16.00</td>
<td><strong>12.26</strong></td>
<td><strong>12.37</strong></td>
<td>12.40</td>
</tr>
<tr>
<td>Skating</td>
<td>176.00</td>
<td>144.00</td>
<td>87.00</td>
<td>136.00</td>
<td>74.00</td>
<td>76.00</td>
<td><strong>15.05</strong></td>
<td>204.00</td>
<td>21.00</td>
<td>19.98</td>
<td>15.05</td>
<td><strong>13.34</strong></td>
</tr>
<tr>
<td>Shaking</td>
<td>55.00</td>
<td>122.00</td>
<td>72.00</td>
<td><strong>12.00</strong></td>
<td>22.00</td>
<td>134.00</td>
<td>24.00</td>
<td>232.00</td>
<td><strong>14.00</strong></td>
<td>28.94</td>
<td>19.57</td>
<td>17.44</td>
</tr>
<tr>
<td>Subway</td>
<td>16.35</td>
<td>115.71</td>
<td>148.43</td>
<td><strong>7.82</strong></td>
<td>96.71</td>
<td>83.06</td>
<td><strong>3.39</strong></td>
<td>21.74</td>
<td>31.93</td>
<td>19.41</td>
<td>11.92</td>
<td>8.43</td>
</tr>
<tr>
<td>Sylvester</td>
<td>47.00</td>
<td>138.00</td>
<td>50.00</td>
<td><strong>9.00</strong></td>
<td>12.00</td>
<td>14.00</td>
<td>10.00</td>
<td><strong>8.00</strong></td>
<td>14.00</td>
<td>14.80</td>
<td>12.90</td>
<td>10.19</td>
</tr>
<tr>
<td>Tiger2</td>
<td>36.00</td>
<td>44.00</td>
<td><strong>48.00</strong></td>
<td>18.00</td>
<td>22.00</td>
<td>30.00</td>
<td><strong>11.00</strong></td>
<td>40.00</td>
<td>17.00</td>
<td>19.03</td>
<td>17.48</td>
<td><strong>15.82</strong></td>
</tr>
<tr>
<td>Average CLE</td>
<td>57.18</td>
<td>66.73</td>
<td>64.96</td>
<td>28.93</td>
<td>48.45</td>
<td>52.11</td>
<td>23.79</td>
<td>67.50</td>
<td>23.63</td>
<td>19.32</td>
<td><strong>16.36</strong></td>
<td><strong>12.43</strong></td>
</tr>
</tbody>
</table>

5 Conclusion

In this paper, an efficient appearance observation model, based on feature selection and feedback strategy, is proposed to augment the multi-scale tracking based compressive sensing of. Our algorithm formulates the tracking task as a dynamic state estimation problem, based on the bootstrap filter. The observation model of the bootstrap filter is represented by a naïve Bayes classifier through a very sparse random projection matrix. The weighted elements in the matrix are adaptively constructed by a target model, which is utilized to discriminate the object from its surrounding background and also to detect occlusion or sudden appearance variations. We also introduce the notion of distracting areas, the appearance of which is similar to the target. On the basis of a learning ratio determined by a feedback strategy based on the object’s scale changes and the potentially distracting regions, the observation model is constantly updated by a set of effective positive and
negative samples to reduce the accumulated error. Extended experimental results on challenging sequences show favorable performance with regard to the precision, stability and adaptation of our tracker when compared to the state-of-the-art tracking algorithms. Moreover, the proposed tracking algorithm has a reasonable computation performance. The average frame per seconds (FPS) of our current implementation is 16.2 FPS in MATLAB.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Nos. 61672429, 61272288, 61231016, 61303123), CSC-VUB scholarship (No. 201406290121), ShenZhen Science and Technology Foundation (No. JCYJ20160229172932237), the Northwestern Polytechnical University (NPU) New AoXiang Star (No. G2015KY0301), the Fundamental Research Funds for the Central Universities (No. 3102015AX007), and the Research Foundation Flanders (FWO) through the CHIST-ERA COACHES project (No GA.018.14N).

References


bayesian state estimation,” in *IEE Proceedings F (Radar and Signal Processing)*, 140(2),

25 F. Teng and Q. Liu, “Robust multi-scale ship tracking via multiple compressed features fu-


in *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*,


29 T. Chen, Y. Zhang, T. Yang, *et al.*, “Dynamic compressive tracking,” in *Proceedings of Inter-

30 H. Grabner, M. Grabner, and H. Bischof, “Real-time tracking via on-line boosting,” in *B-

**List of Figures**

1 Graphical representation of compressing a high-dimensional vector $\mathbf{x}$ to a low-
dimensional vector $\mathbf{v}_{16}$.

2 The main components of the proposed tracking algorithm.

3 The online features selection mechanism based on the weighted random matrix.
4 Visual representation of feature projection by random matrix. (a) Rectangle filters extracted by CT. (b) Rectangle filters using MSERs. (c) Weighted rectangle filters using the proposed approach; (d) The object likelihood (object segmentation). The green rectangles indicate features with high weight, while the red rectangles indicate features with low weight.

5 Search areas for positive and negative samples. (a) Positive samples defined by $\alpha$. (b) Negative samples defined by $\beta$, and $\zeta$, excluding a distractor illustrated in blue.

6 Success and precision plots for the selected video sequences labeled with ground truth.

7 Snapshots of tracking results using CT$^{16}$, DCT$^{29}$, MSCT$^{22}$ and PLCT.

8 Snapshots of tracking results under pose and illumination changes.

9 Snapshots of tracking results under background clutter.

10 Snapshots of tracking results under occlusion.

11 Snapshots of tracking results under low resolution and low contrast.

**List of Tables**

1 Used benchmark sequences.

2 Success Rate (SR) score(%). The best and the second best performing methods are shown in **Bold** and *Italic*, respectively.

3 Average Center Location Errors (in pixels). The best and the second best performing methods are shown in **Bold** and *Italic*, respectively.