Journal of Electronic Imaging

SPIEDigitalLibrary.org/jei

One-class support vector machine-assisted robust tracking

Keren Fu
Chen Gong
Yu Qiao
Jie Yang
Irene Yu-Hua Gu

One-class support vector machine-assisted robust tracking

Keren Fu,
Chen Gong,
Yu Qiao,
Jie Yang
Shanghai Jiao Tong University
Institute of Image Processing and Pattern Recognition
Shanghai, China
and
Ministry of Education of China
Key Laboratory of System Control and Information Processing
Shanghai 200240, China
E-mail: fkrsuper@sjtu.edu.cn

Irene Yu-Hua Gu
Chalmers University of Technology
Department of Signals and Systems
Signal Processing Group
Gothenburg 41296, Sweden

Abstract. *Recently, tracking is regarded as a binary classification problem by discriminative tracking methods. However, such binary classification may not fully handle the outliers, which may cause drifting. We argue that tracking may be regarded as one-class problem, which avoids gathering limited negative samples for background description. Inspired by the fact the positive feature space generated by one-class support vector machine (SVM) is bounded by a closed hyper sphere, we propose a tracking method utilizing one-class SVMs that adopt histograms of oriented gradient and 2bit binary patterns as features. Thus, it is called the one-class SVM tracker (OCST). Simultaneously, an efficient initialization and online updating scheme is proposed. Extensive experimental results prove that OCST outperforms some state-of-the-art discriminative tracking methods that tackle the problem using binary classifiers on providing accurate tracking and alleviating serious drifting.* © *2013 SPIE and IS&T* [DOI: 10.1117/1.JEI.22.2.023002]

1 Introduction

Tracking is regarded as a key point in computer vision field studies, and has been extensively researched for decades. Recently, tracking-by-detection methods[1–9] are explored to formulate tracking as a binary classification problem, which distinguishes the object from the background. That is, the target regions are regarded as positive samples and the nontarget regions are deemed as negative samples, whereas a classifier is trained to seek a decision boundary that can best separate the positive and negative. The classifiers used to tackle this problem, like support vector machine (SVM)[2] or the ones generated by Adaboost algorithm,[3] usually have a good ability to handle high-dimensional data. Babenko et al.[6] adapted multiple instance learning (MIL)[10,11] instead of traditional supervised learning by building an evolving and boosting classifier that tracks bags of image patches, and reports excellent tracking results on challenging video sequences. However, such supervised or MIL-based methods may not guarantee a closed positive feature space, and sometimes may be less robust to the outliers.[12]

The semisupervised learning based methods[7–9] are proposed to treat object tracking as an online semisupervised binary classification problem. In addition to the labeled samples, the semisupervised classification tries to use more unlabeled samples, which brings a stronger ability to handle the outliers. The deficiency of these methods is that beyond the labeled data, a large number of unlabeled data should be collected online, and also many semisupervised algorithms, like transductive SVM,[12] are highly computational, hence degrading the performance of the tracking system to be far away from real-time processing. Besides, the semisupervised based tracking methods have not totally solved the outlier problem as well, which may lead to drifting problem. This will be discussed in the following section.

In this paper, we propose a robust tracking method using one-class SVM, so we call it the one-class SVM tracker (OCST), which falls into the tracking-by-detection category. We propose that the tracking problem may be treated as one-class classification case rather than the binary case. One-class SVM[13] is proposed to estimate the distribution of high-dimensional data, and then it has been used in document classification[14] and image retrieval.[15] Recently, Gong and Cheng[16] use two competing one-class SVMs to segment foreground from video sequences. The most related work to ours is Ref. 17, because to our best knowledge, Ref. 17 is the only tracking method that has employed one-class SVM. Their method first selects candidate samples that achieve high similarity coefficients with the tracked target, and

Paper 12490 received Nov. 22, 2012; revised manuscript received Mar. 12, 2013; accepted for publication Mar. 15, 2013; published online Apr. 8, 2013.

then uses these samples to train the one-class SVM to find the center of the hyper sphere. This center sample is treated as the target estimation; however, we use one-class SVM differently from that in Ref. 17, because one-class SVM in Ref. 17 is more like a refiner rather than classifier. So, their method may not be regarded as discriminative method. In contrast, we introduce one-class SVM as a discriminative classifier while taking advantage of its ability to deal with outliers and process high-dimensional data. Moreover, we consider combining multiple features into our OCST framework.

In summary, this paper has the following contributions:

1. We demonstrate that object tracking should be regarded as a one-class problem (enclosed positive feature space), which avoids gathering limited negative samples for background description. Theoretical analysis of the reason is also shown in this paper.
2. One-class SVM is employed as discriminative classifier rather than a refiner in the conventional case,[17] and is formulated in the typical tracking-by-detection framework, bringing stronger ability to handle outliers.
3. Multiple features [histograms of oriented gradient (HOG) and 2bit binary patterns (2bitBP)] are introduced and combined using one-class SVMs.
4. Experimental results show that the proposed OCST is feasible and even outperforms some state-of-the-art binary discriminative trackers on providing accurate and stable tracking.

A preliminary conference version of this work appeared in Ref. 18.

The rest of this paper is organized as follows. Related work is described in Sec. 2. Details of our implementation with one-class SVM are demonstrated in Sec. 3. Experimental results are analyzed in Sec. 4, and conclusions and future work are in Sec. 5.

2 Related Work

2.1 *Tracking-By-Detection Methods*

Tracking-by-detection methods, or so called discriminative methods are explored to formulate tracking task as a binary classification problem and supervised or semisupervised learning methods are considered. The support vector tracker[2] uses an offline-learned SVM as classifier and embeds it into an optical flow to track moving vehicles. The final tracking position is characterized with the highest SVM score. Yet their SVM never updates online, leading to lower adaptability. In addition, the effort of building such a large off-line sample set manually is usually considerable. Grabner and Bischof[3] utilize the Adaboost algorithm to perform online feature selection. Their positive and negative samples are collected online, so no additional manual effort is needed. However, their method may cause a drifting problem in a complex background, due to the potential effect of outliers,[12] which is also a common problem of these supervised methods. Babenko et al.[6] uses MIL[10,11] to train the appearance classifier, resulting in a relatively robust tracking, and an online boosting (OB) algorithm for MIL is also presented. Tang et al.[8] adopt co-training to take the advantage of multiple independent features for training a set of classifiers online. The classifiers then collaboratively classify the unlabeled data, and use these newly labeled data to update each other. Each feature is used to train an online SVM, and their outputs are combined to give the final classification results. Stalder et al.[9] use an offline detector, on-line supervised identifier, and semisupervised tracker to extend semisupervised tracking by object specific and adaptive priors; however, their model relies strongly on the prior classifier, leading to frequent target loss.

These tracking-by-detection methods mentioned above treat tracking as a binary classification problem, for supervised, MIL,or semisupervised learning-based methods. So, their common problem is that they could not fully handle the outliers (Fig. 1), leading to inaccurate tracking. Also, semisupervised learning requires a large number of unlabeled samples for learning simultaneously with extra time cost in feature extraction and classification.

2.2 *Our Motivation*

The key insight of our approach is to take advantage of one-class SVM for tackling the tracking problem. A vivid illustration of the difference between one-class SVM and a supervised binary classifier as well as a semisupervised binary classifier is shown in Fig. 1. As is shown in Fig. 1(d), in the typical tracking-by-detection framework, the tracker would use a sliding window technique to search for the target in the current frame [denoted as ground truth in Fig. 1(d)]. The searching region is usually centered at the target position in the previous frame. Then, the unlabeled candidate samples corresponding to each sliding window are collected and classified using binary classifiers. We could see that with the supervised or semisupervised learning method, when training samples are collected, a decision boundary is usually generated to classify the positive and negative categories with the maximum margin or minimum error. However, an unlabeled candidate sample (denoted in diamond) that is far away from the positive sample set (i.e., an outlier) may be misclassified as positive [Fig. 1(a) and 1(b)]. This unlabeled sample may correspond to the region that contains a nontarget object or part of the tracked target [notice the solid arrows from Fig. 1(c) to 1(d)] or even any changing background whose information is not previously obtained to train the classifiers. Moreover, a high classification score may be obtained for this unlabeled sample due to its further distance from the decision boundary [see Fig.1(b)]; thus, it is more likely to contribute to the final tracking position, leading to an inaccurate location or drifting.

The key point of the problem above is that the positive feature space separated by the decision boundary in binary case may be unenclosed [Fig. 1(a) and 1(b)]. Moreover, using a limited number of labeled negative or unlabeled samples could hardly describe or estimate the total complex moving environment. Thus, in this case, one-class SVM should be introduced to solve this problem. It can estimate the distribution of high-dimensional samples, and the classifier only needs positive samples as input, and hence, it effectively avoids collecting negative samples for background description. From another sight, one-class SVM makes the judgment whether an unlabeled sample is the target object or the remaining, using the only information of positive training samples.

Another important point in this paper is that the separated positive feature space should be bounded (enclosed) by a

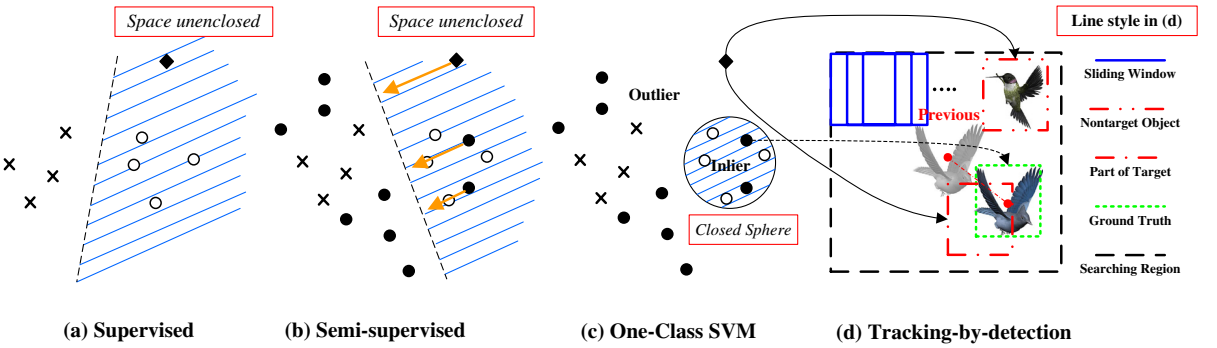


Fig. 1 An illustration of the difference between using supervised binary classifier (a) [e.g., binary support vector machine (SVM)], semisupervised binary classifier (b), and one-class SVM (c). (d) shows a typical tracking-by-detection framework. In (a)–(c), the circles represent the training positive samples, and the crosses stand for the negative ones. The solid balls represent unlabeled candidate samples and the diamond stands for a special unlabeled sample used for illustration convenience.

closed sphere. Thus, an unlabeled sample is classified as "inlier" or "outlier" rather than "positive" or "negative." In Fig. 1(c), all samples far away from the positive sample set, i.e. the closed hyper sphere, would be excluded as outliers by one-class SVM. Only the real region of the tracked target will be classified as inlier [notice the dash arrow from Fig. 1(c) to 1(d)], which results in more accurate tracking as well as alleviating the drifting risk.

3 One-Class SVM Assisted Tracking

In this section, we introduce our tracking algorithm, the OCST, which uses one-class SVM as a discriminative classifier, and takes the advantage of the dense HOG[19] and 2bitBP[20] features. We begin with a brief description of one-class SVM. Next, we illustrate the details of feature extraction and combination. Finally, we review our online tracking framework.

3.1 *One-Class SVM*

The SVM algorithm as it is usually construed is essentially a binary-class algorithm[21] (needs negative and positive samples), especially in computer vision field tasks like tracking and object detection. However, when only positive samples can be acquired while negative samples have no certain distribution and remain irregular, the one-class SVM should be considered. The one-class SVM[13] algorithm considers the following problems. Supposing that there is a dataset drawn from an underlying probability distribution P, one needs to estimate a "simple" subset S of the input space such that the probability that a test point from P lying outside of S is bounded by some a prior-specified $v \in (0, 1)$, because in most cases, it is more feasible to solve for such S rather than the original distribution P. The solution for this problem is obtained by estimating a function f, which is positive on S and negative on $\bar{S}$. In other words, Schölkopf et al.[13] developed an algorithm that returns a function f that is positive in a "small" region capturing most of the data vectors, and is negative elsewhere.

Their strategy can be summarized as mapping the data into a feature space H using an appropriate kernel function, and then trying to separate the mapped vectors from the origin with maximum margin.

Here, let $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3 \ldots \mathbf{x}_m$ be training samples (bold is used since they are usually feature vectors) belonging to one known class X, where X is a compact subset of R^N according to Ref. 13. Let $\phi: X \to H$ be a kernel map that transforms the training samples to another space. Then, to separate the data set from the origin, one needs to solve the following quadratic programming problem:

$$\min \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{vl}\sum_{i=l}^{m}\xi_i - \rho, \tag{1}$$

$$\text{subject to } \mathbf{w}\cdot\phi(\mathbf{x}_i) \geq \rho - \xi_i \quad i = 1, 2, 3, \ldots, m, \quad \xi_i \geq 0. \tag{2}$$

Nonzero slack variables ξ_i are penalized in the objective function. If $\mathbf{w}$ and ρ are worked out by solving the problem above, then the decision function could be formed as

$$f(\mathbf{x}) = \mathbf{w}\cdot\phi(\mathbf{x}) - \rho. \tag{3}$$

Equation (3) will be positive for most samples $\mathbf{x}_i$ contained in the training set; $v \in (0, 1)$ is a parameter that controls the number of samples contained in the hyper sphere. When v is larger, the energy function tends to focus less on the slack variables and focuses more on the separating margin. Thus, less training samples are contained by the hyper sphere. On the contrary, when v turns smaller, the hyper sphere will contain more training samples meanwhile reducing the classification error. However, it would simultaneously degrade the ability of excluding outliers (see Ref. 13 for details). The v for our one-class SVMs will be introduced in Sec. 4.

In fact, Eqs. (1) and (2) could be switched to a dual problem via employing the Lagrangian multipliers and solved by using quadratic programs. The most widely used radial basis function kernel is chosen in this paper to process the feature mapping. For more details, we strongly recommend readers to refer to Ref. 13. Despite the solution to Eqs. (1) and (2), one thing should be noted that because the distance between a specific sample $\bar{\mathbf{x}}$ and the decision boundary $f(\mathbf{x}) = 0$ is $|f(\bar{\mathbf{x}})|/\|\mathbf{w}\|$, thus, when a sample $\bar{\mathbf{x}}$ is positive and obtains a larger $f(\bar{\mathbf{x}})$, it is further from the decision boundary, and may be more reliable. This criterion would help us choose the optimal candidate.[2]

Based on this theory, in a tracking task, we also search for a hyper sphere that contains most of the training samples obtained consequently from the target region. After training, the decision boundary allows us to choose the most appropriate candidate region.

3.2 *Feature Selection*

Beyond using one-class SVM, the feature selection for tracking is also an important part. Good features usually have nice ability to characterize the unique appearance of the tracked target meanwhile distinguishing it from the complex background and other objects. In recent research, many kinds of features such as HOG features,[19] color histogram,[22] Haar-like features,[23] Gabor features,[24] and local binary patterns (LBP) features[25] are adopted for tracking. Tang et al.[8] uses both color histogram and HOG features to train their corresponding SVMs while Grabner et al.[3] use Haar-like, LBP, and HOG features. However, the color histogram has a relatively weaker distinguishing ability, especially when the background color is similar to the target color, thus leading to drifting, e.g., the meanshift (MS) algorithm.[22] So, in this paper, we reject the color histogram and tend to choose the kind of features that describe the target's shape and texture, and the selected features should also be invariant to illumination changes.

We ultimately base our tracker on HOG and a new feature called 2bitBP,[20] which is indeed derived from a Haar-like feature, for characterizing the appearance of target. In this paper, we fuse these two feature extraction processes into the same scheme. First, the target region is divided into some overlapped square blocks, similar to the standard HOG extraction (Fig. 2). In each block, we extract both HOG and 2bitBP features. For HOG features, each block contains four cells. If nine bins are chosen in each cell,[19] a 36-dimension vector is constructed in every block. For 2bitBP features, the block is divided in the horizontal and vertical directions, and the sum intensity [denoted as $\mathbf{I}(a_i)(i \in \{1,2,3,4\})$ in Fig. 2] of the two sides is computed and compared to obtain a 2bit code (Fig. 2).[20]

In traditional HOG for human detection,[19] the recommended block size (BS) is 16 for 64 width and 128 height human images. However, in tracking, the BS should be adaptive to the size of a specific tracked target. So, we define the BS as

$$\mathrm{BS} = \min\left\{\frac{W}{N}, \frac{H}{N}\right\}, \tag{4}$$

where W and H are the width and height of the tracking rectangle, respectively. We force the minimum number of blocks (nonoverlapped) in the horizontal or vertical direction to be N. The default N is chosen as 4, which achieves the best performance in our experiment. When N becomes smaller, the description ability of each block will be degraded for both HOG and 2bitBP features, while a larger N will result in a high-dimensional feature vector for every tracked region. According to the analysis above, the dimension of our HOG feature is at least $(4+3)\times(4+3)\times 36 = 1764$ and the dimension of 2bitBP is $(4+3)\times(4+3)\times 2\mathrm{bit} = 98\mathrm{bit}$. Here, we stack all the bits of the 2bitBP feature that corresponds to each sliding window into a single feature vector and use one-class SVM to classify these "bit" vectors. Such high-dimensional HOG and 2bitBP feature vectors could take the advantage of the high-dimensional processing ability of one-class SVM.

Last, but not the least, the entire block region should be centered in the tracking rectangle, so that the center-surrounded feature can be extracted, thus performing more accurate locating. A vivid example of our feature extraction for a $W = 75$ and $H = 150$ target is shown in Fig. 2.

3.3 *On-Line Tracking*

3.3.1 *Motion model*

Generally, our tracker maintains the target location l_n at every frame step, n. In a certain frame n, a candidate sample set $X^r = \mathbf{x}: \|l(\mathbf{x}) - l_{n-1}\| < r$ is collected using a sliding window (rectangle) technique, in which $l(\mathbf{x})$ is the location (consisting of only horizontal and vertical coordinates) of a candidate sample $\mathbf{x}$, which corresponds to a specific window, and r represents the searching radius. We then compute a combining score function $S(\mathbf{x})$ through one-class SVM for all $\mathbf{x} \in X^r$, and update the target location using the position corresponds to the candidate sample which maximizes the score as

$$l_n = l[\operatorname{argmax}_{\mathbf{x}\in X^r} S(\mathbf{x})], \tag{5}$$

where the combining score function $S(\mathbf{x})$ will be introduced in the next subsection. In other words, we do

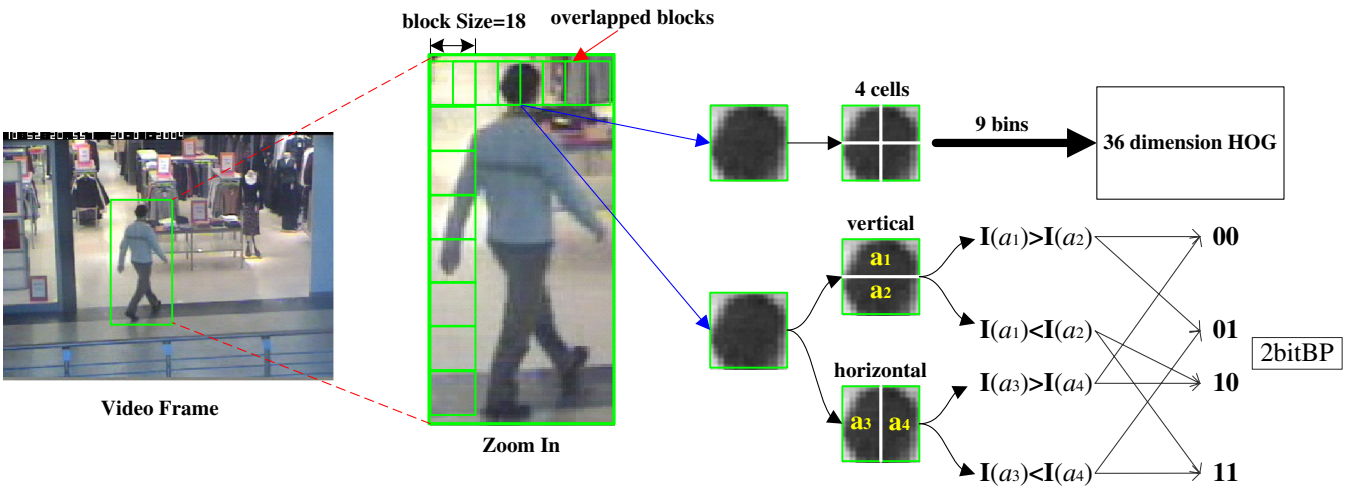


Fig. 2 An example of our feature extraction for a $W = 75$ and $H = 150$ target; $\mathrm{BS} = [\min\{75/4, 150/4\}] = 18$. The feature region is centered and forced to contain at least four nonoverlapped blocks in the horizontal or vertical directions while the stride (block overlap) is fixed at half of the block size (BS) similarly to Dalal et al.[19] In each block, a 36-dimension histograms of oriented gradient (HOG) and 2bit binary patterns (2bitBP) are generated. In the above figure, $\mathbf{I}(a_i)(i \in \{1,2,3,4\})$ denotes the sum of pixel intensity in region a_i.

not maintain a distribution of the target's location at every frame, and our motion model is such that the location of the target in frame n is equally likely to appear within a radius r of the target location in frame $n-1$. This could be extended with something more sophisticated, such as a particle filter.[4] Using a prior distribution produced by particle filters or Kalman filters is likely to perform well in normal cases. However, it may degrade the ability of trackers to handle abrupt or abnormal cases like sudden moving orientation changes of the tracked target. So, for more universal consideration, we assume that the target location in frame n may obey the uniform distribution centered at the target location in frame $n-1$. The searching radius r depends on the moving speed of the tracked target. Since the target's motion between two adjacent frames is not that exaggerated, we set the searching radius r to 10 to 20 pixels, which works well in most cases.

3.3.2 *Initialization and updating*

In the beginning of tracking process, an initial tracking rectangle should be given. This rectangle could be chosen manually or provided by object detection algorithms[26] or even change detection algorithms for moving object. When the first rectangle is given, the tracker begins tracking. Supervised and semisupervised tracking methods need to collect both positive and negative samples online in the first several frames. Tang et al.[8] adopts the MS tracker[22] to track targets in the beginning several frames in order to collect the positive and negative samples. Compared with their method, our method is much simpler and more effective.

We construct a positive sample pool to store the latest positive samples (assume each sample contains HOG and 2bitBP features $\mathbf{x}=\{\mathbf{x}_H,\mathbf{x}_B\}$). Then these samples are used to train our one-class SVMs. By the way, we first build a score function that is similar to the SVM score in the binary classification case.[2] Our score function is obtained using the decision function [Eq. (3)] as

$$S(\mathbf{x})=w_H e^{\alpha_H f_H(\mathbf{x}_H)}+w_B e^{\alpha_B f_B(\mathbf{x}_B)}, \tag{6}$$

where f_H and f_B are the one-class SVM decision functions for HOG and 2bitBP, respectively. The reason of using the decision function f is that, while f maintains positive and turns larger, it indicates that a candidate sample is positive and is further from the hyper sphere, leading to more reliable result. This fact is also consistent with the binary case[2]; α_H and α_B are scaling factors which help pull down Eq. (6) when the exponents of the two terms turn negative. In practice, we find $\alpha_H=\alpha_B=10$ to be suitable for normalized features. The weights w_H and w_B are computed using classification errors as

$$w_H=1-\epsilon_H/(\epsilon_H+\epsilon_B+\tau), \tag{7}$$

$$w_B=1-\epsilon_B/(\epsilon_H+\epsilon_B+\tau), \tag{8}$$

in which ϵ_H and ϵ_B are the respective classification errors of the two one-class SVMs; τ is a small number which avoids the divide-by-0 issue; $w_H+w_B=1$ is also satisfied above.

From Eq. (6), we can see that $S(\mathbf{x})>1$ roughly means the corresponding sample is classified as inlier, while $S(\mathbf{x})<1$ is for an outlier. Because Eq. (6) helps us visualize the classification results, in practice, we always select the sample which makes Eq. (6) achieve its maximum according to Eq. (5) in each frame as our target, and add it into the pool.

In addition, we set a hard threshold to avoid low score samples being added into the positive sample pool. When the current score function [Eq. (6)] at the global maximum is lower than the threshold, the target may suddenly become seriously occluded by some other object or disappearing near the scenario boundary. This threshold should be manually determined, as it represents how conservative one wants to be in their updating scheme.

The initializing process is illustrated by Fig. 3. In the first frame, the sample pool is initialized (empty) and the first positive sample is pushed into the pool. The only sample is then used to train the classifiers. In this case, the one-class SVM may degrade into a nearest neighbor classifier, which seeks a nearest neighbor in feature space of the only training sample. In the following frames (2nd, 3rd, 4th...), more and more samples which maximize Eq. (6) are collected online and pushed into the pool to train the one-class SVMs.

Gradually, the added target samples may increase the computational cost and burden of the classifiers. So, we always retain the latest k (e.g., $k=30$) samples in our sample pool, and the relatively older samples are thrown away. Thanks to one-class SVMs, taking a small number of high-dimensional positive samples can achieve a good generalization performance. Actually, we use a "First In, First Out" (FIFO) to realize this process. When the FIFO is full, adding a new positive sample will cause an old sample to pop out. Figure 4 shows our on-line tracking framework.

4 Experiments and Analysis

Our method is validated on a large amount of video sequences and compared to some state-of-the-art discriminative tracking methods, including OB[3] and beyond semiboosting

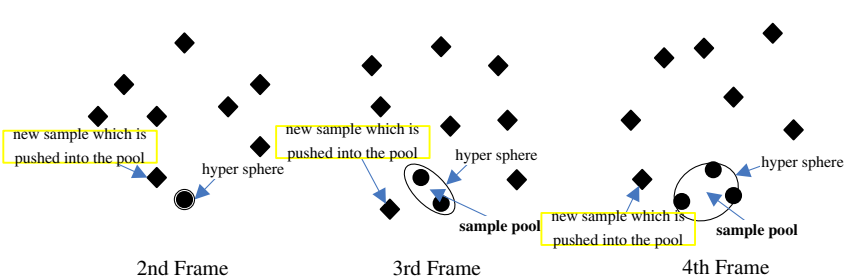


Fig. 3 The initializing process in the first several frames. In the 2nd frame, the unlabeled sample that is nearest to the decision sphere is chosen as our positive sample. The one-class support vector machine (SVM) may degrade into the nearest neighbor classifier. In the following frames, more and more positive samples that maximize Eq. (6) are pushed into the pool.

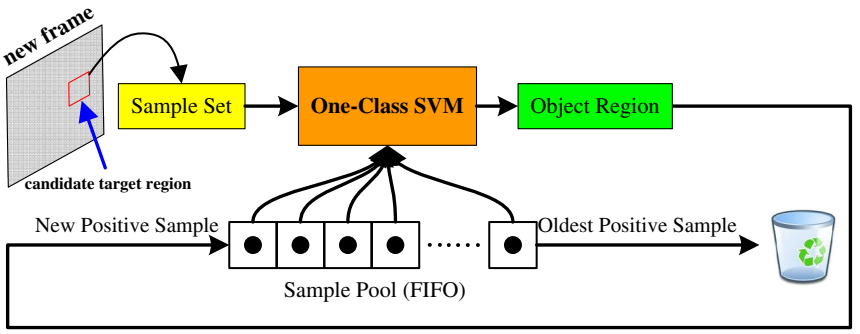


Fig. 4 Online tracking framework based on one-class support vector machine (SVM). The unlabeled sample set in new frame is extracted uniformly surrounding the last target position. The object region is chosen as the sample which maximizes Eq. (6). Our positive sample pool is realized using a "First In, First Out" (FIFO). When the FIFO is full, adding a new positive sample will cause an old sample to pop out.

(BSB)[9] (the codes of these methods are available at the authors' webpage). In addition, a typical MS method[22] and a naive nearest neighbor tracker (NNT) are considered into comparison. As the v seems to be an important parameter for a one-class SVM, in practice, we try different values for v, from 0.1 to 0.9. However, because we always select the candidate with maximum score as target, the performance of our tracker seems to be insensitive to the change of v. Considering that a too-large v results in low classification accuracy while a small v leads to less robustness against outliers, we use the typical 0.5 value for our one-class SVMs. This parameter remains consistent in all our experiments.

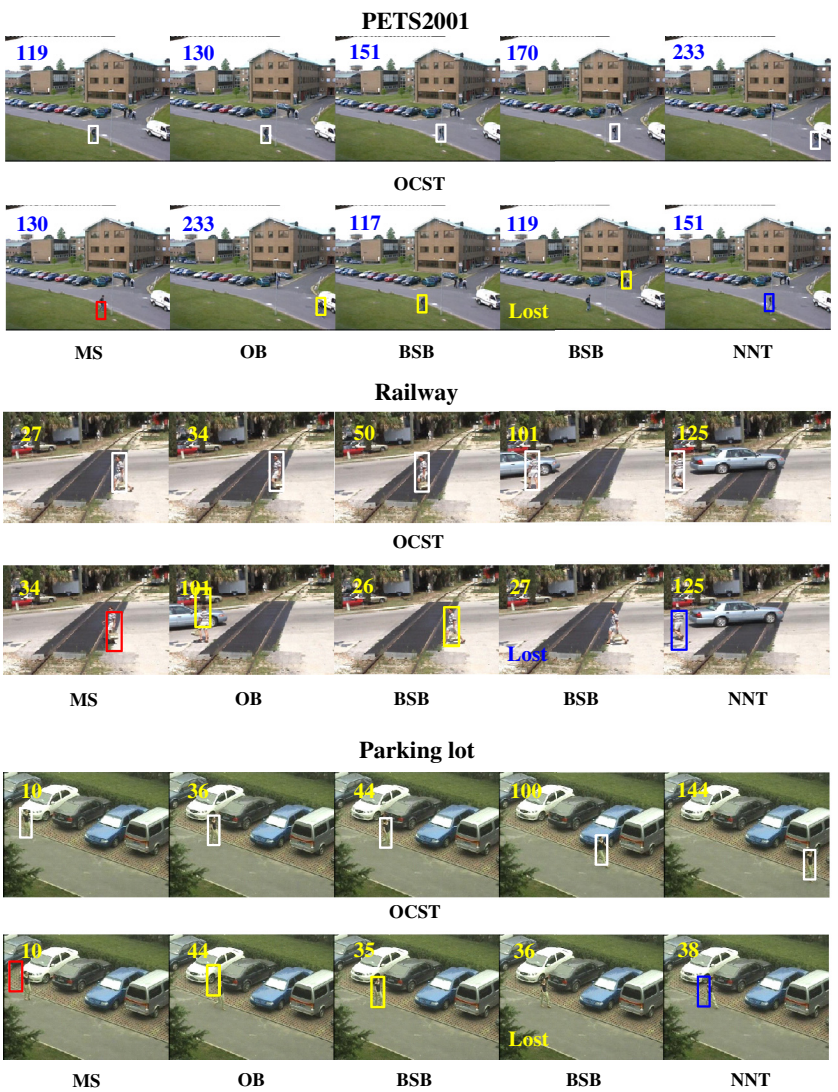


Fig. 5 Tracking performance comparison on sequences PETS2001, railway, and parking lot. Our method is able to provide accurate tracking while the other methods like meanshift (MS), online boosting (OB), beyond semiboosting (BSB) and NNT has drifting or inaccurate locating problems. The comparison between our one-class SVM tracker (OCST) and another certain tracker can be obtained in the frame with the identical frame number.

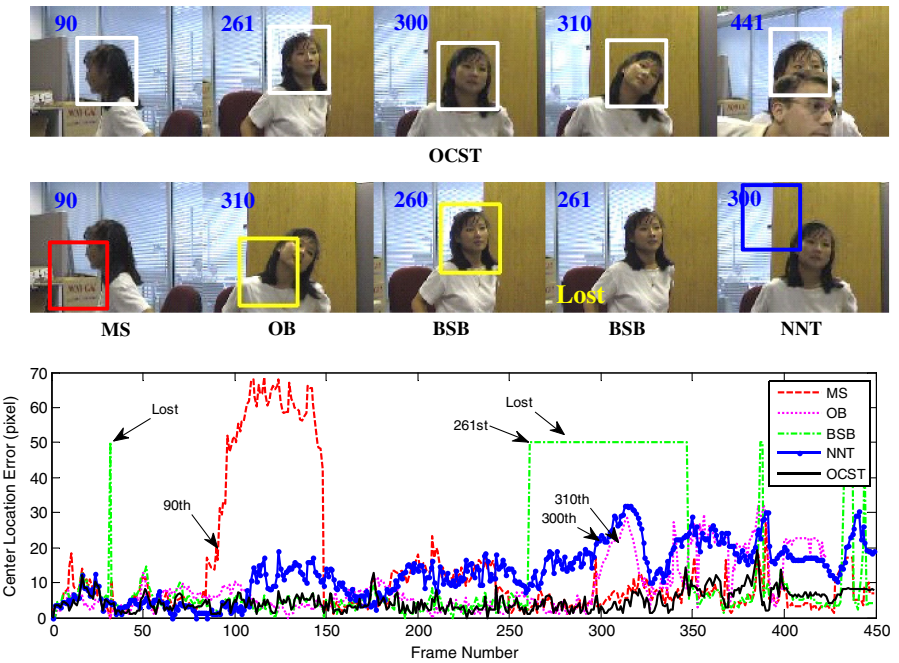


Fig. 6 Tracking performance comparison on girl sequence. The tracking errors of different methods are also shown. OCST = one-class SVM tracker; MS = meanshift, OB = online boosting; BSB = beyond semiboosting; NNT = nearest neighbor tracker.

4.1 *Tracking in Complex Background*

Figure 5 shows the comparison results on three sequences: PETS2001, railway, and parking lot. The challenges of tracking these sequences are complex backgrounds and abrupt background changes. MS drifts seriously when sudden background changes occur, such as when the pedestrian in PETS2001 gets out of the grassland and the buddy in railway starts to cross the railroad.

OB combines multiple features, so it is more robust to the complex background and environmental changes. However, it still treats tracking as a binary classification problem, and thus, the tracking rectangle may sometimes drift, leading to inaccurate locating, as is shown in the 101st frame of the railway sequence and the 44th frame of parking lot sequence.

The BSB method, which combines an off-line detector, supervised on-line identifier, and semisupervised tracker,

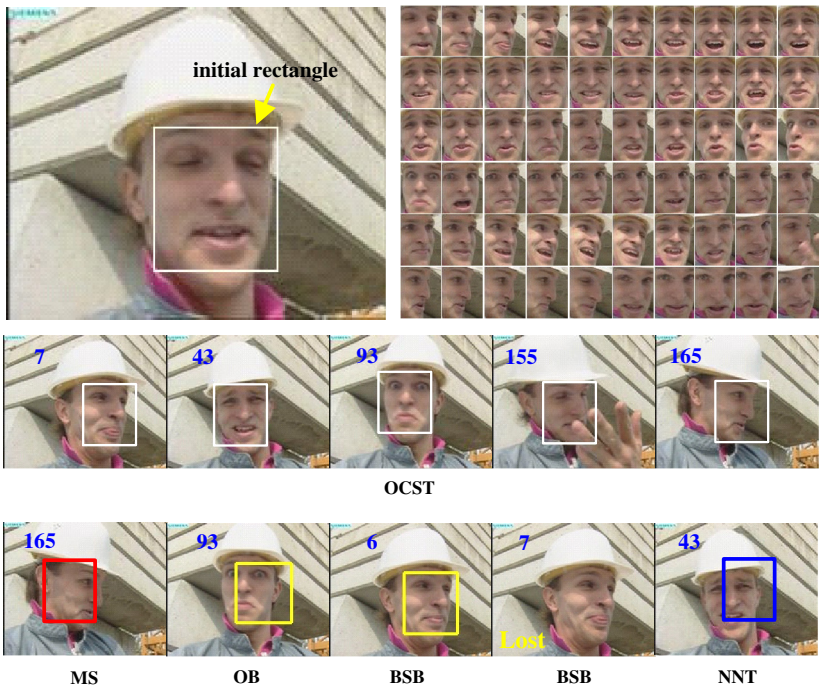


Fig. 7 Tracking performance comparison on foreman sequence. The initial frame with tracking box is presented while the tracked face regions of our method extracted in every three frames are shown on the right. OCST = one-class SVM tracker; MS = meanshift, OB = online boosting; BSB = beyond semiboosting; NNT = nearest neighbor tracker.

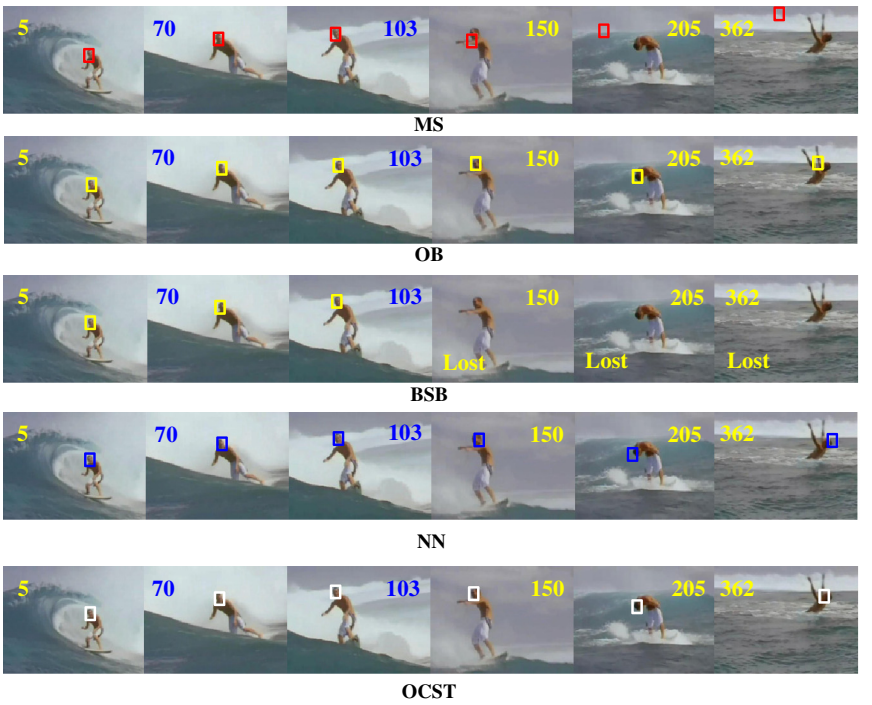


Fig. 8 Tracking performance comparison on surfer sequence. OCST = one-class SVM tracker; MS = meanshift, OB = online boosting; BSB = beyond semiboosting; NNT = nearest neighbor tracker.

is sometimes likely to be confused by target appearance changes. As the pedestrian in PETS2001 gets out of the grassland, the tracker makes the wrong decision that the target is lost, and starts to search for the target in frame 118, and then locates a wrong target in frame 119.

The NNT searches for the region that is nearest to the original target through computing the Bhattacharyya similarity coefficients[17] in feature space. However, the Euclidean distance may not really characterize the distance in the feature space, especially in a high-dimensional representation, so inaccurate locating may happen.

Relatively, our method locates the target more accurately on these three sequences. Combining the dense HOG feature, 2bitBP feature, and one-class SVM classification, the shape and texture information are well-extracted, and only the real target region in the next frame will be found correctly, regardless of the abrupt background changes or if the sudden occlusion happens. The comparison between our OCST and another certain trackers can be obtained in the frame with the identical frame number in Fig. 5.

4.2 *Tracking Part of Human Body*

Tracking a certain part of the human body, like the head, face, or eyes, is very important in identity recognition, video conferencing, and user interaction. We track some parts of the human body using the state-of-the-art methods as well as our OCST method. Usually, tracking certain part of the human body is tougher than tracking a single object that is rigid because we should consider the pose changes.

In the girl sequence (Fig. 6), we track the head of the girl. The challenges include the pan, tilt, zoom control, occlusion by another face, 360-deg rotation, and the flesh-colored board in the background. Our method provides more satisfactory results, even when the head leans and is partially occluded by some other face, whereas the tracking rectangles of other methods still drift when some obvious pose changes occur. As the ground truth of this sequence is also available,[27] we also measure the pixel-wise tracking errors of these methods. From the curves presented in Fig. 6, it can be concluded that methods like MS, OB, and NNT may result in serious drifting when the head leans and rotates, and the BSB will lose target several times during the whole tracking process (values of the green dot-dash curve in Fig. 6 that are rendered 50 indicate the time when the BSB makes the wrong judgment and the target is lost). Compared with other methods, our approach provided steadier tracking with relatively lower error.

In the Foreman sequence (Fig. 7),[28] the face is tracked. Our method also provides impressive tracking result,

Table 1 Comparison of the number of correctly tracked frames. Best result(s) in each sequence are highlighted in bold.

Sequence	Total frames	MS	OB	BSB	NN	OCST
PETS2001	250	127	**250**	117	**250**	**250**
Railway	135	73	105	50	125	**135**
Parking lot	157	5	35	57	37	**157**
Girl	448	340	406	96	266	**448**
Foreman	180	**180**	179	75	110	**180**
Surfer	367	150	362	103	188	**367**
Average accuracy	N/A	54.1%	81.6%	35.2%	64.2%	**100**%

Fig. 9 Tracking score of the whole tracking process. In this sequence, a white car is tracked by our one-class SVM tracker (OCST) method. When the target is partially occluded by other objects, the curve drops down to some level, but comes back in a short time. When part of the target disappears near the boundary of the scenario, the score curve drops enormously and never comes back.

regardless of different head poses, exaggerated facial expression conversions, and partial occlusion by a waving hand occurs. The initial frame with the tracking box is presented, while the tracked face regions of our method extracted in every 3 frames are shown on the right. Comparison between other state-of-the-art methods and our method on a certain frame also can be obtained in Fig. 7.

In the surfer sequence (Fig. 8),[29] we also track the head. As this sequence has a relatively monotonous background, the MS performs better than before, but as abrupt pose changes occur, the tracker drifts away. Both OB and OCST perform well on this sequence, noting that in the last frame, OCST still sticks to the right position.

Finally, Table 1 shows quantitative results for all sequences. A frame is considered as correctly tracked if the real target rectangle overlap with the tracking rectangle is larger than 50%. Thus, this criterion directly shows whether a tracker presents serious drifting during tracking process. For our method, slight deviation may be generated in some frames, but no serious drifting happens in the selected video sequences.

In summary, our method performs relatively accurate and stable tracking compared with other compared state-of-the-art methods. This should be attributed to the ability of one-class SVM for dealing with high-dimensional data. Besides, combining dense HOG and 2bitBP features captures the fine

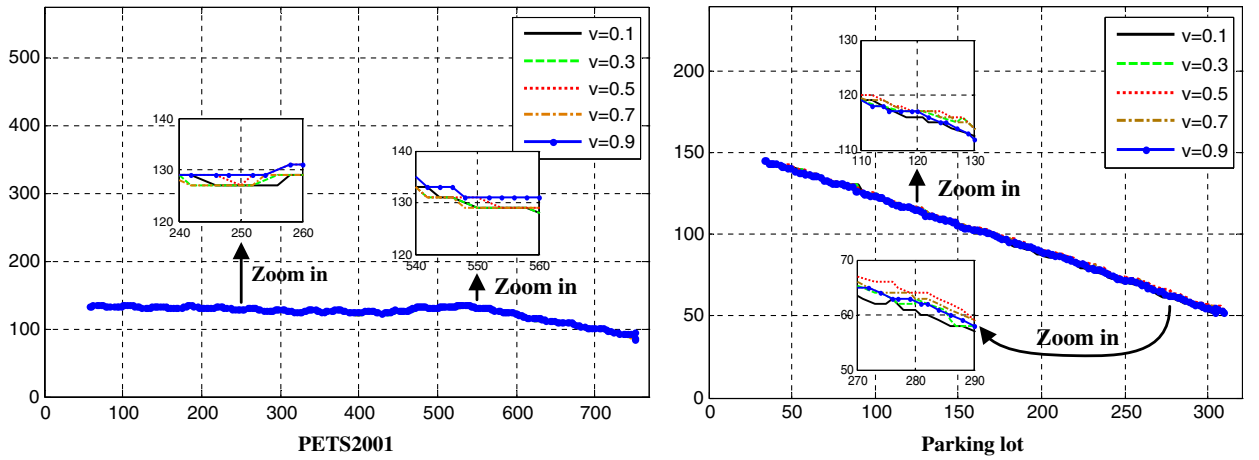


Fig. 10 Changes in parameter *v* results in similar trajectories on the PETS2001 and parking lot sequences. The range of the *x*- and *y*-axes is equal to the width and height of the corresponding video frames. The axis unit is pixel.

shape and texture characteristics of the target. Regarding tracking as a one-class classification problem may well-handle the outliers, and contribute to the alleviation of drifting problem to some extent.

4.3 *Tracking Score*

We also validate our method on the Massachusetts Institute of Technology traffic data set.[30] Figure 9 shows a typical example. In the first frame, the initial box is given to bound a white car in the road. In the 30th frame, the white car is then partially occluded by the lamp post and slightly overlapped by another car, leading to a corresponding classification score curve, which is obtained from Eq. (6), dropping down to about 0.7. However, the positive sample pool of our OCST is updated online, and thus, this appearance change of the target is overcome by adding new samples in the training pool in a short time, which leads to the classification score curve going up.

In the following 99th, 148th, and 282nd frame, the target is continuously overlapped by some other cars. When the classification score curve descends, it will come back to the original level in a short time. So, our system has suitability to target appearance modification and partial occlusion. In the 360th frame, when part of the target get out of the scenario, the classification score curve drops down enormously below 0.6 (to about 0.3), then to 0.2, and never comes back. From the value of the classification score curve of the current frame, one can judge whether or not the object is partially occluded or disappeared. Actually, when the curve is lower than 0.3, we can make the decision that the target is disappearing and stop our tracking.

4.4 *Adjustment of Parameter* ***v***

As v seems to be an important parameter for one-class SVM in the original theory,[11] which controls the size of the estimated positive feature space, in practice, we have tried different v values, varying from 0.1 to 0.9. Figure 10 shows the tracking results (trajectories) under different v values, including 0.1, 0.3, 0.5, 0.7, and 0.9. The left subfigure and right subfigure are from the aforementioned sequence PETS2001 and Parking lot, respectively. It could be concluded that under different v values, the resulting trajectories turn out to be very similar and nearly overlap each other. This indicates the performance of our tracker is insensitive to the parameter v of one-class SVM. The reason should be attributed to the fact that a varied v would not change the score rank of the candidates too much. This is natural because the high score candidates are usually those most inside ones (Fig. 11). When v turns larger, the sphere will shrink toward the sphere center. As is shown in Fig. 11, in both cases (small v and large v), the score rank/order of the candidates does not change too much. So when we seek the maximum score candidate, the final result also changes very little (Fig. 10). According to the above conclusion, we typically set $v = 0.5$ in all experiments, as was introduced at the beginning of this section.

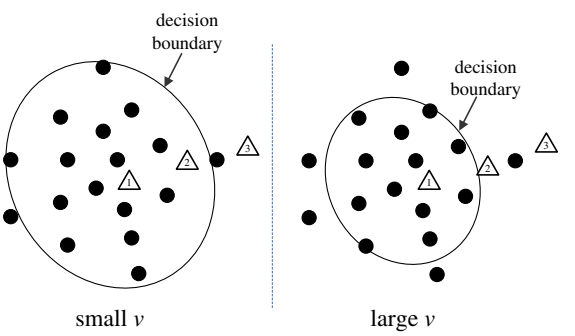


Fig. 11 Illustration of the effect of parameter *v*. Solid circles represent the training positive samples, and the three triangles stand for three candidates. Despite that small *v* results in large estimated region while large *v* leads to small estimated region, because high-score candidates are usually those most inside ones, the score rank of the three candidates under these two cases (small *v* and large *v*) remains the same (both are candidate1 > candidate2 > candidate3).

5 Conclusion and Future Works

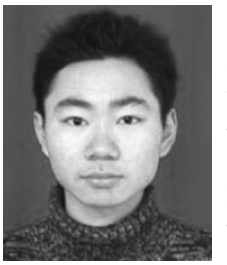
In this paper, we propose a tracking method using a one-class SVM. Combining the dense HOG feature and 2bitBP with a one-class SVM, OCST may well-handle the outliers and alleviate drifting. Because in this paper OCST is still a holistic tracker, a challenge for us in the future is trying to track articulated objects that cannot be easily delineated with a bounding box. These objects may require a part-based appearance model, which may let us develop our OCST for part-based learning.

Acknowledgments

This research is partly supported by National Natural Science Foundation of China (Grant No. 61273258, No. 61105001), PhD Programs Foundation of Ministry of Education of China (Grant No. 20120073110018).

References

1. M. D. Breitenstein et al., "Robust tracking-by-detection using a detector confidence particle filter," in *Proc. IEEE 12th Int. Conf. on Comput. Vis.*, pp. 1515–1522, IEEE, Kyoto, Japan (2009).
2. S. Avidan, "Support vector tracking," *IEEE Trans. Pattern Anal. Mach. Intell.* **26**(8), 1064–1072 (2004).
3. H. Grabner and H. Bischof, "On-line boosting and vision," in *Proc. IEEE Comput. Soc. Conf. on Comput. Vis. and Pattern Recognit.*, Vol. 1, pp. 260–267, IEEE, New York (2006).
4. R. Hess and A. Fern, "Discriminatively trained particle filters for complex multi-object tracking," in *Proc. IEEE Conf. on Comput. Vis. and Pattern Recognit.*, pp. 240–247, IEEE, Miami, Florida (2009).
5. S. Avidan, "Ensemble tracking," *IEEE Trans. Pattern Anal. Mach. Intell.* **29**(2), 261–271 (2007).
6. B. Babenko, M. H. Yang, and S. J. Belongie, "Visual tracking with online multiple instance learning," in *Proc. IEEE Conf. Comput. Vis. and Pattern Recognit.*, pp. 983–990, IEEE, Miami, Florida (2009).
7. H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *Proc. 10th European Conf. on Comput. Vis.: Part I*, pp. 234–247, Springer-Verlag, Berlin, Heidelberg (2008).
8. F. Tang et al., "Co-tracking using semi-supervised support vector machines," in *Proc. IEEE 11th Int. Conf. on Comput. Vis.*, pp. 1–8, IEEE, Rio de Janeiro (2007).
9. S. Stalder, H. Grabner, and L. Van Gool, "Beyond semi-supervised tracking: tracking should be as simple as detection, but not simpler than recognition," in *Proc. IEEE 12th Int. Conf. on Comput. Vis. Workshops*, pp. 1409–1416, IEEE, Kyoto, Japan (2009).
10. P. Viola, J. Platt, and C. Zhang, "Multiple instance boosting for object detection," in *Proc. Adv. in Neural Inform. Process. Syst.*, pp. 1417–1426, MIT Press, Cambridge (2005).
11. T. G. Dietterich, R. H. Lathrop, and L. T. Perez, "Solving the multiple-instance problem with axis parallel rectangles," *Artif. Intell.* **89**(1–2), 31–71 (1997).
12. X. Zhu and A. Goldberg, *Introduction to Semi-Supervised Learning*, Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan & Claypool Publishers, San Rafael, California (2009).
13. B. Schölkopf et al., "Estimating the support of a high-dimensional distribution," *J. Neural Comput.* **13**(7), 1443–1471 (1999).
14. L. M. Maneviitz and M. Yousef, "One-class SVMs for document classification," *J. Mach. Learn. Res.* **2**, 139–154 (2002).
15. Y. Chen, X. S. Zhou, and T. S. Huang, "One-class SVM for learning in image retrieval," in *Proc. IEEE Int. Conf. on Image Process.*, Vol. 1, pp. 34–37, IEEE, Thessaloniki (2001).
16. M. L. Gong and L. Cheng, "Foreground segmentation of live videos using locally competing 1SVMs," in *Proc. IEEE Conf. on Comput. Vis. and Pattern Recognit.*, pp. 2105–2112, IEEE, Providence, Rhode Island (2011).
17. L. L. Zhenjun Han, Q. Ye, and J. Jiao, "Visual object tracking via one-class SVM," in *Proc. Int. Conf. on Comput. Vis.*, pp. 216–225, Springer-Verlag, Berlin, Heidelberg (2011).
18. K. Fu et al., "One-class SVM assisted accurate tracking," in *Proc. 6th ACM/IEEE Int. Conf. on Distributed Smart Cameras*, pp. 1–6, IEEE, Hong Kong, China (2012).
19. N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. on Comput. Vis. and Pattern Recognit.*, Vol. 1, pp. 886–893, IEEE, San Diego, California (2005).
20. Z. Kalal, J. Matas, and K. Mikolajczyk, "Online learning of robust object detectors during unstable tracking," in *Proc. IEEE 12th Int. Conf. on Comput. Vis. Workshops*, pp. 1417–1424, IEEE, Kyoto, Japan (2009).
21. V. Vapnik, *The Nature of Statistical Learning Theory*, Springer–Verlag, New York (1995).
22. D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proc. IEEE Conf. on Comput. Vis. and Pattern Recognit.*, Vol. 2, pp. 142–149, IEEE, Hilton Head Island, South Carolina (2000).
23. P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE Comput. Soc. Conf. on Comput. Vis. and Pattern Recognit.*, Vol. 1, pp. I-511–I-518, IEEE (2001).
24. T. S. Lee, "Image representing using 2D Gabor wavelets," *IEEE Trans. Pattern Anal. Mach. Intell.* **18**(10), 959–971 (1996).
25. T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(7), 971–987 (2002).
26. P. F. Felzenszwalb et al., "Object detection with discriminatively trained part based models," *IEEE Trans. Pattern Anal. Mach. Intell.* **32**(9), 1627–1645 (2010).
27. "BMP image sequences for elliptical head tracking," (7 August 1998), http://www.ces.clemson.edu/~stb/research/headtracker/seq/.
28. "Xiph.org Video Test Media [derf's collection]," http://media.xiph.org/video/derf.
29. http://vision.ucsd.edu/~bbabenko/project_miltrack.shtml.
30. X. Wang, X. Ma, and E. Grimson, "Unsupervised activity perception in crowded and complicated scenes using hierarchical bayesian models," *IEEE Trans. Pattern Anal. Mach. Intell.* **31**(3), 539–555 (2009).



Keren Fu received his BSc degree in the major of Automation from Huazhong University of Science and Technology (HUST) in 2011. Currently, he is a PhD candidate at Shanghai Jiao Tong University (SJTU) in the Institute of Image Processing and Pattern Recognition under the supervision of Professor Jie Yang. His research interests include object detection, saliency detection, visual tracking, and machine learning.

Chen Gong received his BSc degree from East China University of Science and Technology (ECUST) in 2010. Currently he is a PhD candidate at Shanghai Jiao Tong University (SJTU) in the Institute of Image Processing and Pattern Recognition under the supervision of Professor Jie Yang. His research interests mainly include machine learning, object detection, and tracking.

Yu Qiao received his BEng and MEng degrees from Shanghai Jiao Tong University in 1991 and 1997, respectively, and PhD degree from National University of Singapore in 2004. He is currently an associate professor with the Institute of Image Processing and Pattern Recognition, Department of Automation, Shanghai Jiao Tong University. His research interests include medical image processing, machine learning, pattern recognition, signal processing, and data mining.

Jie Yang received his PhD from the Department of Computer Science, Hamburg University, Germany, in 1994. Currently, he is a professor at the Institute of Image Processing and Pattern recognition, Shanghai Jiao Tong University, China. He has led many research projects (e.g., National Science Foundation, 863 National High Tech. Plan), had one book published in Germany, and authored more than 200 journal papers.

Irene Yu-Hua Gu received the PhD degree in electrical engineering from the Eindhoven University of Technology, Eindhoven, The Netherlands, in 1992. From 1992 to 1996, she was a research fellow at Philips Research Institute IPO, Eindhoven, The Netherlands, and a post-doc at Staffordshire University, Staffordshire, United Kingdom, and lecturer at the University of Birmingham, Birmingham, United Kingdom. Since 1996, she has been with the Department of Signals and Systems, Chalmers University of Technology, Göteborg, Sweden, where she is currently a full professor. She was an associate editor for the *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans, and Part B: Cybernetics* from 2000 to 2005. She has been an associate editor with the *EURASIP Journal on Advances in Signal Processing* since 2005.