SSOCBT: A Robust Semisupervised Online CovBoost Tracker That Uses Samples Differently

Guorong Li, Qingming Huang, *Senior Member, IEEE,* Lei Qin, *Member, IEEE,* and Shuqiang Jiang, *Senior Member, IEEE*

***Abstract*—Most existing feature selection methods for object tracking assume that the samples in the previous frames are governed by the same distribution of the labeled samples obtained in the current frame and unlabeled samples collected in the next frame. However, according to our statistical analysis on very common videos, this assumption is not true in many scenarios. As a result, the selected features are not suitable for discriminating between the target from the background in the next frame. A tracking error accumulates and finally the drift problem happens. In this paper, we consider data distribution in tracking from a new perspective to adapt to target's and background's changes. We classify the samples into three categories: auxiliary samples (samples in the previous frames), target samples (samples collected in the current frame), and unlabeled samples (samples obtained in the next frame). To make the best use of them for tracking, we propose a novel semisupervised transfer learning approach that treats samples differently. Specifically, we assume that only target samples follow the same distribution as the unlabeled samples that we want to classify. Then, a novel and interesting semisupervised CovBoost method is developed utilizing the information provided by the three kinds of samples effectively when training the best strong classifier for tracking. Furthermore, we develop a new online updating algorithm for semisupervised CovBoost, making our tracker handle with significant variations of the tracked target and background successfully. Our experimental results demonstrate the advantages of treating samples differently during tracking. Our tracker outperforms state-of-the-art trackers on the benchmark datasets.**

***Index Terms*—Boosting, covariate shift, CovBoost, feature selection, object tracking, semisupervised learning, transfer learning.**

I. Introduction

OBJECT TRACKING has been a hot research topic for decades due to its wide applications in many domains. During the last few years, researchers have made great progress, e.g., [1]–[7]. However, there still exist many open problems facing complicated variations in target's appearance and motion, as well as background. In many works, such as [8] and [9], object tracking is considered as a local discrimination problem: foreground and background. Good features can be used to discriminate the target from background easily and accurately. Thus, feature selection is a challenging, but very important problem for object tracking.

To be adaptive to dynamic background and object's variations, many online feature selection methods, such as [8] and [10]–[12], are proposed. In [8], a variance ratio is used to evaluate feature's discriminative ability and the best k features are selected to describe the target's appearance. Meanwhile, due to its good performance and computation efficiency, online boosting methods [10], [11] have become the most popular feature selection algorithms for object tracking. For example, in [10], based on the obtained labeled samples, a classifier is trained and updated through online boosting. To alleviate the drift problem and make use of the prior knowledge, semisupervised methods, such as [14]–[17], are developed for tracking. Recently, multiple instance learning methods such as [18] and [19] are developed to resolve the ambiguities as to where to take positive updates during tracking. Unlike other semisupervised methods, in [20] a very interesting paradigm called "*P–N* learning," which is guided by positive and negative constraints, is proposed to restrict the labeling of unlabeled data. It could guarantee the improvement of the classifier during the learning process. However, we would like to point out that these approaches make a common assumption: in tracking, the samples they used for training follow the same distribution with the samples they aim to classify. However, when target's appearance or background varies greatly or continuously, the underlying data distribution also keeps changing and is not necessarily from independent and identical distributions [21]. In other words, there is a difference between previous and current samples. Perhaps, in human eyes, the difference is not very large, but in this case features selected using existing methods based on previous samples, such as [10] and [11], may generate incorrect results on the target samples, although they could lead to the minima classification error on the previous samples. As a result, tracking results become inaccurate and finally "drift" [22] happens. Fig. 1 shows a very good example.

Apparently, it is necessary to use previous samples, current samples, and unlabeled samples differently. In the field of

Manuscript received February 13, 2012; revised July 14, 2012; accepted August 12, 2012. Date of publication October 2, 2012; date of current version April 1, 2013. This work was supported in part by the National Basic Research Program of China's 973 Program 2009CB320906; in part by the National Natural Science Foundation of China under Grants 61025011, 61035001, 61133003, 61202322, and 61003165; and in part by the Beijing Natural Science Foundation under Grant 4111003. This paper was recommended by Associate Editor C.-W. Lin.

G. Li is with the University of the Chinese Academy of Sciences, Beijing 100049, China (e-mail: grli@jdl.ac.cn).

Q. Huang is with the University of the Chinese Academy of Sciences, Beijing 100049, China, and also with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: qmhuang@jdl.ac.cn).

L. Qin and S. Jiang are with the Key Laboratory of Intelligent Information Processing and the Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: lqin@jdl.ac.cn; sqjiang@jdl.ac.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2012.2221257

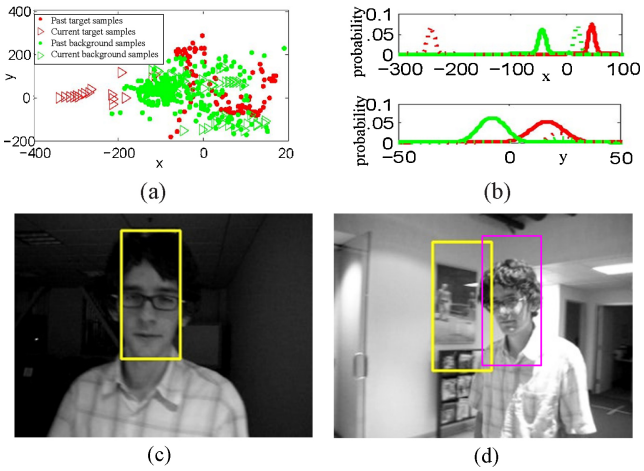

Fig. 1. Video example in which appearances of the tracking target and background vary significantly because of lightness and motion. We extract 100 Haar features for every sample and then reduce the dimension to 2 using PCA [13]. The red and green dots denote target samples and background samples collected from frame 3 to frame 142. The red and green triangles denote target samples and background samples collected in 10 frames around frame 146. Apparently, dot samples and triangle samples follow different distributions, as shown in (b). So training with dot samples, boosting method [10] generates an incorrect classification result on triangle samples. (a) Samples' distribution in the 2-D feature space. (b) Marginal probability density function of the samples. (c) Frame 2. The tracking target is labeled with a yellow rectangle. It is shown that training with dot samples, the boosting method [10], generates an incorrect classification result on triangle samples. (d) Frame 146. Tracking result of dot samples method (yellow rectangle) and our method (magenta rectangle).

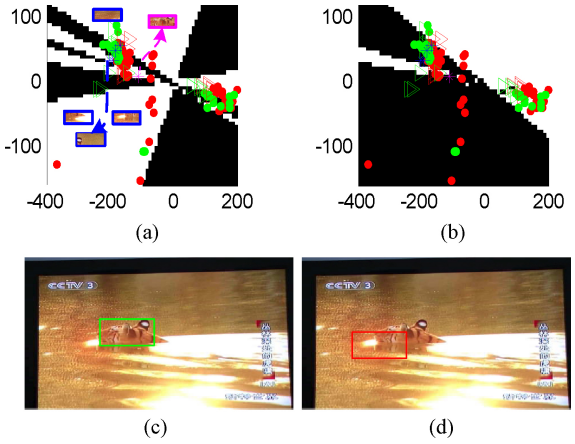

Fig. 2. Best viewed in color. A toy problem: the dots, triangles, and stars denote auxiliary samples, target samples, and unlabeled samples, which are collected in the previous frames, current frame, and next frame, respectively. The red ones are positive while the green ones are negative. The pink star represents the image patch in (a) with pink rectangle corresponding to foreground in the next frame. The blue stars denote image patches in (a) with blue rectangle corresponding to the background in the next frame. We can see that our proposed algorithm generates an effective decision plane, which classifies the pink star into the positive class and generates an accurate tracking result, as shown in (c). (a) Decision plane generated by semisupervised CovBoost. (b) Decision plane generated by Boost. (c) Tracking result corresponding to (a). (d) Tracking result corresponding to (b).

machine learning, a similar idea that distinguishes between training and test distributions has been a popular problem. Many works such as [23]–[25] have developed new learning algorithms for solving it. Meanwhile, in object detection, some works, such as [26] and [27], have developed efficient transfer learning methods [23] to deal with cases in which the training and test data follow different distributions. However, the covariate shift phenomenon has not yet been studied for object tracking and it is conceivable in many situations such as videos containing fast-moving targets or motion blur, low frame-rate videos, and home videos that are self-recorded with hand-held cameras. As a result, it is necessary to investigate this case for tracking. In this paper[1], we first make a careful study of the target's and background's distribution in common videos. Then, we propose to consider data's distribution from a new and more reasonable angle and investigate the "covariate shift" problem in the field of tracking. In more specific terms, we classify the samples in object tracking into three categories: labeled auxiliary samples, labeled target samples, and unlabeled target samples as shown in Fig. 4. We assume that the labeled auxiliary samples and target samples are under covariate shift. To select the best features for discriminating the unlabeled target samples and obtain tracking results, we develop a semisupervised online boosting method. Experimental results demonstrate that it can alleviate "drift" to some extent and thus could achieve superior performance. See Fig. 2 for a concrete example.

The rest of this paper is organized as follows. In Section II we quantize the difference between samples' distribution in several common videos to verify our assumption. We then provide a brief introduction to CovBoost and propose a new method for solving it in Section III. Section IV details the derivations of the proposed semisupervised CovBoost classifying them as unlabeled target samples. The online learning algorithm for tracking is then elaborated on in Section V. Experimental results on various test videos are reported in Section VI. Finally, we conclude this paper in Section VII. The notations for frequently used variables in the following part are provided in Table I.

[1]This paper is an extension of [28].

TABLE I

NOTATIONS FOR SOME FREQUENTLY USED VARIABLES

Notation	Description
x, y	Observation and label of a sample ($y = 1$, positive sample; $y = -1$, negative samples)
f_i, h_i	Haar feature and the corresponding weak classifier
M, N_w	The number of selectors and the number of weak classifiers
$h_{s_m}(\cdot)$, α_{s_m}	The selected weak classifier and its corresponding weight, $H = \sum_{m=1}^{M} \alpha_{s_m} h_{s_m}$
$\Im$, X^L, U	Auxiliary data set, labeled target data set and unlabeled data set
z_i, ℓ_i	Observation and label of the sample that is in auxiliary data set
x_i, y_i	Observation and label of the sample that is in labeled target data set
u_i	Observation of the sample that is in unlabeled data set
n_a, n, n_u	Number of samples in auxiliary data set, Number of samples in labeled target data set, Number of samples in unlabeled data set
$\mu^i_{o,t}$, $\sigma^i_{o,t}$	At time t, mean and covariance of the positive samples with respect of f_i
$\mu^i_{b,t}$, $\sigma^i_{b,t}$	Mean and covariance of the negative samples with respect of f_i

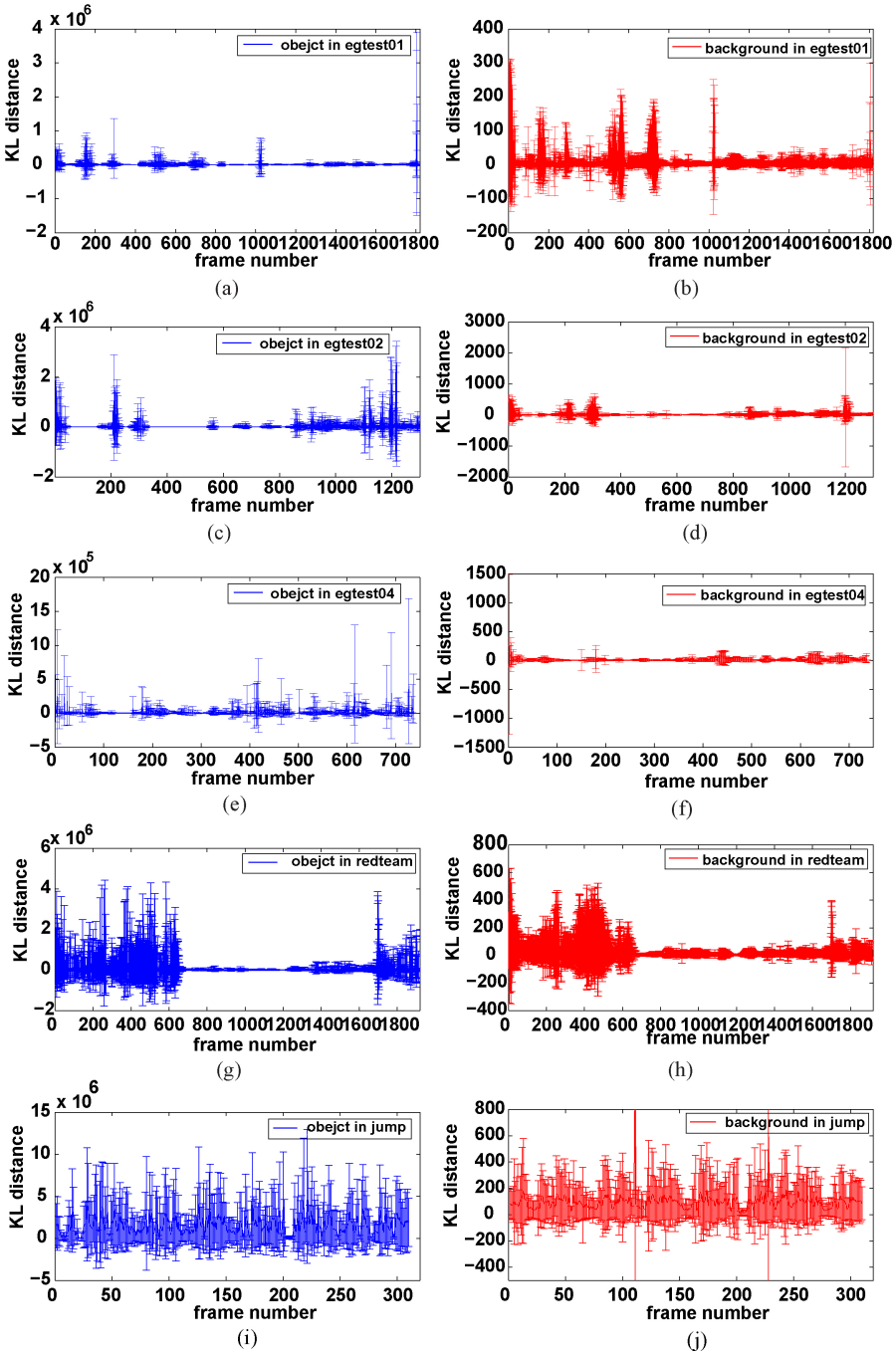

Fig. 3. Distance between samples' distributions in adjacent frames of five test videos. The mean and variance of $\{D_t^i, i = 1, 2, \ldots, 1000\}$ are provided in every frame. (a) Object in *egtest01*. (b) Object in *egtest01*. (c) Background in *egtest02*. (d) Background in *egtest02*. (e) Object in *egtest04*. (f) Object in *egtest04*. (g) Object in *Redteam*. (h) Background in *Redteam*. (i) Object in *Jump*. (j) Background in *Jump*.

II. Difference Between Samples' Distribution in Adjacent Frames

We observe that in most videos object or background usually changes. Intuitively, their probability density functions should vary too. In every frame, we collect target samples and background samples, and each sample is represented by the observation x and its label y. $y = 1$ denotes that the sample belongs to target class while $y = -1$ means the sample belongs to the background. The observation x is combined with 1000 Haar features $f_i (i = 1, 2, \ldots, 1000)$. In frame t, we assume $p_t(f_i|y = 1)$ and $p_t(f_i|y = -1)$ follow the Gaussian distribution $N(\mu_{o,t}^i, \sigma_{o,t}^i)$ and $N(\mu_{b,t}^i, \sigma_{b,t}^i)$. The Kullback–Leibler (KL) divergence [29] is usually used as the distance metric on the space of probability distributions, but it is not symmetric.

In past years, several kinds of KL distances, such as [29]–[32], were proposed. In our experiment, we simply use (1) to measure the difference of those distributions in adjacent frames

$$D_t^i = D_{KL}(p_t(f_i|y) \parallel p_{t+1}(f_i|y)) + D_{KL}(p_{t+1}(f_i|y) \parallel p_t(f_i|y)) \tag{1}$$

where

$$D_{KL}(p_t(f_i|y) \parallel p_{t+1}(f_i|y)) = \int_{f_i} p_t(f_i|y) \log \frac{p_t(f_i|y)}{p_{t+1}(f_i|y)} df_i$$

is the KL divergence of $p_t(f_i|y)$ and $p_{t+1}(f_i|y)$. Substitute $p_t(f_i|y=1) = \frac{1}{\sqrt{2\pi\sigma^i_{o,t}}} \exp\left(-\frac{(x-\mu^i_{o,t})^2}{2(\sigma^i_{o,t})^2}\right)$ and $p_t(f_i|y=-1) = \frac{1}{\sqrt{2\pi\sigma^i_{b,t}}} \exp\left(-\frac{(x-\mu^i_{b,t})^2}{2(\sigma^i_{b,t})^2}\right)$ into (1), we could derive that

$$D^i_t = -1 + \sum_{c\in\{o,b\}} \frac{(\sigma^i_{c,t})^2 + (\mu^i_{o,t} - \mu^i_{b,t})^2}{2(\sigma^i_{c,t})^2}. \tag{2}$$

From the public database, we analyze the variation of the target's and background's distributions. Here, we mainly report five videos[2] recorded under very common situations. We compute D^i_t in those test videos to analyze the difference between samples' distributions in adjacent frames. The result is shown in Fig. 3. We can see that in those videos the target in *Jump* varies most violently, because the target is skipping and there is significant motion blur. Meanwhile, the KL distance of the target in *egtest01* is smaller than KL distance of the target in the other four videos, because only sunlight and target's pose vary and they have little effects on the Haar feature. From those results, we could easily find that samples' distribution varies greatly, although sometimes according to human vision, the variations of the object and background are not very large (e.g., *egtest01* and *egtest04*). In our paper, we simply assume that data's distributions in previous frames and current frame are under "covariate shift."

III. Covariate Shift and CovBoost Algorithm

Given a labeled auxiliary set $\Im = \{\langle z_j, \ell_j\rangle\}_{j=1}^{n_a}$, where z_j is drawn from auxiliary distribution $p_a(Z)$ and target training set $\chi^L = \{\langle x_i, y_i\rangle\}_{i=1}^{n}$, where x_i is drawn from target distribution $p_t(X)$, let $D = \{z_1, z_2, \ldots, z_{n_a}, x_1, x_2, \ldots, x_n\}$. If $p(\ell|Z = z) = p(Y|X = z)$ for all $z \in D$, but $p_a(Z) \neq p_t(X)$, then the difference between the two datasets is referred to as "covariate shift" [23].

Then, the loss function on those labeled samples with respect to target data distribution $p(z_j, \ell_j|\langle z_j, \ell_j\rangle \in \chi^L)$ becomes

$$L_c(H, \Im, \chi^L) \approx \sum_{i=1}^{n} \exp\{-2y_i H(x_i)\} + \sum_{j=1}^{n_a} \gamma(z_j, \ell_j) \exp\{-2\ell_j H(z_j)\} \tag{3}$$

where $\gamma(z_j, \ell_j) = \frac{p_t(z_j)}{p_a(z_j)}$ and $H = \sum_{m=1}^{M} \alpha_{s_m} h_{s_m}$.

CovBoost aims to form a strong classifier $H(\cdot)$ with weak classifiers $\{h_k(\cdot), k = 1, 2, \ldots, N_w\}$ by minimizing the extended loss function $L_c(H, \Im, \chi^L)$. Following the derivation of AdaBoost [34], in the mth iteration, we first select $h_{s_m}(\cdot)$ in a greedy method and then obtain α_{s_m} by solving the equation $\frac{\partial H_m}{\partial \alpha_{s_m}} = 0$ $(H_m = \sum_{i=1}^{m} \alpha_{s_m} h_{s_m})$

$$h_{s_m}(\cdot) = \text{argmin}_{h(\cdot)} \left\{ \sum_{i=1}^{n} w^m_i \delta(h(x_i), y_i) + \sum_{j=1}^{n_a} \varpi^m_j \delta(h(z_j), \ell_j) \right\} \tag{4}$$

$$\alpha_{s_m} = \frac{1}{4} \ln \frac{1-\epsilon_m}{\epsilon_m} \tag{5}$$

where

$$w^m_i = \exp\{-2y_i H_{m-1}(x_i)\}$$
$$\varpi^m_j = \gamma(z_j, \ell_j) \exp\{-2\ell_j H_{m-1}(z_j)\}$$
$$\epsilon_m = \frac{\Sigma_{h(x_i)\neq y_i} w^m_i + \Sigma_{h(z_j)\neq \ell_j} \varpi^m_j}{\Sigma^n_{i=1} w^m_i + \Sigma^{n_a}_{j=1} \varpi^m_j}.$$

In [26], $\gamma(x, y)$ is proved to be equivalent with $\frac{p(<x,y>\in\chi^L|x,y)}{p(\langle x,y\rangle\in\Im|x,y)} \frac{p(\langle x,y\rangle\in\Im)}{p(<x,y>\in\chi^L)}$. Unlike the solving method in [26], using the boosting method, we first learn two classifiers $H_t(x)$ and $H_a(x)$ on χ^L and $\Im$, respectively. Then, conditional distributions $p(\langle x, y\rangle \in \chi^L|x, y)$ and $p(\langle x, y\rangle \in \Im|x, y)$ could be modeled as $\frac{1}{1+\exp\{-yH_t(x)\}}$ and $\frac{1}{1+\exp\{-yH_a(x)\}}$, respectively. In short, $\frac{p(\langle x,y\rangle\in\Im)}{p(\langle x,y\rangle\in\chi^L)}$ is denoted by $r_{a,t}(x, y)$. Then

$$\gamma(x, y) = r_{a,t}(x, y) \frac{1 + \exp\{-yH_a(x)\}}{1 + \exp\{-yH_t(x)\}}. \tag{6}$$

IV. Semisupervised CovBoost for Feature Selection

From the loss function of CovBoost, we can see that unlabeled samples are not used. However, there are generally many unlabeled samples waiting to be classified. Although they do not have labels, they still could provide very useful information. Usually, samples with similar observations should share the same label. This is referred to as data consistency in many works such as [35] and it was proven to be helpful for tracking in [16] and [19]. So, in the proposed tracker, we want to develop semisupervised CovBoost to combine the information implied in the unlabeled data with the loss function $L_c(H, \Im, \chi^L)$.

A. Semisupervised CovBoost

Considering unlabeled set $U = \{u_1, u_2, \ldots, u_{n_u}\}$ with $S(\cdot, \cdot)$ denoting the similarity function of two samples, the loss between unlabeled samples and labeled samples $L_u(H, \Im, \chi^L, U)$ is defined as

$$L_u(H, \Im, \chi^L, U) = \underbrace{\frac{\sum_{x_i,u_j} S(x_i, u_j) \exp\{-2y_i H(u_j)\}}{nn_u}}_{\text{the first item}} + \underbrace{\frac{\sum_{u_i,u_j} S(u_i, u_j) D(H, u_i, u_j)}{n^2_u}}_{\text{the second item}} + \underbrace{\frac{\sum_{z_i,u_j} \gamma(z_i, l_i) S(z_i, u_j) \exp\{-2\ell_i H(u_j)\}}{n_a n_u}}_{\text{the third item}} \tag{7}$$

[2] *egtest01, egtest02, egtest04*, and *Redteam* are from CMU_VIVD [33]. *Jump* is from [20]. For *egtest04*, only frame 0 to frame 787 are used.

where $D(H, u_i, u_j) = \frac{\exp\{H(u_i)-H(u_j)\}+\exp\{H(u_j)-H(u_i)\}}{2}$. The first and third items measure the data inconsistency between labeled data and unlabeled data, and the second item represents that between unlabeled samples. Then, we design the final loss function $L(H, \Im, \chi^L, U)$ by extending $L_c(H, \Im, \chi^L)$ to include $L_u(H, \Im, \chi^L, U)$

$$L(H, \Im, \chi^L, U) = L_c(H, \Im, \chi^L) + L_u(H, \Im, \chi^L, U). \quad (8)$$

Simulating the underlying of AdaBoost, we first minimize the loss $L(\cdot)$ only with respect to weak classifiers. After the best weak classifier $h_{s_m}(\cdot)$ is determined, we then optimize its weight α_{s_m}. Through the first-order Taylor expansion of $L(\cdot)$ at $H_{m-1}(\cdot)$, $h_{s_m}(\cdot)$ can be selected as

$$h_{s_m}(\cdot) = \text{argmin}_{h(\cdot)} \begin{bmatrix} \sum_{i=1}^{n} w_i^m \delta(h(x_i), y_i) \\ +\sum_{j=1}^{n_a} \varpi_j^m \delta(h(z_j), \ell_j) \\ -\frac{1}{n_u}\sum_{j=1}^{n_u}(p_m(u_j) - q_m(u_j))h(u_j) \end{bmatrix} \quad (9)$$

$$\alpha_{s_m} = \frac{1}{4}\ln\left\{\frac{1-\epsilon_m^{'}}{\epsilon_m^{'}}\right\} \quad (10)$$

where

$$\begin{aligned} p_m(u_j) = \frac{1}{n}\exp\{-2H_{m-1}(u_j)\}\sum_{y_i=1} S(x_i, u_j) \\ +\frac{1}{n_a}\exp\{-2H_{m-1}(u_j)\}\sum_{\ell_i=1}\gamma(z_i, \ell_i)S(z_i, u_j) \\ +\frac{1}{n_u}\sum_{i=1}^{n_u} S(u_i, u_j)\exp\{H(u_i) - H(u_j)\} \end{aligned} \quad (11)$$

denoting the confidence that u_i is positive and

$$\begin{aligned} q_m(u_j) = \frac{1}{n}\exp\{2H_{m-1}(u_j)\}\sum_{y_i=-1} S(x_i, u_j) \\ +\frac{1}{n_a}\exp\{2H_{m-1}(u_j)\}\sum_{\ell_i=-1}\gamma(z_i, \ell_i)S(z_i, u_j) \\ +\frac{1}{n_u}\sum_{i=1}^{n_u} S(u_i, u_j)\exp\{H(u_j) - H(u_i)\} \end{aligned} \quad (12)$$

representing the confidence that u_i is negative. Let $\epsilon_m^u = \sum_{h(u_j)=-1} p_m(u_j) + \sum_{h(u_j)=1} q_m(u_j)$ and it can be treated as the classification error on unlabeled samples. Then, the classification error $\epsilon_m^{'} = \frac{\sum_{h(x_i)\neq y_i}\omega_i^m + \sum_{h(z_j)\neq \ell_j}\varpi_j^m + \epsilon_m^u}{\sum_{x_i}\omega_i^m + \sum_{z_j}\varpi_j^m + \sum_{u_j}(p_m(u_j)+q_m(u_j))}$.

In summary, the semisupervised CovBoost minimizes the loss function that treats labeled samples differently and takes labeled samples and unlabeled samples into account. When no unlabeled data are used ($U = \{\}$), (9) and (10) reduce to CovBoost. Besides, if $H_t(\cdot) = H_a(\cdot)$, which means that labeled auxiliary samples and labeled target samples follow the same distribution, and (9) and (10) degenerate to the well-known SemiBoost formulas.

B. Boosting the Similarity Function

Generally, any similarity measure (e.g., sum of squared difference [36], normalized cross correlation [37], and Bhattacharyya distance [38]) could be used for defining $S(\cdot, \cdot)$. However, as the number of target samples and auxiliary samples is fairly large, in order to compute the similarity efficiently, similar to [16], we boost the similarity according to [39] and [40]. Meanwhile, $\gamma(z_i, \ell_i)$ is not a constant and computing the third item in (7) would be time consuming. Therefore, we want to develop a new similarity function $\widetilde{S}(\cdot, \cdot)$ meeting the following requirements:

$$\begin{cases} \widetilde{S}(x_i, u_j) \approx S(x_i, u_j) \\ \widetilde{S}(z_i, u_j) \approx \gamma(z_i, u_i)S(z_i, u_j) \\ \widetilde{S}(u_i, u_j) \approx S(u_i, u_j). \end{cases} \quad (13)$$

In other words, when we learn similarity function, target samples and auxiliary samples are used differently. We train a prior classifier H^P on auxiliary dataset and target dataset through minimizing loss function L_c defined by (3). Then

$$p(y = 1|x = u_i) = \frac{\exp\{H^P(u_i)\}}{\exp\{H^P(u_i)\} + \exp\{-H^P(u_i)\}}. \quad (14)$$

$p(y = 1|x = u_i)$ is used as the similarity between u_i and positive target or auxiliary samples. Similarly, $p(y = -1|x = u_i)$ is used as the similarity between u_i and negative samples

$$\widetilde{S}(x_i, u_j) = \frac{\exp\{y_i H^P(u_j)\}}{\exp\{H^P(u_j)\} + \exp\{-H^P(u_j)\}}. \quad (15)$$

In the current frame, the number of unlabeled samples could be very large, so we simply consider $|U| \to \propto$ and the third addend in both (11) and (12) converge to zero. Substituting $\widetilde{S}(\cdot, \cdot)$ into (11) and (12), $p_m(u_j)$ and $q_m(u_j)$ could be approximated by $\tilde{p}_m(u_j)$ and $\tilde{q}_m(u_j)$, respectively

$$\tilde{p}_m(u_j) = (\frac{n^+}{n} + \frac{n_a^+}{n_a})\frac{\exp\{-H_{m-1}(u_j)\}\exp\{H^P(u_j)\}}{\exp\{H^P(u_j)\} + \exp\{-H^P(u_j)\}} \quad (16)$$

$$\tilde{q}_m(u_j) = (\frac{n^-}{n} + \frac{n_a^-}{n_a})\frac{\exp\{H_{m-1}(u_j)\}\exp\{-H^P(u_j)\}}{\exp\{H^P(u_j)\} + \exp\{-H^P(u_j)\}} \quad (17)$$

where n^+ and n^- denote the number of positive samples and negative samples in χ^L, n_a^+, and n_a^- represent the number of positive samples and negative samples in $\Im$. In our experiments, $n^+ = n^- = 0.5n$ and $n_a^+ = n_a^- = 0.5n_a$.

The details of our online updating algorithm for semisupervised CovBoost are displayed in Algorithm 1.

V. Online Learning for Object Tracking

During tracking, we store examples from several previous frames and use them as auxiliary samples. Meanwhile, once we obtain tracking result on the current frame, we recollect target samples according to the result. First, training with auxiliary samples and target samples, respectively, using the boosting method, we obtain classifier $H_a(\cdot)$ and $H_t(\cdot)$. Then, based on $H_a(\cdot)$ and $H_t(\cdot)$ and training with auxiliary and target samples together via CovBoost method, a classifier $H^P(\cdot)$ is obtained. Based on $H^P(\cdot)$ and continue to train with the

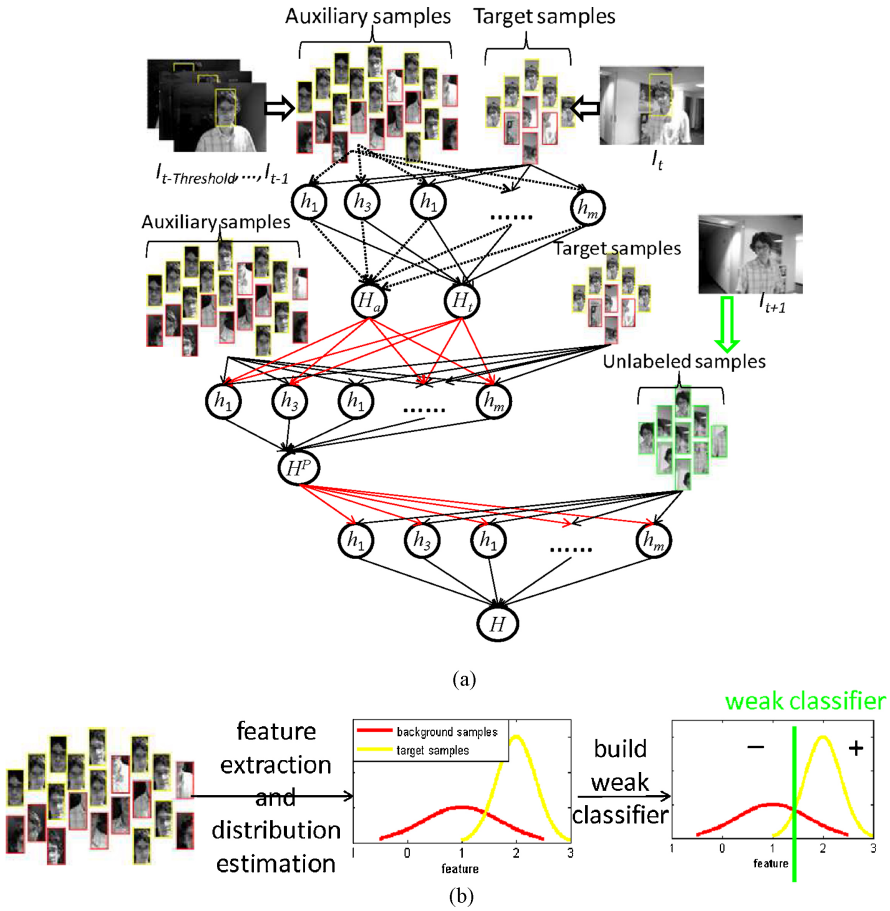

Fig. 4. Framework of our semisupervised online CovBoost tracker. (a) Flowchart for how to train the strong classifier H with semisupervised CovBoost. In previous frames, we collected auxiliary samples and trained classifier H_a using the boosting method. In the current frame, we collect target samples and train H_t using boosting too. Then, we use auxiliary and target samples to train H^p using CovBoost for OCBT tracker. In the next frame, we collect unlabeled samples, and combine them with H^p, auxiliary target samples to train the final classifier H for SSOCBT. (b) Illustration of how to construct a weak classifier.

unlabeled samples using semisupervised CovBoost algorithm, the final classifier $H(\cdot)$ is developed. As the tracking procedure goes on, the auxiliary sample, target sample, and unlabeled sample are updated and then our tracker keeps evolving with them. Fig. 4 shows the framework of our tracking method.[3] We can see that for every frame, the computation complexity is $O((|\chi|+|\Im|+|U|)MN_w)$. Here, we want to point out that there are two crucial differences between our method and online boosting. One is that we assume target samples follow different distribution with auxiliary samples while the other boosting trackers treat them equally. The other is that we make use of unlabeled samples in the next frame.

A. Implementation

We implement our tracking algorithm with C++ code. The executable program is available from the webpage http://www.jdl.ac.cn/user/grli/SSOCBT/index.htm. Some related details that readers will care about are described specifically.

Feature. Although many features such as color, texture, and HoG could be used for tracking, we select the Haar-based feature in our experiment because of its excellent performance reported in many works [11], [41]. Moreover, resorting to integral image, the Haar feature value is very computationally efficient.

Weak Classifier. Provided the Haar feature value f_i ($i = 1, 2, \ldots, 1000$), the weak classifiers are defined as $h(f_i) = sign(p(f_i \in C_o|f_i) - p(f_i \in C_b|f_i))$. C_o and C_b denote object and background class, respectively. We model $p(f_i \in C_o|f_i)$, $p(f_i \in C_b|f_i)$ with the Gaussian distribution $N(\mu_o^i, \sigma_o^i)$ and $N(\mu_b^i, \sigma_b^i)$. During tracking, when receiving the new n labeled samples with feature value $\{\langle f_i^1, y_1\rangle, \langle f_i^2, y_2\rangle, \ldots, \langle f_i^n, y_n\rangle\}$, the parameters (μ_o^i, σ_o^i) are updated according to the following equation. It is similar for μ_b^i, σ_b^i

$$\begin{cases} \mu_o^i = \tau\mu_o^i + (1-\tau)\dfrac{\sum_{y_j=1} f_i^j}{\sum_{j=1}^{n}(y_j == 1)} \\ \sigma_o^i = \tau\sigma_o^i + (1-\tau)\dfrac{\sum_{y_j=1}(f_i^j - \mu_o^i)^2}{\sum_{j=1}^{n}(y_j == 1)} \end{cases} \tag{18}$$

where τ represents the learning rate.

In our implementation, the number (N_w) of weak classifiers is 1000 and the number (M) of selectors is 100.

[3]The detail procedure of the proposed tracking method is provided at http://www.jdl.ac.cn/user/grli/SSOCBT/index.htm.

Algorithm 1 Semisupervised Online CovBoost (SSOCB)

Input: $H_t(\cdot)$, $H_a(\cdot)$, $H^P(\cdot)$ and training examples: $\{\langle x_1, y_1\rangle, \ldots, \langle x_N, y_N\rangle\}$.
Output: Strong classifiers $H(\cdot)$

$s^w_{m,k} = 0, s^c_{m,k} = 0, \widetilde{\lambda}_m = 1, m = 1, 2, \ldots, M; k = 1, 2, \ldots, N_w$;
/* $s^c_{m,k}$ denotes the sum of the weight of samples that are classified correctly and $s^w_{m,k}$ is that of the samples misclassified */
; **for** $i = 1; i \leq N; i++$ **do**
  for $m = 1; m \leq M; m++$ **do**
    if x_i *is target sample* **then**
      $y_m = y_i$, $\lambda_m = \widetilde{\lambda}_m$;
    else
      if x_i *is auxiliary sample* **then**
        $y_m = y_i$, $\lambda_m = r_{a,t}(x_i, y_m)\widetilde{\lambda}_m$;
      else
        $y_m = sign\,(\tilde{p}_m(x_i) - \tilde{q}_m(x_i))$,
        $\lambda_m = |\tilde{p}_m(x_i) - \tilde{q}_m(x_i)|$;
      end
    end
    for $k = 1; k < N_w; k++$ **do**
      if $y_m \neq h_{m,k}(x_i)$ **then**
        $s^w_{m,k} += \lambda_m$;
      else
        $s^c_{m,k} += \lambda_m$;
      end
      $\varepsilon_{m,k} = \frac{s^c_{m,k}}{s^c_{m,k}+s^w_{m,k}}$;
    end
    $s_m = \arg_k \min \varepsilon_{m,k}$;
    $\alpha_{s_m} = \ln \frac{1-\varepsilon_{m,s_m}}{\varepsilon_{m,s_m}}$;
    $\widetilde{\lambda}_m = \widetilde{\lambda}_m \exp\{-2y_m\alpha_{s_m}h_{s_m}(x_i)\}$;
  end
  $\alpha_{s_m} = \frac{\alpha_{s_m}}{\sum_{m=1}^{M}\alpha_{s_m}}$;
end
return $H(x) = \sum_{m=1}^{M}\alpha_{s_m}h_{s_m}(x)$;

Memory. As shown in Fig. 4, we need to store auxiliary samples for updating. To reduce the computational complexity, we store $h_i(z)$ ($i = 1, 2, \ldots, N_w$) instead of z. This is because when we perform updating with samples, we need to evaluate $h_i(\cdot)$. The maximal size of auxiliary samples is set as 1000 (*Threshold* = 1000). It is a compromise between tracking speed and accuracy.

Samples Collection. After we obtain tracking result in the current frame I_t, we collect labeled samples as shown in Fig. 5(a). In frame I_{t+1}, we collect unlabeled data using the dense sampling method (overlap ratio is 0.99) in the search region and Fig. 5(b) provides a graphic representation.

VI. Experimental Results and Analysis

In this section, we design experiments to demonstrate the superior properties of the proposed tracker. To analyze the effectiveness of unlabeled data, we also design online CovBoost tracker considering the loss function defined by (3). We compare with related works of state-of-the-art trackers [4]–[6], [10], [16], [18], and [20], whose codes and parameters are provided on their websites. The description of every tracker and its abbreviation are provided in Table II. For simplicity, we use the abbreviation of every tracker at the following parts.

TABLE II
Description of the Trackers Appearing in Our Experimental Section

Abbreviation	Description of the Tracker
OBT	Online boosting tracker [10]
SSOBT	Semisupervised online boosting tracker [16]
MILT	Online multiple instance learning tracker [18]
TLD	Tracking-learning-detection tracker [42]
ILT	Incremental learning tracker [5]
FF	Fragment-based tracker [4]
SRPT	Particle filter based on sparse representation [6]
OCBT	Our online CovBoost tracker
SSOCBT	Our semisupervised online CovBoost tracker

All of those trackers except for TLD[4] do not adapt to scale changes. We compare OCBT, SSOCBT with OBT, and SSOBT, respectively, to demonstrate the effectiveness of our novel assumption. Furthermore, we compare our tracker with four trackers on various test videos described in Table III to prove that our tracker could alleviate drift to some extent.

A. Parameter Tuning for $r_{a,t}$

We know that $p(\langle x, y\rangle \in \Im)$ and $p(\langle x, y\rangle \in \chi^L)$ denote the probability that the sample $\langle x, y\rangle$ is drawn from auxiliary data $\Im$ and the probability that $\langle x, y\rangle$ comes from target data χ^L, respectively. Using maximum likelihood, $\frac{|\Im|}{|\chi^L|+|U|}$ is the estimation for $r_{a,t}(x, y)$. However, the size of $|\Im|$, χ^L or U is set manually in tracking. So, we simply consider $r_{a,t}(x, y)$ to be a constant $r_{a,t}$ and test different values for it to get an appropriate one. Fig. 6 shows the tracking error corresponding to different values of $r_{a,t}$. Intuitively, a larger $r_{a,t}$ means the auxiliary samples are more reliable. Fig. 6 shows the tracking performance with different $r_{a,t}$ ($\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0\}$).

Experimental results are evaluated by overlap-criterion[5] and the position error computed according to

$$overlap(g_f, o_f) = \frac{area\left(g_f \cap o_f\right)}{area\left(g_f \cup o_f\right)} \quad (19)$$

$$error(g_f, o_f) = \|center(g_f) - center(o_f)\|_2 \quad (20)$$

where g_t denotes ground truth, o_f is the tracking result, and $\|\cdot\|_2$ is L_2 norm. In our experiment, the tracked target is denoted by a rectangle and both g_f and o_f denote rectangles. If the overlap in a frame is larger than zero, we consider the object to be tracked successfully in that frame. We define successful frame rate (SFR) as the rate of frames in which the

[4]The learning part of TLD is P-N learning [20].
[5]http://www.pascal-network.org/challenges/VOC/voc2009

(a) (b)

Fig. 5. (a) Illustration of how to obtain labeled samples based on tracking result. (b) Instructions for collecting unlabeled data in frame I_{t+1}.

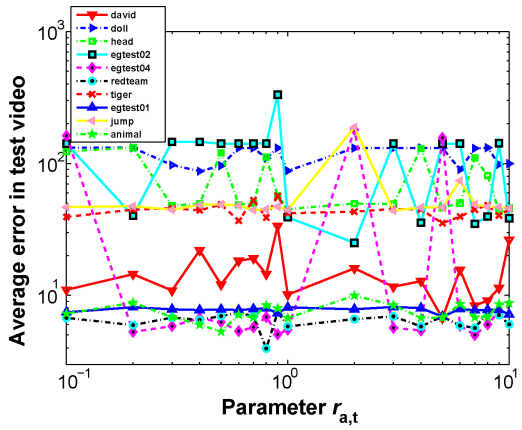

Fig. 6. Tracking performances as average error of tracking results and ground truth independent of parameter $r_{a,t}$.

TABLE III
INFORMATION ABOUT OUR TEST VIDEOS

Data Set	Video Name	Total Frames
CMU_VIVD	*egtest01*	1831
	egtest02	1301
	egtest04	800
	Redteam	1919
Self-recorded	*Tiger*	200
	Head	300
	Doll	200
TRECVID2010	*Store*	43
Other	*David* [5]	461
	Motor [20]	2665
	Jump [20]	313
	Animal [43]	71

target is tracked successfully and the total number of frames in the test video. This is

$$SFR = \frac{number\ of\ successful\ frames}{the\ number\ of\ total\ frames\ in\ the\ test\ video}. \quad (21)$$

Surprisingly, the successful frame rate is not sensitive to the value of $r_{a,t}$. This is because $\gamma(x, y) = r_{a,t} \frac{1+\exp\{-yH_a(x)\}}{1+\exp\{-yH_t(x)\}}$ and it is mainly effected by $\frac{1+\exp\{-yH_a(x)\}}{1+\exp\{-yH_t(x)\}}$. However, as shown in Fig. 6, the average tracking errors in every test video changes with the value of $r_{a,t}$. Intuitively, a large $r_{a,t}$ means training $H(\cdot)$ mainly with auxiliary sample while a small $r_{a,t}$ means training $H(\cdot)$ mainly with target samples. As shown in Fig. 6, the curves can be divided into three categories: relatively smooth curves (e.g., *egtest01*), fluctuating curves (e.g., *David, Head, Animal*), and smooth-fluctuating curve (e.g., *egtest02, egtest04, Jump, Redteam, Tiger, Doll*). In the videos (e.g., *egtest01*) in which the target and background vary little and the distributions of auxiliary sample, target sample, and unlabeled sample have little difference, so the value of $r_{a,t}$ has little affect on the tracking result. While for videos *Jump, egtest02, egtest04, Redteam, Tiger, Doll*, in which the target and background change relatively smooth, $r_{a,t}$ tends to be a constant, so some value of $r_{a,t}$ could lead to better tracking performance. As a result, a part of the curve is flat. In videos, such as *David*, *Head*, and *Animal*, the target or background varies sharply, and the $r_{a,t}$ keeps changing. Therefore, it is hard to achieve a good performance with a fixed $r_{a,t}$ and curves that fluctuate acutely. In other words, fixed $r_{a,t}$ could not generate the best results, so in the future, we will concentrate on a more sophisticated algorithm for estimating $r_{a,t}$ automatically.

In the following comparison experiments, we set $r_{a,t} = 1$ for all test videos.

B. Effectiveness of Our Covariate Shift Assumption

As in our experiments, the only difference between OCBT and OBT is that their assumptions on samples' distribution are different. The test video is *Jump* [20], which is recorded with a moving camera. The boy is skipping rope. As a result, his face is often blurry. In human eyes, the appearance of his face may not change significantly, but actually the differences between faces in adjacent frame are considerable as shown in the fourth image in Fig. 7. However, OBT and SSOBT begin to drift soon after tracking because of motion blur. The incorrect updates severely degrade their performances. With the covariate shift assumption on samples' distribution, OCBT and SSOCBT succeed in tracking the face through the whole video. Fig. 8 displays quantitative comparison results using overlap-criterion and the position error computed according to (19) and (20). From the result, we can see that the improvement brought by our assumption is apparent and significant.

C. Comparison With Other Trackers

To show the superiority of SSOCBT over other trackers, we perform experiments using OCBT, SSOCBT, MILT, TLD, OBT, and SSOBT on a number of video sequences.

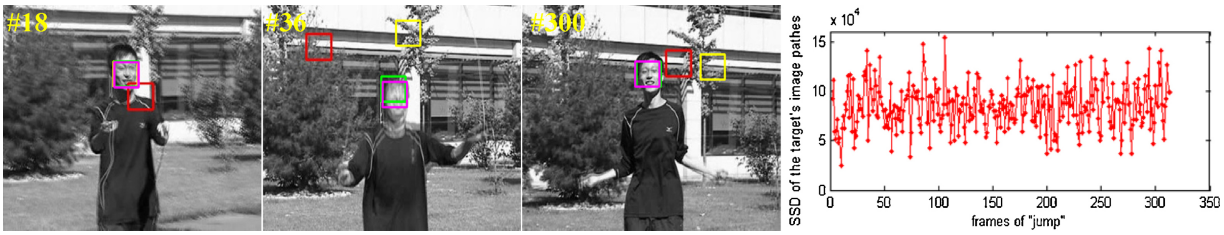

Fig. 7. Qualitative comparison results of SSOCBT (in magenta), OCBT (in green), OBT (in red), SSOBT (in yellow) in the *Jump* sequence. The last image is the SSD of the face's image patches in adjacent frames.

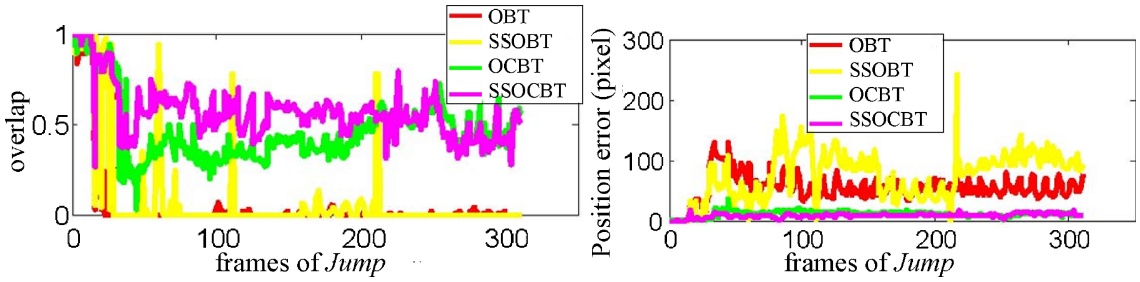

Fig. 8. Quantitative comparison results of SSOCBT, OCBT, OBT, SSOBT in the *Jump* sequences.

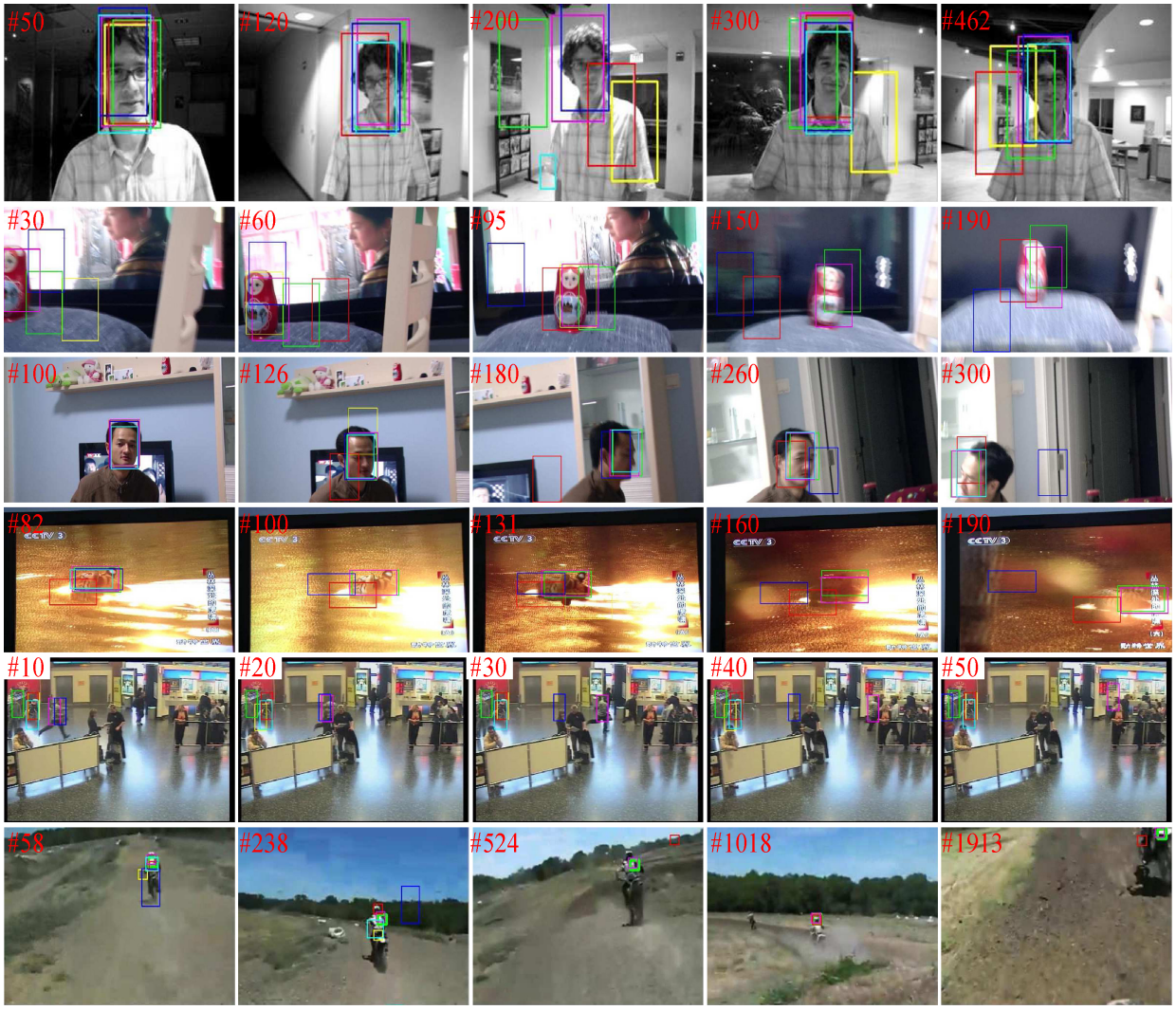

Fig. 9. Qualitative comparison results of SSOCBT (in magenta), OCBT (in green), OBT (in red), SSOBT (in yellow), MILT (in blue), and TLD (in cyan) on six test video sequences. In some frames, SSOBT or TLD does not provide tracking results, because it infers that the target disappears in those frames.

Fig. 10. Qualitative comparison results of SSOCBT (in magenta), OCBT (in green), OBT (in red), SSOBT (in yellow), MILT (in blue), and TLD (in cyan) on six test video sequences. In some frames, SSOBT or TLD does not provide tracking results, because it infers that the target disappears in those frames.

TABLE IV

SFR OF NINE TRACKERS IN TEST VIDEOS AND THE AVERAGE POSITION ERRORS OF TRACKING RESULTS IN SUCCESSFUL FRAMES

Sequence	*David*		*Doll*		*Head*		*Tiger*		*Store*		*Motor*	
	SFR	error	SFR	error	SFR	error	SFR	error	SFR	error	SFR	error
SSOCBT	100%	10.86	100%	77.74	100%	45.51	100%	58.88	100%	8.62	41.0%	20.76
OCBT	87.1%	10.49	76.3%	131.60	100%	53.94	100%	61.21	11.6%	22.63	27.2%	17.70
OBT [10]	71.3%	19.48	63.2%	121.37	67.1%	63.14	100%	136.86	11.6%	22.63	19.0%	17.98
SSOBT [16]	74.8%	53.79	73.7%	71.29	39.9%	28.40	40.6%	49.96	9.3%	19.93	10.9%	20.44
MILT [18]	100%	11.27	53.2%	228.24	87.4%	27.47	78.7%	134.92	44.2%	16.21	4.8%	28.30
TLD [42]	88.5%	12.08	0.5%	88.07	100%	39.51	43.0%	31.30	11.6%	26.02	40.6%	7.49
ILT [5]	87.4%	11.40	100%	78.14	100%	60.95	100%	84.19	4.7%	20.88	0.4%	15.23
FT [4]	49.6%	32.05	50.5%	167.38	43.5%	58.13	87.0%	131.42	4.7%	29.30	0.1%	10.31
SRPT [6]	32.8%	44.80	40%	178.60	96.3%	69.38	88.9%	96.64	4.7%	21.21	0.9%	15.62
Sequence	*egtest01*		*egtest02*		*egtest04*		*Redteam*		*Jump*		*Animal*	
	SFR	Error	SFR	Error	SFR	Error	SFR	Error	SFR	Error	SFR	Error
SSOCBT	100%	6.40	100%	3.75	100%	3.33	100%	5.90	100%	5.02	100%	6.74
OCBT	100%	6.71	100%	6.20	100%	9.40	100%	4.46	100%	11.20	100%	6.84
OBT [10]	100%	7.01	29.3%	8.20	64.9%	7.11	79.5%	4.38	20.8%	27.83	88.6%	23.82
SSOBT [16]	13.5%	3.03	18.4%	9.03	77.9%	7.90	75.8%	5.45	22.5%	20.10	85.7%	10.20
MILT [18]	100%	8.30	100%	3.82	98.5%	9.33	100%	7.80	100%	9.94	90.0%	25.56
TLD [42]	96.0%	4.95	100%	8.88	67.2%	3.88	93.8%	4.59	88.1%	3.79	81.4%	164.76
ILT [5]	20.7%	5.73	37.6%	7.17	40.8%	2.37	24.1%	12.02	100%	6.74	100%	18.61
FT [4]	99.1%	5.73	96.7%	7.17	24.1%	2.37	6.0%	12.02	29.7%	14.75	32.9%	59.66
SRPF [6]	14.3%	12.77	34.2%	10.50	24.3%	5.44	6.6%	4.26	48.6%	72.97	45.7%	40.45

The dominant measure is SFR. First, the larger SFR is, the better the tracker is. If the SFR of the two trackers are the same, the tracker that generates lower tracking error is better.

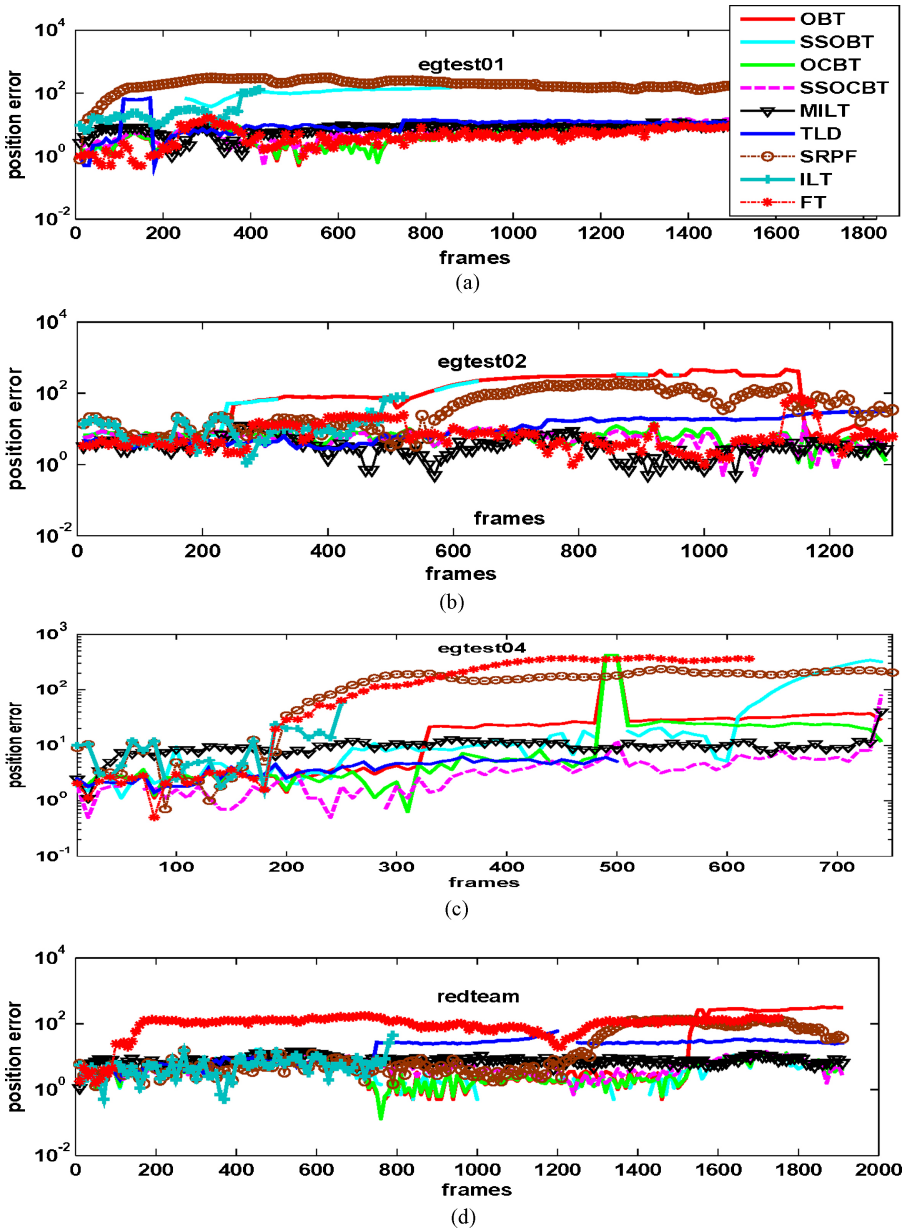

Fig. 11. Quantitative results on videos from CMU_VIVID dataset using different trackers.

The first test video is *David*, which was downloaded from http://www.cs.utoronto.ca/~dross/ivt/. In this video, a boy walks from a dark room to a lit room. His pose, the appearance of his face, and the background constantly change. The second video is *Doll*, which was recorded by a moving camera, and has a television in the background. The television set is on and dynamic, so the background is continuously changing, and sometimes there are sudden variations. Meanwhile, due to the camera's movement, the target comes out fuzzy from time to time. The third video is *Head*, which was recorded with a hand-held camera. The great challenge of this video is that it includes 180° out-of-plane rotation of the head and motion blur. The fourth test video is *Tiger*, which shows a tiger floating on the water. The glistening water reflecting the sun on the ripples and the shadows of the surroundings make it very challenging to track the tiger. Sometimes, the tiger is very blurred and appears very similar to the water. The fifth test video is *Store*, which is from the video data of TRECVID2010. In this video, a man runs through a room. The appearance of the background varies drastically. The last video is *Motor*, which was used in [20] and depicts a motor bumping along a rough mountain road. The motor disappears sometimes and its scale changes greatly.

The qualitative results on the above six test videos are displayed in Fig. 9. We can see that OCBT performs better than OBT, because our novel assumption on training samples helps the tracker select more discriminative features. Through exploiting the structure information implied in unlabeled samples, SSOCBT makes further improvement. During the process of tracking, SSOBT and TLD infer that whether the target is present or not; they do not provide tracking results if they think the target disappears. In some frames in Fig. 9, SSOBT,

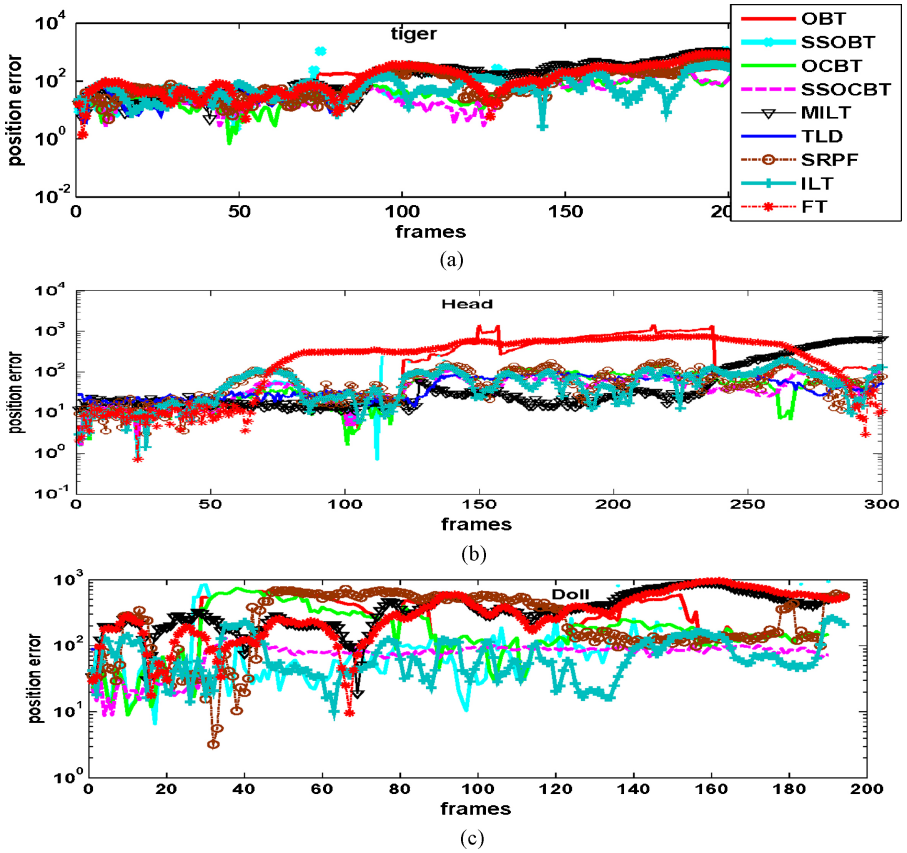

Fig. 12. Quantitative results on self-recorded videos using different trackers. (a) *Tiger*. (b) *Head*. (c) *Doll*.

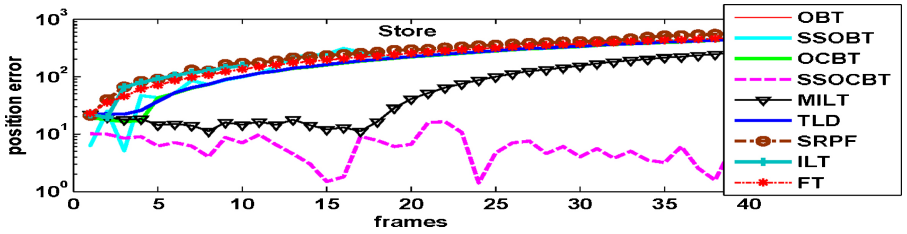

Fig. 13. Quantitative results on *Store*, which is from the TRECVID dataset, using different trackers.

and TLD could not provide tracking results because of their incorrect inferences. For example, in the *Doll* sequence, TLD judges that the target disappears in the 3^{rd} frame and never appears again. SSOBT performs redetection when it loses the target, so it could recover from failure (e.g., frame 60 in *Doll*). Applying CovBoost, when the doll is blurry, SSOCBT identifies the doll and succeeds in capturing it through the whole video. *Tiger* is a very good test video for illustrating the advantage of our tracker. The appearances of the tiger and water vary gradually. Some other trackers (e.g., OBT, SSOBT) lose the tiger quickly. One reason is that the data distribution is changing while it is assumed to be drawn from the same distribution for those trackers. As we show in Fig. 2, semi-supervised CovBoost could generate a correct decision plane, so SSOCBT can adapt to dynamic changes and performs well. When there is a distracting feature, OBT, SSOBT, and TLD fail. An example of this is in the tenth frame of *Store*. MILT insists on tracking the target for a few frames, but it loses the target when the target is closer to the second distracter. SSOCBT performs better, but it loses the target in frame 50, because there is partial occlusion and the background is very similar to the background. The results in *Motor* prove that SSOCBT can cope with significant variations.

To illustrate the effectiveness of SSOCBT, we test it on more image sequences from public datasets (e.g., CMU_VIVID [33]). The information about those videos is described in Table III. Some tracking results are displayed in Fig. 10. In every video, SSOCBT tracks the target from the beginning to the end, while other trackers fail when a partial occlusion, blur, or distracter appears.

Table IV displays the comparison results of the nine trackers described in Table IV in terms of SFR and the average position error of tracking results in those frames. The larger SFR is, the better the tracker is. If the SFR of the two trackers are the same, the tracker that generates lower tracking error is better. As we can see, SFR of SSOCBT is the largest in all the test videos. Although in some videos, the tracker achieves the same SFR with SSOCBT or OCBT, it has higher tracking

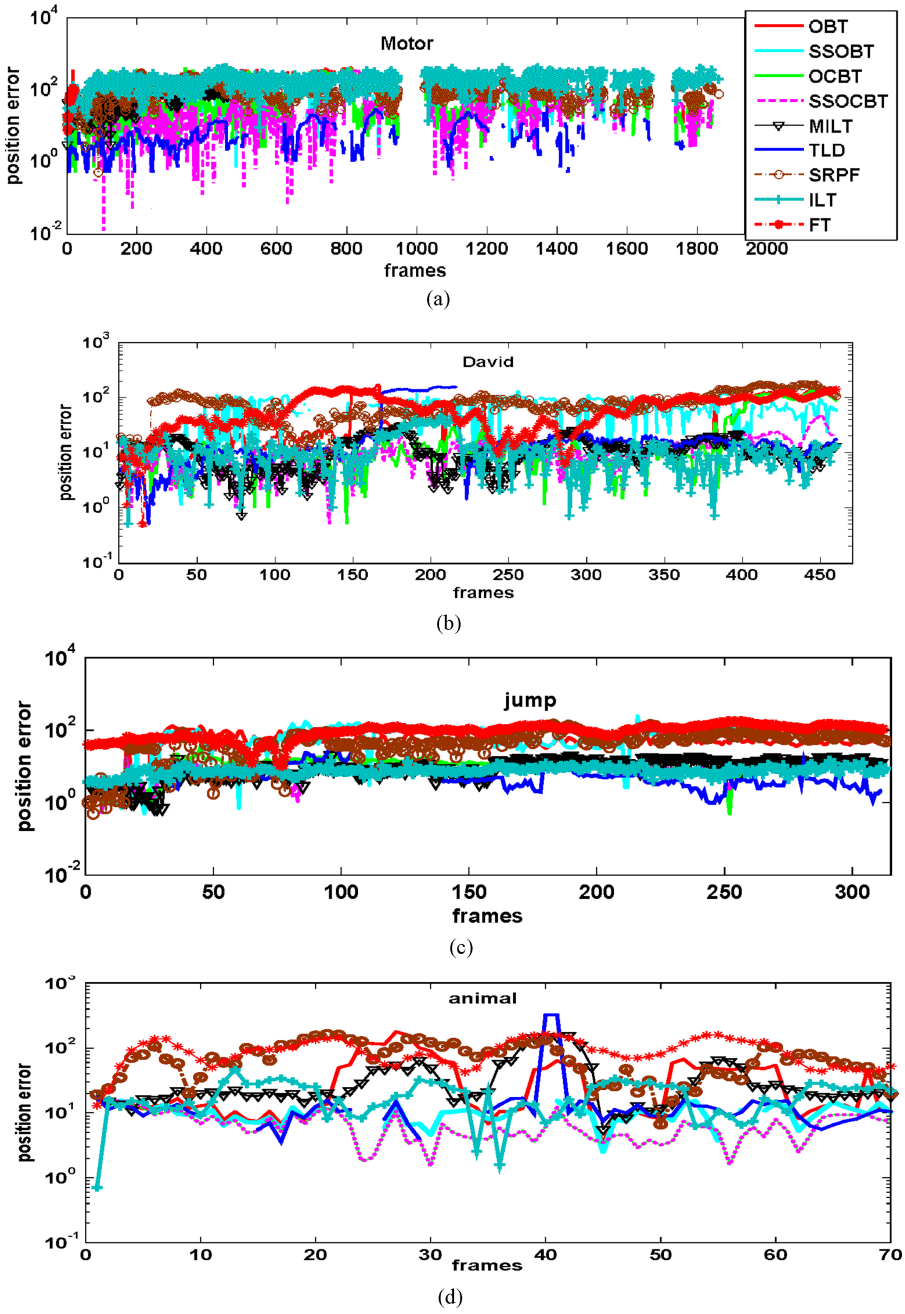

Fig. 14. Quantitative results on some videos used in other papers using different trackers. (a) *Motor*. (b) *David*. (c) *Jump*. (d) *Animal*.

error. This means SSOCBT has the best performance in most of the test videos.

To provide more concrete quantitative comparisons, we provide tracking error curves of all the trackers in every test video. Fig. 11 shows the tracking error on videos from CMU_VIVID dataset. Apparently, SSOCBT performs better than OBT, SSOBT, and MILT measuring with position error. As we discussed in Section II, among videos *egtest01, egtest02, egtest04*, and *Redteam*, the variations of object and background in *egtest01* are the smallest. Therefore, OBT performs better on *egtest01*. However, SSOBT and OBT cannot handle changes in the other three videos and their tracking results are very inaccurate compared with SSOCBT. Fig. 12 is about tracking error on self-recorded videos, in which the target and background vary gradually. We can see that the error curves of SSOCBT are smoother than that of the other trackers and its tracking error in every frame is very small. This means the performance of SSOCBT is very stable. Figs. 13 and 14 display error curves on videos frequently used in other papers. The tracking error of SSOCBT is much lower than that of other trackers especially in *Motor, Store*, and *Animal*.

D. Analysis

In our experiments, we discover that SSOBT's and OBT's performance are affected by the object's and background's KL distance in adjacent frames. For example, in *egtest01*, OBT and SSOBT begin to drift to the distracter in frame 246, while the KL distance of background's distribution in adjacent frames

increases around that frame, as shown in Fig. 3(d). In *Jump*, the performance of OBT becomes unstable in frame 27 and it often loses the target from then on. Fig. 3(i) and (j) shows the KL distance of the target and background in video *Jump*; we can see that the KL distance of the target varies greatly from frame 27 to frame 50. In a word, object and background changes have a significant effect SSOBT's and OBT's performance because they assume that data follows the same distribution in every frame. Because of our "covariate shift" assumption on samples' distribution in different frames, SSOCBT and OCBT treat samples differently and are able to handle their variations. Therefore, SSOCBT and OCBT are not sensitive to object and background changes during tracking, and thus achieve good performance.

VII. Conclusion

In this paper, we treated tracking as an online binary classification problem. After analyzing how much the object's and background's distributions change in every frame, we proposed that when providing the tracking result in the current frame, the samples collected in the previous frames would be considered auxiliary data, while the samples obtained in the current frame would be regarded as target data. We assumed that the target and the auxiliary data were under "covariate shift" instead of following the same distribution. Furthermore, we thought that the unlabeled samples in the next frame, which we wanted to classify, could provide useful information. We formed a strong classifier based on the above three kinds of samples using the semisupervised CovBoost algorithm. It selects effective features that distinguish the target from the background. We developed an online updating algorithm for semisupervised CovBoost, which made our approach easy to use for tracking and greatly improved the tracking performance, especially when the target's appearance changed heavily or when partial occlusion and motion blur occurred. Experimental results demonstrated that the proposed tracker outperformed state-of-the-art trackers.

References

[1] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 5, pp. 564–577, May 2003.

[2] M. Isard and A. Blake, "Condensation-conditional density propagation for visual tracking," *Int. J. Comput. Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[3] M. Black and A. Jepson, "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation," *Int. J. Comput. Vision*, vol. 26, no. 1, pp. 63–84, 1998.

[4] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, vol. 1. Jun. 2006, pp. 798–805.

[5] D. Ross, J. Lim, R. Lin, and M. Yang, "Incremental learning for robust visual tracking," *Int. J. Comput. Vision*, vol. 77, no. 1, pp. 125–141, 2008.

[6] X. Mei and H. Ling, "Robust visual tracking using l1 minimization," in *Proc. IEEE Int. Conf. Comput. Vision*, Oct. 2009, pp. 1436–1443.

[7] G. Li, W. Qu, and Q. Huang, "A multiple tragets appearance tracker based on object interaction models," *IEEE Trans. Circuit Syst. Video Technol.*, vol. 22, no. 3, pp. 450–464, Mar. 2012.

[8] R. Collins, Y. Liu, and M. Leordeanu, "On-line selection of discriminative tracking features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 1, pp. 1631–1643, Oct. 2005.

[9] S. Avidan, "Ensemble tracking," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, Feb. 2005, pp. 494–501.

[10] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting," in *Proc. Brit. Mach. Vision Conf.*, 2006, pp. 47–56.

[11] J. Wang, X. Chen, and W. Gao, "Online selectiong discriminative tracking features using particle filter," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, vol. 2. Jun. 2005, pp. 1037–1042.

[12] G. Li, Q. Huang, J. Pang, S. Jiang, and L. Qin, "Online selection of the best *k*-feature subset for object tracking," *J. Vis. Comun. Image Represent.*, vol. 23, no. 2, pp. 254–263, 2012.

[13] K. Pearson, "Onlines and planes of closest fit to systems of points in space," *Philosoph. Mag.*, vol. 2, no. 6, pp. 559–572, 1901.

[14] F. Tang, S. Brennan, Q. Zhao, and H. Tao, "Co-tracking using semi-supervised support vector machines," in *Proc. IEEE Int. Conf. Comput. Vision*, Oct. 2007, pp. 1–8.

[15] Q. Yu, T. Dinh, and G. Medioni, "Online tracking and reacquisition using co-trained generative and discriminative trackers," in *Proc. Eur. Conf. Comput. Vision*, 2008, pp. 678–691.

[16] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *Proc. Eur. Conf. Comput. Vision*, 2008, pp. 234–247.

[17] R. Liu, J. Cheng, and H. Lu, "A robust boosting tracker with minimum error bound in a co-training framework," in *Proc. IEEE Int. Conf. Comput. Vision*, Sep.–Oct. 2009, pp. 1459–1466.

[18] B. Babenko, M. H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Proc. Comput. Vision Pattern Recognit.*, 2009, pp. 798–805.

[19] B. Zeisl, C. Leistner, A. Saffari, and H. Bischof, "On-line semi-supervised mulitple-instance boosting," in *Proc. Comput. Vision Pattern Recognit.*, 2010, p. 1879.

[20] Z. Kalal, J. Matas, and K. Mikolajczyk, "P–N learning: Bootstrapping binary classifiers by structural constraints," in *Proc. Comput. Vision Pattern Recognit.*, 2010, pp. 49–56.

[21] A. Goldberg, M. Li, and X. Zhu, "Online manifold regularization: A new learning setting and empirical study," in *Proc. Eur. Conf. Comput. Vision*, 2008, pp. 393–407.

[22] L. Matthews, T. Ishikawa, and S. Baker, "The template update problem," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 6, pp. 810–815, Jun. 2004.

[23] H. Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function," *J. Statist. Planning Inference*, vol. 90, no. 2, pp. 227–244, 2000.

[24] M. Sugiyama, M. Krauledat, and K. R. Muller, "Covariate shift adaptation by importance weighted cross validation," *J. Mach. Learning Res.*, vol. 8, pp. 985–1005, Dec. 2007.

[25] S. Bickel, M. Brijckner, and T. Scheffer, "Discriminative learning under covariate shift," *J. Mach. Learning Res.*, vol. 10, pp. 2137–2155, Dec. 2009.

[26] J. Pang, Q. Huang, S. Yan, S. Jiang, and L. Qin, "Transferring boosted detectors toward viewpoint and scene adaptiveness," *IEEE Trans. Image Process.*, vol. 20, no. 5, pp. 1388–1400, May 2011.

[27] Y. Yao and G. Doretto, "Boosting for transfer learning with multiple sources," in *Proc. Comput. Vision Pattern Recognit.*, 2010, pp. 1855–1862.

[28] G. Li, L. Qin, Q. Huang, J. Pang, and S. Jiang, "Treat samples differently: Object tracking with semi-supervised online CovBoost," in *Proc. IEEE Int. Conf. Comput. Vision*, Nov. 2011, pp. 627–634.

[29] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 1951.

[30] B. Bigi, "Using Kullback–Leibler distance for text categorization," in *Proc ECIR*, 2003, pp. 205–319.

[31] B. Fuglede and F. Topse, "Jensen–Shannon divergence and Hilbert space embedding," in *Proc. IEEE Int. Symp. Inform. Theory*, Jun. 2004, p. 31.

[32] C. H. Bennett, P. Gacs, M. Li, P. Vitanyi, and W. Zurek, "Information distance," *IEEE Trans. Inform. Theory*, vol. 44, no. 4, pp. 1207–1423, Jul. 1998.

[33] *Cmu Vivid Dataset* (2008) [Online]. Available: http:/www.vividevaluation.ri.cmu.edu/

[34] Y. Freund and R. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," in *Proc. EuroCOLT*, 1994, pp. 23–37.

[35] X. Zhu, "Semi-supervised learning literature survey," Univ. Wisconsin, Madison, Comput. Sci. Tech. Rep. 1530, 2005.

[36] T. Brox, A. Bruhn, N. Papenberg, and J. Wiecker, "High accuracy optical flow estimation based on a theory of warping," in *Proc. Eur. Conf. Comput. Vision*, 2004, pp. 25–36.

[37] F. Zhao, Q. Huang, and W. Gao, "Image matching by normalized cross-correlation," in *Proc. IEEE Int. Conf. Acoustics, Speech Signal Process.*, May 2006, p. II.
[38] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proc. Comput. Vision Pattern Recognit.*, 2000, pp. 142–149.
[39] T. Hertz, A. Bar-Hillel, and D. Weinshall, "Learning distance functions for image retrieval," in *Proc. Comput. Vision Pattern Recognit.*, 2004, pp. II-570–II-577.
[40] C. Leistner, H. Grabner, and H. Bischof, "Semi-supervised boosting using visual similarity learning," in *Proc. Comput. Vision Pattern Recog.*, 2008, pp. 1–8.
[41] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. Comput. Vision Pattern Recognit.*, 2001, pp. I-511–I-518.
[42] Z. Kalal, K. Mikolajczyk, and J. Matas, "Face-TLD: Tracking-learning-detection applied to faces," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2010, pp. 3789–3792.
[43] J. Kwon and K. M. Lee, "Visual tracking decomposition," in *Proc. Comput. Vision Pattern Recognit.*, 2010, pp. 1269–1276.

Guorong Li received the B.S. degree in technology of computer application from the Renmin University of China, Beijing, China, in 2006, and the Ph.D. degree in technology of computer application from the Graduate University of the Chinese Academy of Sciences (GUCAS), Beijing, in 2012.

She is currently a Post-Doctoral Fellow with GUCAS. Her current research interests include object tracking, video analysis, pattern recognition, and computer vision.

Qingming Huang (SM'08) received the B.S. degree in computer science and the Ph.D. degree in computer engineering from the Harbin Institute of Technology, Harbin, China, in 1988 and 1994, respectively.

He is currently a Professor with the University of the Chinese Academy of Sciences, Beijing, China, and an Adjunct Research Professor with the Institute of Computing Technology, Chinese Academy of Sciences. He has authored or coauthored more than 200 academic papers in prestigious international journals and conferences. His current research interests include multimedia computing, video adaptation, image processing, computer vision, and pattern recognition.

Dr. Huang was the recipient of the Distinguished Young Scientist Award in 2010 from the National Natural Science Foundation of China. He is a Reviewer for the IEEE TRANSACTIONS ON MULTIMEDIA, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, and the IEEE TRANSACTIONS ON IMAGE PROCESSING. He has served as an Organization Committee Member and a TPC Member for various conferences, including ACM Multimedia, CVPR, ICCV, ICME, and PCM.

Lei Qin (M'06) received the B.S. and M.S. degrees in mathematics from the Dalian University of Technology, Dalian, China, in 1999 and 2002, respectively, and the Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences (CAS), Beijing, China, in 2008.

He is currently a Faculty Member with the Key Laboratory of Intelligent Information Processing and the Institute of Computing Technology, CAS. He has authored or coauthored more than ten technical papers in the area of computer vision. His current research interests include image or video processing, computer vision, and pattern recognition.

Shuqiang Jiang (SM'08) received the M.S. degree from the College of Information Science and Engineering, Shandong University of Science and Technology, Shandong, China, in 2000, and the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences (CAS), Beijing, China, in 2005.

He is currently a Faculty Member with the Digital Media Research Center, Institute of Computing Technology, CAS. He is also with the Key Laboratory of Intelligent Information Processing, CAS. His current research interests include multimedia processing and semantic understanding, pattern recognition, and computer vision.