Object Tracking via Partial Least Squares Analysis

Qing Wang, *Student Member, IEEE*, Feng Chen, *Member, IEEE*, Wenli Xu, and Ming-Hsuan Yang, *Senior Member, IEEE*

***Abstract*— We propose an object tracking algorithm that learns a set of appearance models for adaptive discriminative object representation. In this paper, object tracking is posed as a binary classification problem in which the correlation of object appearance and class labels from foreground and background is modeled by partial least squares (PLS) analysis, for generating a low-dimensional discriminative feature subspace. As object appearance is temporally correlated and likely to repeat over time, we learn and adapt multiple appearance models with PLS analysis for robust tracking. The proposed algorithm exploits both the ground truth appearance information of the target labeled in the first frame and the image observations obtained online, thereby alleviating the tracking drift problem caused by model update. Experiments on numerous challenging sequences and comparisons to state-of-the-art methods demonstrate favorable performance of the proposed tracking algorithm.**

***Index Terms*— Appearance model, object tracking, partial least squares analysis.**

I. Introduction

OBJECT tracking is an important problem in image analysis with numerous applications. It is concerned with low-level visual processing and high-level image analysis, and is widely used in image understanding, human-computer interaction, surveillance, and robotics, to name a few. This problem is challenging as it needs to deal with appearance variations caused by numerous factors such as illumination, pose angle, occlusion, background clutter, and camera motion. To tackle these challenges, this paper presents a tracking method that learns a robust object representation by partial least squares analysis and adapts to appearance change of the target and background while reducing drift.

Manuscript received October 3, 2011; revised March 17, 2012; accepted May 11, 2012. Date of publication June 22, 2012; date of current version September 13, 2012. The work of Q. Wang and F. Chen was supported in part by the Natural Science Foundation of China under Grant 61071131 and the Beijing Natural Science Foundation (NSF) under Grant 4122040. The work of W. Xu was supported in part by the National Key Basic Research and Development Program of China under Grant 2009CB320602. The work of M.-H. Yang was supported in part by the NSF CAREER under Grant 1149783 and NSF IIS under Grant 1152576. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Theo Gevers.

Q. Wang, F. Chen, and W. Xu are with the Department of Automation, National Laboratory for Information Science and Technology, Tsinghua University, Beijing 10084, China (e-mail: qing-wang07@mails.tsinghua.edu.cn; chenfeng@mail.tsinghua.edu.cn; xuwl@mail.tsinghua.edu.cn).

M.-H. Yang is with the Department of Electrical Engineering and Computer Science, University of California, Merced, CA 95344 USA (e-mail: mhyang@ucmerced.edu).

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/TIP.2012.2205700

Recent years have seen significant progress in effective representation schemes for robust object tracking. One successful approach is to exploit features in a high-dimensional space for object recognition. However, the computational complexity is likely to increase significantly as a result of high dimensionality of features. Since object tracking can be posed as a binary classification problem with the goal to separate the target object from the background, a discriminative object representation scheme greatly facilitates this task. Therefore, feature selection is of crucial importance for generating an effective low-dimensional discriminative subspace. In this paper, we achieve this by learning a feature subspace with a few positive and negative samples in the high-dimensional feature space via partial least squares (PLS) analysis. The learned feature subspace is then utilized to construct an appearance model. As appearance of an object in consecutive frames is temporally correlated and likely to repeat over time, we learn and adapt multiple appearance models with PLS analysis for robust tracking.

When new tracking results are obtained during tracking, the proposed appearance models are adapted to account for the target and background appearance change. Since the only ground truth during tracking is obtained in the first frame by manually labeling or automatically detection, we retain the appearance model initialized at the outset as a static representation for inference. Based on the adaptive and static appearance models, we define two likelihood functions to measure the probabilities that a target object (with respect to some state parameter) being generated from them. Object tracking is carried out within the Bayesian inference framework, and a two-stage particle filtering method is developed to estimate tracking results sequentially. Due to the temporal correlation of tracking results in consecutive frames, we use the adaptive likelihood function to estimate the approximate target state in the first stage, and then use the static likelihood function to determine the object location in the second stage.

The main contributions of this paper are as follows. First, we use PLS analysis to learn low-dimensional discriminative feature subspace for object representation. Since object tracking is posed as a task to discriminate the target object from the background, object representation based with PLS analysis is more effective than the widely used generative models such as principal component analysis (PCA) or its variants [17], [24]. As no exhaustive search is carried out to select or combine features, our representation scheme is also more efficient than existing discriminative methods [7], [10]. Second, we represent an object with

multiple appearance models for robust tracking. To account for large and complex appearance change of an object, we use more than one appearance model which is more effective than existing methods with one single linear representation [17], [24]. Third, we propose a two-stage particle filtering method. This tracking method makes use of the appearance model initialized in the first frame and image observations obtained online, thereby alleviating the drift problem during model update. Evaluated against several state-of-the-art tracking methods, the proposed tracking algorithm achieves favorable performance with higher success rates and lower tracking errors.

II. Related Work

Existing tracking algorithms in the literature can be roughly categorized as either generative or discriminative. Most recent tracking methods focus on robust representation schemes with either generative appearance models or discriminative feature selection to account for the inevitable appearance change of objects as well as background.

Generative methods track objects by searching for the image region that best matches a template or an appearance model. Reference templates based on pixel intensity [12] and color histograms [8] have been used for visual tracking. Such methods perform well when there is no drastic change of object appearance and when the background is not cluttered. On the other hand, more effective appearance models can be learned with a set of training data. Black *et al.* [6] learn a subspace appearance model offline and integrate it with the optical flow framework for object tracking. This method has been shown to perform well within the predefined views. In [5], Black *et al.* present a mixture model to better account for various types of appearance change caused by shape deformation and illumination variation. Different from static appearance models, numerous adaptive appearance models have also been proposed for object tracking. Jepson *et al.* [14] learn a Gaussian mixture model via an online expectation maximization algorithm to handle target appearance variations during tracking. Aside from mixture models, online subspace learning methods have been developed to model appearance variation of objects for tracking [13], [17], [24]. Kwon *et al.* [16] extend the classic particle filter framework with multiple dynamic and observation models to account for appearance and motion variation. Recent findings in sparse representation have also been applied to object tracking [20] due to their robustness to occlusion or image noise. Compared with static appearance models, adaptive schemes are usually more flexible and effective for object tracking.

Discriminative methods pose object tracking as a binary classification problem within a local image region in which the goal is to separate the target object from the background. While generative methods model only the target appearance, discriminative algorithms exploit visual information from both the target and the background. Avidan [1] extends a support vector machine (SVM) classifier within the optical flow framework for object tracking. In addition, the relevance vector machine (RVM) has also been used for designing tracking algorithms [27]. These methods use a predefined and fixed feature set for classifier learning. In the literature, there are numerous tracking methods that select good features for appearance models. Collins *et al.* [7] use the variance ratio of two classes to select discriminative color features for object tracking. In contrast, Lin *et al.* [18] propose a method that extends Fisher linear discriminant (FLD) analysis to learn an adaptive discriminative generative model for object tracking. Several algorithms based on online boosting [2], [3], [10], [15] have also been developed to distinguish the foreground from the background by an ensemble of classifiers.

In order to reduce tracking drift caused by updating the generative appearance models or discriminative classifiers with newly obtained results, several algorithms have been developed. Matthews *et al.* [19] propose an update method for the Lucas-Kanade algorithm by using a dynamic template extracted in the most recent frames to estimate the initial tracking result first, and then using the static template obtained from the first frame to determine the target location. In addition to supervised approaches for discriminative object tracking [2], [7], [10], Grabner *et al.* [11] treat all the observations obtained online as unlabeled data within the semi-supervised learning framework for classifier update. Babenko *et al.* [3] present an online multiple instance learning method that handles ambiguously labeled positive and negative data to reduce tracking drift. Recently, Kalal *et al.* [15] also posit the target observations as unlabeled data and exploit their underlying structure to select both positive and negative samples for classifier update.

III. Object Representation

Partial least squares analysis [22], [28] is a statistical method for modeling relations between sets of variables via some latent quantities. In PLS analysis, the observed data is assumed to be generated by a process driven by a small number of latent variables. In this paper, we formulate object tracking as a classification problem with PLS analysis to learn a low-dimensional and discriminative feature subspace.

A. Partial Least Squares Analysis

Let $\mathcal{X} \in \mathbb{R}^m$ be an m-dimensional space of variables and $\mathcal{Y} \in \mathbb{R}^n$ be an n-dimensional space of other variables. With N observed samples from each space $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ that form two blocks of variables, $X \in \mathbb{R}^{N\times m}$ and $Y \in \mathbb{R}^{N\times n}$, PLS methods find new spaces where most variations of the observed samples can be preserved, and the learned latent variables from two blocks are more correlated than those in the original spaces

$$\begin{aligned} X &= TP^\top + E \\ Y &= UQ^\top + F \end{aligned} \tag{1}$$

where $T \in \mathbb{R}^{N\times p}$ and $U \in \mathbb{R}^{N\times p}$ are factor (score, component) matrices, $P \in \mathbb{R}^{m\times p}$ and $Q \in \mathbb{R}^{n\times p}$ are loading matrices, and $E \in \mathbb{R}^{N\times m}$ and $F \in \mathbb{R}^{N\times n}$ are error terms. With PLS analysis, each variable is represented by a p-dimensional vector.

To decompose X and Y by Equation (1), PLS algorithms first compute the weight vectors $\mathbf{w}_1$ and $\mathbf{c}_1$ such that most variations in X and Y can be retained by $\mathbf{t}_1 = X\mathbf{w}_1$ and $\mathbf{u}_1 = Y\mathbf{c}_1$

$$\begin{aligned} &\max_{\mathbf{w}_1} Var(\mathbf{t}_1) \\ &\max_{\mathbf{c}_1} Var(\mathbf{u}_1) \end{aligned} \tag{2}$$

where $\mathbf{t}_1$ and $\mathbf{u}_1$ are the first columns of T and U, respectively, and $Var(\cdot)$ denotes the variance.

Meanwhile, PLS analysis also requires $\mathbf{t}_1$ to best explain $\mathbf{u}_1$

$$\max_{\mathbf{w}_1,\mathbf{c}_1} \rho(\mathbf{t}_1, \mathbf{u}_1) \tag{3}$$

where $\rho(\mathbf{t}_1, \mathbf{u}_1) = Cov(\mathbf{t}_1, \mathbf{u}_1)/\sqrt{Var(\mathbf{t}_1)Var(\mathbf{u}_1)}$ defines the correlation coefficient between $\mathbf{t}_1$ and $\mathbf{u}_1$, and $Cov(\mathbf{t}_1, \mathbf{u}_1) = \mathbf{t}_1^\top \mathbf{u}_1/N$ denotes the sample covariance between $\mathbf{t}_1$ and $\mathbf{u}_1$. Combining Equation (2) and Equation (3), PLS analysis maximizes the covariance between $\mathbf{t}_1$ and $\mathbf{u}_1$ in the first step

$$\max_{\mathbf{w}_1,\mathbf{c}_1} Cov(\mathbf{t}_1, \mathbf{u}_1) = \max_{\mathbf{w}_1,\mathbf{c}_1} \sqrt{Var(\mathbf{t}_1)Var(\mathbf{u}_1)}\rho(\mathbf{t}_1, \mathbf{u}_1). \tag{4}$$

Therefore, $\mathbf{w}_1$ and $\mathbf{c}_1$ can be computed by solving the following optimization problem:

$$\begin{aligned} &\max \langle X\mathbf{w}_1, Y\mathbf{c}_1 \rangle \\ &s.t. \;\; {\mathbf{w}_1}^\top \mathbf{w}_1 = 1, {\mathbf{c}_1}^\top \mathbf{c}_1 = 1 \end{aligned} \tag{5}$$

where $\langle X\mathbf{w}_1, Y\mathbf{c}_1 \rangle$ denotes the inter product of $X\mathbf{w}_1$ and $Y\mathbf{c}_1$. The optimal weight vector $\mathbf{w}_1$ for the above optimization problem is the first eigenvector of the following eigenvalue problem [22], [28]

$$X^\top Y Y^\top X \mathbf{w}_1 = \lambda \mathbf{w}_1. \tag{6}$$

Similarly, $\mathbf{c}_1$ can be obtained by solving another eigenvalue problem

$$Y^\top X X^\top Y \mathbf{c}_1 = \lambda \mathbf{c}_1. \tag{7}$$

After the first step, the PLS method iteratively computes other weight vectors. When $\mathbf{w}_1$ and $\mathbf{c}_1$ are available, the score vectors can be computed by $\mathbf{t}_1 = X\mathbf{w}_1$, $\mathbf{u}_1 = Y\mathbf{c}_1$, and loadings (first columns of P and Q) can be computed by $\mathbf{p}_1 = \frac{X^\top \mathbf{t}_1}{\mathbf{t}_1^\top \mathbf{t}_1}$ and $\mathbf{q}_1 = \frac{Y^\top \mathbf{u}_1}{\mathbf{u}_1^\top \mathbf{u}_1}$, respectively. The data matrices X and Y are then deflated by subtracting their rank-one approximations

$$\begin{aligned} X &\leftarrow X - \mathbf{t}_1 \mathbf{p}_1^\top \\ Y &\leftarrow Y - \mathbf{u}_1 \mathbf{q}_1^\top. \end{aligned} \tag{8}$$

The new X and Y are used to compute $\mathbf{w}_2$, $\mathbf{c}_2$ based on Equation (6) and Equation (7). This process is repeated until the residuals are small enough or a predefined number of weight vectors $\{\mathbf{w}_1, \ldots, \mathbf{w}_p\}$ and $\{\mathbf{c}_1, \ldots, \mathbf{c}_p\}$ are obtained.

Since its early applications in the field of chemometrics, PLS has become a useful tool in neuroscience, bioinformatics, pattern recognition, and data mining [21], [23], [25], [26]. More details about PLS analysis can be found in [22], [28].

B. Learning Appearance Models With PLS Analysis

In this paper, we pose object tracking as a classification problem which labels the target (positive) and background (negative) feature variables with different values. The above discussion of PLS analysis indicates that a low-dimensional space can be learned where the latent quantities from different sets of observed variables are more correlated than those in the original spaces. Therefore, we use PLS analysis to model the correlation of object appearance and class label due to its capacity for both dimensionality reduction and classification.

Within PLS formulation, the variables in our tracking task consist of two classes including feature vectors and class label. In the following sections, we use $\mathcal{X} \in \mathbb{R}^m$ to denote the feature space for object description, and $\mathcal{Y} \in \mathbb{R}$ to denote the class label space of an object. After the target object is manually or automatically located in the first frame, we have a positive sample $\mathbf{x}_1$ by extracting a feature vector from the warped image specified by the state parameter. If more positive samples are needed for training, we generate virtual data by small perturbations and extract corresponding feature vectors. In order to collect negative samples, we randomly draw samples from an annular region defined by $\gamma < \| \mathbf{l}_{neg} - \mathbf{l} \| < \beta$ (γ and β are inner and outer radiuses, respectively), in which $\mathbf{l}$ is the target location, and $\mathbf{l}_{neg}$ is the location of a negative sample. Fig. 1 illustrates how positive and negative samples are drawn in a frame. With the obtained training data set, we use PLS analysis to determine an appearance model of the target object.

Let $X = [\mathbf{x}_1^p, \ldots, \mathbf{x}_{N_p}^p, \mathbf{x}_1^n, \ldots, \mathbf{x}_{N_n}^n]^\top \in \mathbb{R}^{N \times m}$ denote the feature vectors we have collected, and $\mathbf{y} = [1, \ldots, 1, 0, \ldots, 0]^\top \in \mathbb{R}^{N \times 1}$ denote the corresponding class labels, where N_p and N_n are the numbers of positive and negative samples, respectively ($N = N_p + N_n$). We center X and $\mathbf{y}$ by subtracting their corresponding means $\bar{\mathbf{x}}$ and $\bar{y}$ to form $\bar{X}$ as well as $\bar{\mathbf{y}}$. With PLS analysis, $\bar{X}$ and $\bar{\mathbf{y}}$ can be decomposed by

$$\begin{aligned} \bar{X} &= TP^\top + E \\ \bar{\mathbf{y}} &= U\mathbf{q}^\top + \mathbf{f} \end{aligned} \tag{9}$$

where $T \in \mathbb{R}^{N \times p}$ and $U \in \mathbb{R}^{N \times p}$ contain N observations of the p extracted latent variables, $P \in \mathbb{R}^{m \times p}$ and $\mathbf{q} \in \mathbb{R}^{1 \times p}$ represent loadings, and $E \in \mathbb{R}^{N \times m}$ as well as $\mathbf{f} \in \mathbb{R}^{N \times 1}$ are residuals. Since $\bar{\mathbf{y}}$ has one variable, $\mathbf{u}_1 = \bar{\mathbf{y}}\mathbf{c}_1$, and $\mathbf{c}_1^\top \mathbf{c}_1 = 1$, therefore $\mathbf{c}_1$ must be a scalar. It follows that, $\mathbf{c}_1 = 1$, $\mathbf{u}_1 = \bar{\mathbf{y}}$, and we only need to learn the latent variables for the feature vectors. As discussed in Section III-A, we compute $\mathbf{w}_1 = \bar{X}^\top \bar{\mathbf{y}}/\|\bar{X}^\top \bar{\mathbf{y}}\|$ in the first step. In the k-th ($k > 1$) iteration, we first compute $\bar{X}_k = \bar{X}_{k-1} - \mathbf{t}_{k-1}\mathbf{p}_{k-1}^\top$ (where $\bar{X}_1 = \bar{X}$), and then obtain $\mathbf{w}_k = \bar{X}_k^\top \bar{\mathbf{y}}/\|\bar{X}_k^\top \bar{\mathbf{y}}\|$.

Once the weight matrix $W = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_p]$ is computed, the initial appearance model can be denoted by $\mathcal{A}_1 = \{\bar{\mathbf{x}}^p, \bar{\mathbf{x}}, W\}$, where $\bar{\mathbf{x}}^p$ is the mean of the positive samples. A test sample, $\mathbf{x} \in \mathbb{R}^m$, can be projected onto the learned latent feature space specified by $\mathcal{A}_1$ to get a latent feature vector $\mathbf{z} = W^\top \mathbf{x}^c \in \mathbb{R}^p$, where $\mathbf{x}^c = \mathbf{x} - \bar{\mathbf{x}}$. Using the latent feature space $\mathcal{Z} \in \mathbb{R}^p$ with lower dimensionality, a target object can be more easily discriminated from the background than in the original feature space $\mathcal{X} \in \mathbb{R}^m$.

(a) (b)

Fig. 1. Samples are drawn in a frame. (a) Target object is marked by a red polygon from which a positive sample can be collected. The annular region specified by blue lines is used to generate candidate locations of negative samples. (b) Negative samples are collected from the blue polygons.

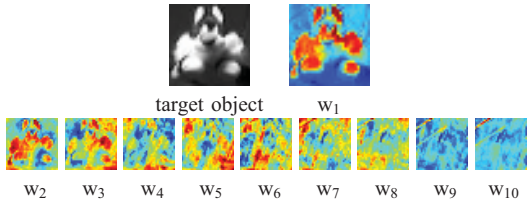

Fig. 2. Saliency (importance) maps generated by PLS analysis. The warped target image and the saliency map of the target region specified by $\mathbf{w}_1$ are shown in the first row. The second row shows the other saliency maps. The chosen color map is "jet," which ranges from blue to red, and passes through the colors cyan, yellow, and orange, from the lowest to the highest importance.

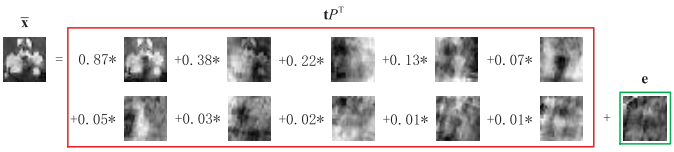

Fig. 3. Decomposition of the object image using (9). The loadings and residual are illustrated in the image form.

The weight vector $\mathbf{w}_i \in \mathbb{R}^m$ $(i = 1, \ldots, p)$ of W reflects the importance of each original feature variable for object description and classification. If each feature variable in the selected feature space $\mathcal{X}$ is a function of pixel location in an object region, then the importance of this feature variable is related to the discriminability between the target and the background classes at a given location. Therefore, we can use $\mathbf{w}_i$ to generate a saliency (importance) map, which shows the discriminative strength of different locations in an object region. For example, if each variable describes the intensity of one pixel and a feature vector represents the ensemble of pixel intensities in an object region, a subspace can be learned by PLS analysis with some positive and negative samples, and the saliency maps specified by $\mathbf{w}_i$ (e.g., $i = 1, \ldots, 10$) are shown in Fig. 2 where red pixels indicate higher importance of a feature variable (i.e., with more discriminative strength). It is worth noticing the red pixels of the saliency map with $\mathbf{w}_1$ concentrate on the target object and blue pixels (with less importance) appear in the background region. With the learned subspace, a feature vector can be decomposed as illustrated in Fig. 3, where the coefficients are values of the learned latent variables. The discriminative strength of the latent variables is shown in decreasing order.

C. Relation to Other Subspace Models

In the tracking literature, numerous methods have been proposed to learn feature subspaces using generative and discriminative approaches such as Fisher linear discriminant (FLD) analysis [18] and principal component analysis (PCA) [17], [24]. In this section, we draw some connections between PLS analysis and FLD as well PCA methods within the context of feature extraction for object tracking.

1) Connection to Fisher Linear Discriminant Analysis: The goal of FLD analysis is to maximize a projection function that renders the largest separation between the class means of projected samples and the smallest variance within each class, thereby minimizing overlapping points with different labels in the feature subspace. Similar to PLS methods, FLD algorithms can learn a low-dimensional discriminative feature subspace for classification. However, for a data set with C classes, FLD can find at most $C-1$ weight vectors which span a $(C-1)$-dimensional subspace. For problems with binary classes considered in this paper, FLD learns a 1-D subspace unless the positive and negative samples are further clustered into multiple classes to increase the subspace dimensionality. More importantly, when the number of training samples is smaller than the dimensionality of the feature space (known as small sample problem which often happens in object tracking and recognition), the covariance matrix used for learning the FLD subspace is rank deficient and some techniques need to be employed for solving the optimization problem. In contrast, there are no such problems in PLS analysis and the dimensionality of the feature subspace can be determined flexibly.

2) Connection to Principal Component Analysis: PCA is an unsupervised learning method which is often used to model data with a generative subspace. This subspace model is designed for dimensionality reduction as no label information or discriminative constraints are exploited. In [17], [24], object tracking is by incrementally learning a linear subspace model with both mean (of the target observations) update and eigenbasis update. Although these online PCA-based tracking methods have shown good performance in several scenarios, their formulations do not exploit the label information and they can hardly work well in cluttered background. For object tracking, the mean of the observed image needs to be updated due to appearance change [24], which corresponds to the largest eigenvector for uncentered data. We show in Section V that a template-based tracking method with adaptive mean update can generate similar tracking results as the approach proposed in [24]. Instead, PLS analysis computes weight vectors by maximizing covariance between training samples and their corresponding class labels. Thus, PLS based object representation is likely to be more discriminative and effective for object tracking.

D. Object Representation With Multiple Appearance Models

Since the appearance change of an object during a long period of time may be quite nonlinear and complex, one linear appearance model is not likely to suffice. However, appearance of a target object may be temporally correlated and may repeat over time. We therefore learn multiple appearance models

for more effective object representation. Fig. 4 shows that observations of a target over a long period can be divided into multiple sets. Within the i-th set, the object appearance does not change much and we use PLS analysis to learn a discriminative appearance model $\mathcal{A}_i = \{\bar{\mathbf{x}}_i^p, \bar{\mathbf{x}}_i, W_i\}$. Therefore, the appearance of a target object can be represented by multiple appearance models $\mathscr{A} = \{\mathcal{A}_1, \ldots, \mathcal{A}_k\}$, where k ($k \leq K$) is the number of appearance models and K is a predefined value. The proposed representation scheme is more effective than existing methods based on one single linear appearance model [17], [18], [24]. Note that only one appearance model $\mathcal{A}_1$ is initialized in the first frame, and we present an adaptive method for learning multiple appearance models online in the next section.

With our representation scheme, we define a distance metric for the tracking task. In this paper, the distance between a test sample $\mathbf{x} \in \mathbb{R}^m$ and the learned appearance model set $\mathscr{A}$ is defined as

$$d = \min\{d_i | i = 1, \ldots, k\} \tag{10}$$

where d_i is the distance between $\mathbf{x}$ and the i-th appearance model $\mathcal{A}_i$, and is defined by

$$\begin{aligned} d_i &= \left\| W_i^\top (\mathbf{x} - \bar{\mathbf{x}}_i) - W_i^\top (\bar{\mathbf{x}}_i^p - \bar{\mathbf{x}}_i) \right\|_2^2 \\ &= \left\| W_i^\top (\mathbf{x} - \bar{\mathbf{x}}_i^p) \right\|_2^2 \end{aligned} \tag{11}$$

where $\bar{\mathbf{x}}_i^p$ is the mean of the positive samples used in training $\mathcal{A}_i$, $\bar{\mathbf{x}}_i$ is the mean of all the samples in training $\mathcal{A}_i$, and $\|\cdot\|_2$ is the Euclidean norm.

E. Adaptive Appearance Model

The target and background appearances may change due to factors such as illumination, pose, occlusion, camera motion, and so on. To deal with this problem, we propose an adaptive object representation method. Let the current set of appearance models be $\mathscr{A} = \{\mathcal{A}_i | i = 1, \ldots, k, k \leq K\}$. When the tracking result at time t is obtained, we use the corresponding target observation $\mathbf{x}_t$ to update $\mathscr{A}$. Since we have computed the distances from the target observation $\mathbf{x}_t$ to all the appearance models in $\mathscr{A}$ for determining the tracking result, we select the appearance model $\mathcal{A}_s$ with the smallest distance d_s and appearance model $\mathcal{A}_l$ with the largest distance d_l. If d_s is less than a predefined threshold, $\mathbf{x}_t$ is utilized to update $\mathcal{A}_s$. The update process includes three components: the mean of the positive samples $\bar{\mathbf{x}}_s^p$, the mean of all the training samples $\bar{\mathbf{x}}_s$, and the weight matrix W_s. The mean of the positive examples $\bar{\mathbf{x}}_s^p$ can be updated by using a small forgetting factor. The negative samples at time t are collected using the method presented in Section III-B. Both $\bar{\mathbf{x}}_s$ and W_s can be updated by PLS analysis with the positive and negative samples. If d_s is larger than the predefined threshold and $k < K$, a new appearance model $\mathcal{A}_{k+1}$ is added to $\mathscr{A}$. If d_s is larger than the predefined threshold and $k = K$, a new appearance model is initialized to replace $\mathcal{A}_l$ in $\mathscr{A}$. The proposed adaptive appearance model is summarized in Algorithm 1, where T is the number of frames. With our method, only the appearance model set $\mathscr{A}$ is maintained and all the previous target and background observations are discarded.

Algorithm 1 Proposed Adaptive Appearance Model

```
Initialize A_1 with PLS analysis when t = 1.
for t = 2 : T do
  Find A_s and A_l in 𝒜 which have the smallest and largest
  distances d_s and d_l to the target sample x_t, respectively.
  if d_s < Threshold (e.g., (1/5)||x_t||^2) then
    x̄_s^p ← f x̄_s^p + (1 − f) x_t (f is a forgetting factor).
    Update x̄_s and W_s using PLS analysis.
  else
    if k < K then
      Learn a new appearance model A_{k+1}, and add
      it to 𝒜.
    else
      Learn a new appearance model and use it to replace
      A_l in 𝒜.
    end if
  end if
end for
```

Fig. 4. Appearance variation of a target object during tracking.

IV. Two-Stage Tracking Method

Based on the proposed adaptive appearance model, object tracking is carried out using the Bayesian inference framework. In order to reduce visual drift, we present a two-stage particle filtering method to estimate the tracking result using both the initial and adaptive appearance models.

A. Particle Filtering

Given the observation set of the target $\mathbf{x}_{1:t} = [\mathbf{x}_1, \ldots \mathbf{x}_t]$ up to time t, the tracking result $\mathbf{s}_t$ can be determined by the Maximum A Posteriori (MAP) estimation, $\hat{\mathbf{s}}_t = \arg\max p(\mathbf{s}_t | \mathbf{x}_{1:t})$, where $p(\mathbf{s}_t | \mathbf{x}_{1:t})$ is inferred by the Bayes theorem recursively with $p(\mathbf{s}_t | \mathbf{x}_{1:t}) \propto p(\mathbf{x}_t | \mathbf{s}_t) p(\mathbf{s}_t | \mathbf{x}_{1:t-1})$ and $p(\mathbf{s}_t | \mathbf{x}_{1:t-1}) = \int p(\mathbf{s}_t | \mathbf{s}_{t-1}) p(\mathbf{s}_{t-1} | \mathbf{x}_{1:t-1}) d\mathbf{s}_{t-1}$. This inference is governed by the dynamic model $p(\mathbf{s}_t | \mathbf{s}_{t-1})$ which describes the temporal correlation of the tracking results in consecutive frames, and the likelihood function (i.e., observation model) $p(\mathbf{x}_t | \mathbf{s}_t)$ which denotes the likelihood of $\mathbf{s}_t$ observing $\mathbf{x}_t$.

With particle filtering, the posterior $p(\mathbf{s}_t | \mathbf{x}_{1:t})$ can be approximated by a finite set of N_s samples $\{\mathbf{s}_t^i | i = 1, ..., N_s\}$ with importance weights $\{\omega_t^i | i = 1, ... N_s\}$. The candidate sample $\mathbf{s}_t^i$ is drawn from an importance distribution $q(\mathbf{s}_t | \mathbf{s}_{1:t-1}, \mathbf{x}_{1:t})$ and the weight of the i-th sample is $\omega_t^i = \omega_{t-1}^i \frac{p(\mathbf{x}_t | \mathbf{s}_t^i) p(\mathbf{s}_t^i | \mathbf{s}_{t-1}^i)}{q(\mathbf{s}_t^i | \mathbf{s}_{1:t-1}^i, \mathbf{x}_{1:t})}$, where $q(\mathbf{s}_t | \mathbf{s}_{1:t-1}, \mathbf{x}_{1:t}) = p(\mathbf{s}_t | \mathbf{s}_{t-1})$ in this paper and the weight ω_t^i is equal to the observation likelihood $p(\mathbf{x}_t | \mathbf{s}_t^i)$.

The dynamic model $p(\mathbf{s}_t | \mathbf{s}_{t-1})$ generates a set of plausible hypotheses for efficient computation. In this paper, we use a random walk model for its simplicity. We model the

motion of a target between two consecutive frames with affine transform and the target state at time t is denoted by $\mathbf{s}_t = (x_t, y_t, s_t, \alpha_t, \theta_t)$, where $x_t, y_t, s_t, \alpha_t, \theta_t$ are x, y translations, scale, aspect ratio, and in-plane rotation angle, respectively. Each parameter in $\mathbf{s}_t$ is modeled independently with a Gaussian distribution based its previous state parameter $\mathbf{s}_{t-1}$, $p(\mathbf{s}_t|\mathbf{s}_{t-1}) = \mathcal{N}(\mathbf{s}_t; \mathbf{s}_{t-1}, \Sigma)$, where Σ is a diagonal covariance matrix whose diagonal elements are the corresponding variances of respective parameters. Within the Bayesian inference framework, the most important issue for tracking is the likelihood function as it corresponds to the main challenge, i.e., determining the image patch most similar to the current appearance model. The adaptive likelihood function $p(\mathbf{x}_t|\mathbf{s}_t)$ is computed based on our adaptive appearance model. With distance metric defined in (10), the likelihood function is computed by

$$p(\mathbf{x}_t|\mathbf{s}_t) \propto \exp(-d_t) \tag{12}$$

where d_t is the distance between the test sample $\mathbf{x}_t$ and the learned appearance model set $\mathscr{A}$ at time t. The likelihood function adapts over time as a result of the proposed appearance model with online update.

B. Two-Stage Particle Filtering for Object Tracking

The main issue for any adaptive appearance model is that it is likely to use noisy or mis-aligned observations for update, thereby causing tracking drift gradually. For online tracking, the only ground truth at our disposal is the labeled target object in the first frame. All the other samples obtained online are most likely different from the ground truth data. To reduce tracking drift, we present a two-stage particle filtering method for state prediction.

In our method, the appearance model $\mathcal{A}_1 = \{\bar{\mathbf{x}}_1^p, \bar{\mathbf{x}}_1, W_1\}$ initialized in the first frame is used to construct a static likelihood function (we omit t for clarify of presentation), $p^s(\mathbf{x}|\mathbf{s}) \propto \exp(-d^s)$, where $d^s = \left\| W_1^\top (\mathbf{x} - \bar{\mathbf{x}}_1^p) \right\|_2^2$. The adaptive appearance model set $\mathscr{A}$ is used to construct another likelihood function (we omit t for clarify of presentation), $p^a(\mathbf{x}|\mathbf{s}) \propto \exp(-d^a)$, where $d^a = \min\{d_i^a | i = 1, \ldots, k\}$ and d_i^a is the distance from $\mathbf{x}$ to the i-th appearance model $\mathcal{A}_i$, i.e., $d_i^a = \left\| W_i^\top (\mathbf{x} - \bar{\mathbf{x}}_i^p) \right\|_2^2$.

At each frame, we first use a particle filter with the adaptive likelihood function $p^a(\mathbf{x}|\mathbf{s})$ to estimate an initial tracking result. With the initial estimate, we use another particle filter with the static likelihood function $p^s(\mathbf{x}|\mathbf{s})$ to determine the final predicted state in the second stage. The two-stage tracking algorithm is summarized in Algorithm 2, where T is the number of frames. The filter with adaptive likelihood function can effectively avoid the local minimum problem since the appearance change between two consecutive frames is not expected to be too large. The filter with static likelihood function can effectively alleviate the drift problem since it requires that the final tracking result to be as similar as the only ground truth obtained in the first frame. Similar strategy has also been successfully demonstrated to reduce drift [19].

Algorithm 2 Two-Stage Tracking Algorithm

Input: Image frames $F_1, \ldots, F_T$.
Output: Tracking results $\hat{\mathbf{s}}_t$ at time t.
for $t = 1, \ldots, T$ **do**
  if $t = 1$ **then**
    Label the target manually or using a detector. Collect positive as well as negative samples, and compute the static appearance model $\mathcal{A}_1 = \{\bar{\mathbf{x}}_1^p, \bar{\mathbf{x}}_1, W_1\}$.
  else
    Stage 1. Perform particle filtering to estimate an initial result $\mathbf{s}_t$ using the previous tracking result $\hat{\mathbf{s}}_{t-1}$ and the adaptive likelihood function $p^a(\mathbf{x}|\mathbf{s})$.
    Stage 2. Perform particle filtering to determine the final tracking result $\hat{\mathbf{s}}_t$ with the initial tracking result $\mathbf{s}_t$ and the static likelihood function $p^s(\mathbf{x}|\mathbf{s})$.
    Output the tracking result $\hat{\mathbf{s}}_t$.
    Update the adaptive appearance model set $\mathscr{A}$ with $\hat{\mathbf{s}}_t$ using Algorithm 1.
  end if
end for

V. Experiments

We demonstrate the merits of the proposed algorithm on challenging image sequences with evaluation against several state-of-the-art tracking methods. The challenging factors in these sequences include occlusion, image blur, camera motion, change of pose, illumination, and scale, among others.

A. Implementation Details

In each experiment, the target object is manually labeled in the first frame. Each image observation of the target object is normalized to a 32×32 patch and represented by 1024-dimensional vector of intensity values. To initialize an appearance model, we use one positive sample and 30 negative samples for PLS analysis. The number of weight vectors, p, is set to 10, and the maximum number of appearance models K is set to 5, empirically. In addition, the forgetting factor f is empirically set to 0.8 and 600 samples are drawn for particle filtering. Implemented in MATLAB on an Intel Core E2180 2.0 GHz computer with 2 GB RAM and no code optimization, it takes about one second to process each frame.

B. Performance Evaluation

We use 12 sequences to evaluate our algorithm with 8 state-of-the-art online tracking algorithms: adaptive discriminative generative tracker (ADT) [18], incremental visual tracking method (IVT) [24], variance ratio tracking method (VRT) [7], online boosting method (BOT) [10], ℓ_1 minimization tracking method (L1T) [20], multiple instance learning tracker (MIL) [3], visual tracking decomposition algorithm (VTD) [16], tracking-learning-detection method (TLD) [15], and relevance vector machine tracker (RVM) [27]. We implement another adaptive tracking method called adaptive template tracking (ATT) in which the template is updated online using the data mean update method proposed in IVT. The observation model of the ATT method is defined by

$p(\mathbf{x}|\mathbf{s}) \propto \exp(-\|\mathbf{x}-\mathbf{t}\|_2^2)$, where $\mathbf{t}$ is the adaptive template, and all the other components are the same as IVT except subspace learning.

1) Adaptive Object Trackers: All the evaluated tracking algorithms use either generative (i.e., IVT, ATT, and L1T) or discriminative (i.e., ADT, VRT, BOT, MIL, VTD, TLD and RVM) representation schemes. For ADT, IVT, ATT, L1T methods and the proposed tracker, we use the *same* feature vector of intensity values for object description, and use the *same* parameters (including the dynamic model and the number of particles) for particle filtering. The VRT method selects discriminative features for object representation from a set of color spaces and implements tracking using the mean shift algorithm [8]. The BOT algorithm selects from a set of Haar-like, HOG and LBP features for object representation and online tracking. In the MIL tracker, the generalized Haar-like features are adopted with an online multiple instance learning algorithm to reduce visual drift. The VTD system combines multiple observation models (based on hue, saturation, intensity, and edge features) and multiple dynamic models to account for the appearance and motion change in object tracking. In the TLD method, Haar-like features are employed and the underlying structure of image observations obtained online is exploited to alleviate the drift problem. The RVM method learns a regressor online by a probabilistic SVM directly from the intensity values, and utilizes an object detector in tandem for automatic initialization and recovery. Since the IVT and ATT algorithms update the appearance models by combining newly arrived data and previous observations with a forgetting factor, they are able to deal with tracking drift to some degree. For fair comparison, we set all the test trackers with the *same* initialization parameters.

2) Evaluation Criteria: Performance evaluation of object tracking is an important and challenging problem. In this paper, we evaluate the above-mentioned trackers qualitatively and quantitatively. For qualitative assessments, we present representative tracking results from each video sequence. For quantitative evaluation, we measure the tracking success rate and center location error using the ground truth object locations obtained by manual labels at every 5 frames. We employ the criterion used in the PASCAL VOC challenge [9] to determine whether each tracking result is a success. Given the tracked bounding box ROI_T and the ground truth bounding box ROI_G, the score is defined as $\text{score} = \frac{\text{area}(\mathrm{ROI_T} \bigcap \mathrm{ROI_G})}{\text{area}(\mathrm{ROI_T} \bigcup \mathrm{ROI_G})}$. The tracking result in one frame is considered as a success when this score is above 0.5. Table I shows the tracking results in terms of success rates. The center location error is defined as the distance between the central locations of the tracked target and the ground truth. The tracking results in terms of center location errors are illustrated in Fig. 5, and the average errors are presented in Table II.

Illumination: In the *car4* sequence [24] shown in Fig. 6, there is significant illumination change when the car passes beneath the overpass and trees. The scale change of the target and camera movement also make this sequence challenging. The IVT, ATT, TLD, RVM methods and our algorithm perform well in tracking all or most of the frames in this sequence while the others do not. The tracking errors of these four trackers are also lower than those of the other methods. The IVT and ATT methods use all the previous target observations for appearance modeling, our method makes use of a static observation model, and the TLD maintains a detector along with the adaptive tracker. That is why these trackers are less sensitive to drift after the illumination changes. In the *car11* sequence [24], the contrast between the target object and the background is low and the ambient light changes significantly. Furthermore, the low image resolution of the target object makes tracking difficult. Fig. 7 shows some results where the IVT, ATT, BOT, VTD trackers and our method are able to track the target with low center location errors. The BOT and VTD methods perform well as the HOG and edge features are less sensitive to illumination change. The VRT method does not work well since it is difficult to find discriminative features as the color distributions of the target object and background are similar. These two experiments demonstrate that our method is able to handle drastic illumination changes.

Pose: In the *Gym* sequence, the target object undergoes out-of-plane pose change and shape deformation. Some tracking results shown in Fig. 8. The quantitative results shown in Tables I–II as well as Fig. 5 indicate that the

TABLE I
SUCCESS RATES (%). THE BEST TWO RESULTS ARE PRESENTED WITH BOLD FACE AND ITALIC FONTS

	ADT	IVT	ATT	VRT	BOT	L1T	MIL	VTD	TLD	RVM	Ours
car4	76	**100**	**100**	33	32	33	36	50	*96*	**100**	**100**
car11	23	**100**	**100**	0	96	65	13	*98*	51	47	**100**
Gym	16	43	43	**88**	20	7	47	71	56	**88**	*81*
surfer	4	86	79	28	**100**	83	*99*	96	85	31	**100**
Sylvester	39	45	44	72	78	36	68	79	*86*	30	**88**
faceocc2	92	93	**97**	3	85	65	89	57	81	52	*94*
Mei	70	30	29	27	15	43	14	*77*	36	33	**97**
girl	3	34	35	27	72	38	14	38	*63*	37	**95**
square1	11	31	31	66	23	30	52	31	46	*81*	**91**
square2	50	50	41	9	91	29	95	*96*	95	30	**100**
Wall-E	59	11	11	8	8	51	8	20	*70*	14	**90**
chasing	5	61	*62*	11	8	29	8	9	59	19	**65**

TABLE II
AVERAGE CENTER LOCATION ERRORS (IN PIXELS). THE BEST TWO RESULTS ARE PRESENTED WITH BOLD FACE AND ITALIC FONTS

	ADT	IVT	ATT	VRT	BOT	L1T	MIL	VTD	TLD	RVM	Ours
car4	14	**3**	**3**	116	46	71	52	76	13	*6*	*6*
car11	25	*3*	*3*	79	4	28	41	4	29	32	**2**
Gym	55	19	19	10	15	123	16	*9*	12	**7**	10
surfer	43	7	9	65	**4**	9	*5*	6	9	39	**4**
Sylvester	24	60	58	14	10	19	13	*9*	10	63	**8**
faceocc2	*12*	**9**	**9**	66	22	33	16	53	**9**	30	**9**
Mei	18	19	20	19	42	32	33	*12*	18	68	**10**
girl	139	39	39	39	*18*	31	43	36	24	35	**7**
square1	35	92	89	**5**	39	95	11	80	13	20	*8*
square2	27	22	24	45	**4**	73	*6*	*6*	7	105	**4**
Wall-E	42	21	22	21	52	**14**	28	*18*	28	41	**14**
chasing	27	8	8	128	34	30	43	34	**5**	61	*7*

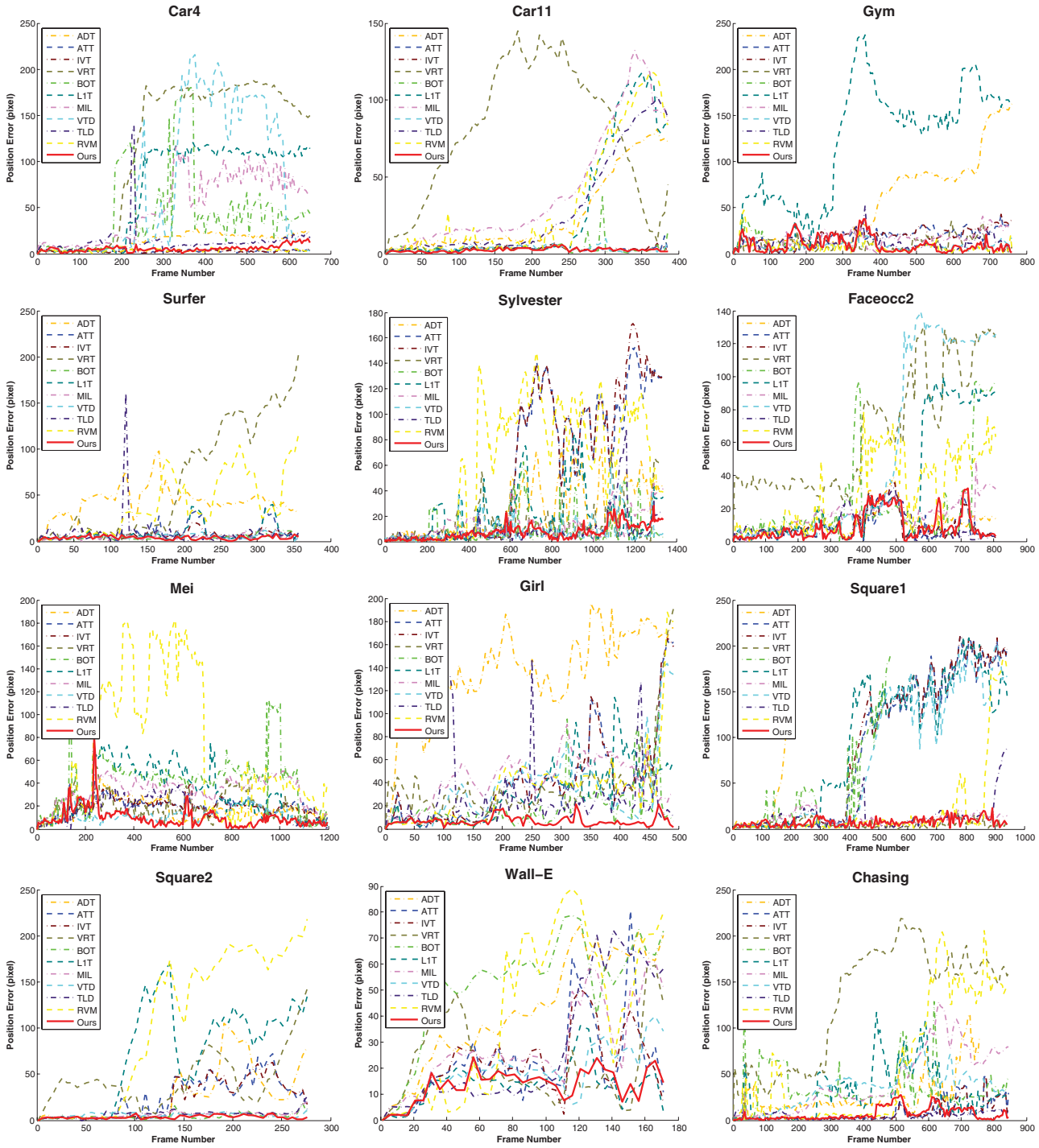

Fig. 5. Error plots of test sequences.

VRT, RVM and the proposed methods perform better than the other trackers. The L1T method does not perform well in this sequence as the appearance change due to shape deformation is not effectively accounted for by the holistic sparse representation. For the *surfer* video shown in Fig. 9, the target moves with out-of-plane pose change. The BOT, MIL, VTD and the proposed methods are able to tack more image frames than the ADT, IVT, ATT, L1T, TLD and RVM methods. From the error plot shown in Fig. 5 and the average location error illustrated in Table II, it is clear that all the trackers except ADT and VRT perform well in tracking the center location of the target. In the *Sylvester* sequence [24], the target object undergoes large pose and illumination change. Some representative tracking results are shown in Fig. 10. Our tracker and the TLD method achieve higher success rates than the others as shown in Table I. Fig. 5 and Table II demonstrate that the VRT, BOT, MIL, VTD, TLD trackers and our method achieve relatively lower center location errors than the other methods. In these three

Fig. 6. Tracking results of the *Car4* sequence.

Fig. 7. Tracking results of the *Car11* sequence.

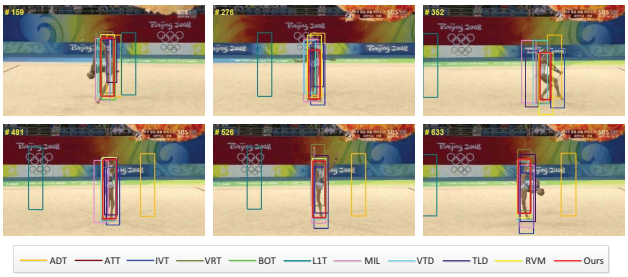

Fig. 8. Tracking results of the *Gym* sequence.

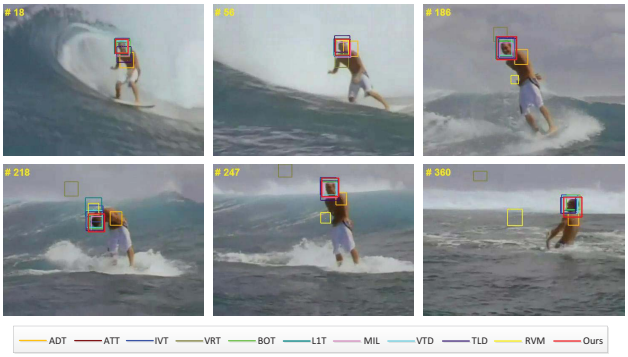

Fig. 9. Tracking results of the *Surfer* sequence.

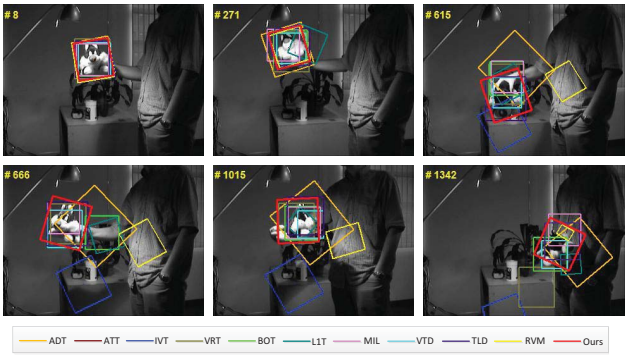

Fig. 10. Tracking results of the *Sylvester* sequence.

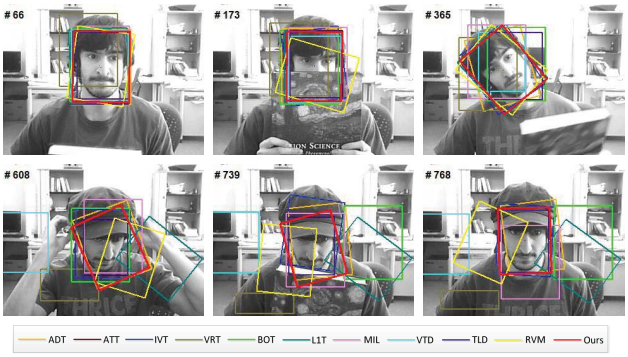

Fig. 11. Tracking results of the *Faceocc* sequence.

sequences, the out-of-plane pose change leads to nonlinear appearance variation of the object. As our method makes use of multiple models to represent such nonlinear variation, it deals with the pose change problem well.

Occlusion: In the *faceocc2* sequence [3], the target object undergoes occlusion and in-plane pose change. Some tracking results are shown in Fig. 11. Overall, the ADT, IVT, ATT methods and our algorithm perform well. For the *Mei* sequence [13], there are illumination, scale, and pose changes aside from occlusion. The proposed tracking algorithm is able to track the target well and some tracking results are shown in Fig. 12. The ADT and VTD methods also have relatively high success rates, and the IVT, ATT, VRT and TLD methods are also able to track the target location well.

In the *girl* sequence [4], the target object undergoes heavy occlusion, large pose change, and scale variation. Some tracking results are shown in Fig. 13. The error plot (Fig. 5) and the average center location error (Table II) show that our method achieves the best performance in this sequence. As the holistic sparse representation method cannot deal with heavy occlusions and there is no drift alleviation mechanism, the L1T method does not work very well on these sequences. In contrast, our method not only represents the appearance variations with several models but also uses a static observation model to recapture the target after severe occlusions, thereby generating better results on these sequences.

Image blur: In the *square1* sequence, the target object undergoes significant scale change, image blur and occlusion. The error plot shown in Fig. 5 and the average location error illustrated in Table II indicate that the VRT, MIL and the

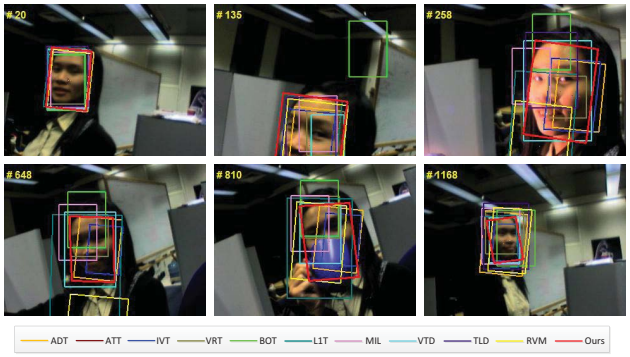

Fig. 12. Tracking results of the *Mei* sequence.

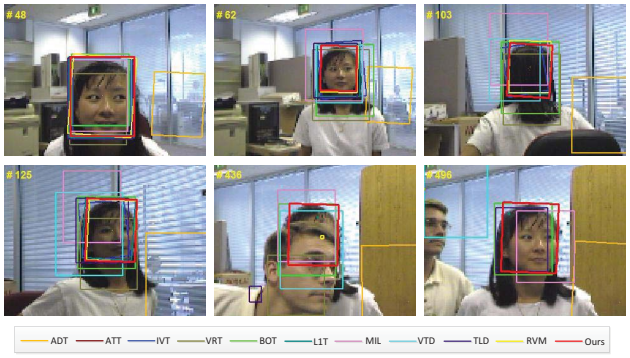

Fig. 13. Tracking results of the *Girl* sequences.

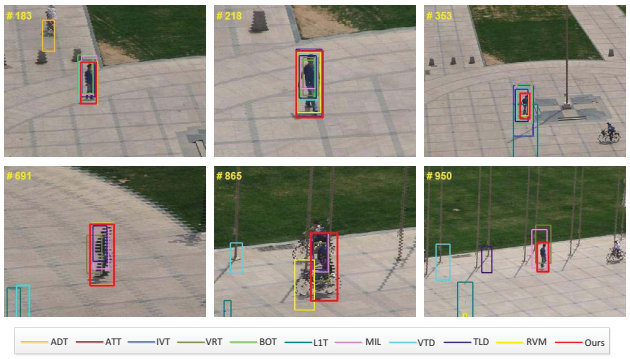

Fig. 14. Tracking results of the *Square1* sequence.

Fig. 15. Tracking results of the *Square2* sequence.

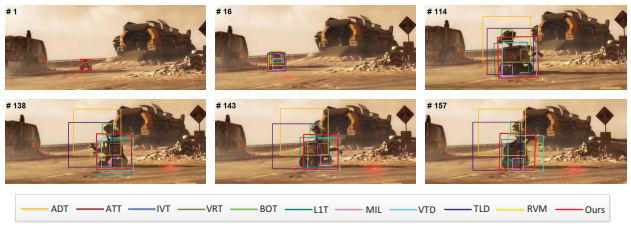

Fig. 16. Tracking results of the *Wall-E* sequence.

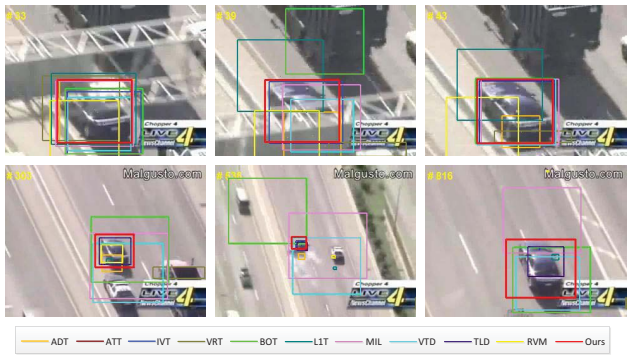

Fig. 17. Tracking results of the *Chasing* sequence.

proposed methods perform well. However, the success rates of the VRT and MIL algorithms are low since they do not estimate the scale change of the target object well. The RVM method also performs well with high success rate although it loses the target after the occlusion (shown in frame 865 of Fig. 14). The target object in the *square2* sequence (Fig. 15) is difficult to track as it moves through the scene with motion blur and partial occlusion. Overall, the BOT, MIL, VTD, TLD and our tracking method perform well with higher success rates and lower location errors. While the ADT, IVT and ATT algorithms perform better than the VRT, L1T and RVM methods, they lose track of the target object when image blur occurs (frame 139). Our method can deal with image blur as it extracts discriminative feature subspace with the PLS analysis. Aside from this, our method also makes use of the initial appearance model to alleviate the drift problem.

Scale: The object in the *Wall-E* sequence exhibits drastic change in scale and out-of-plane rotation. The color similarity between the target and the background also makes this sequence difficult for object tracking. Some qualitative tracking results are illustrated in Fig. 16. Overall, our method performs much better than other methods. The L1T algorithm perform well in terms of location error but with much lower success rate than the proposed method. In addition, the IVT, ATT, VRT, and VTD methods also perform reasonably well in terms of location errors but with low success rates, which suggests that these trackers do not deal with object scale change well. In the *chasing* sequence (Fig. 17), the target object undergoes significant scale change and heavy occlusion. The fast appearance change of background due to movement of the camera and the target object also makes it difficult to track this vehicle. From the success rates and center location

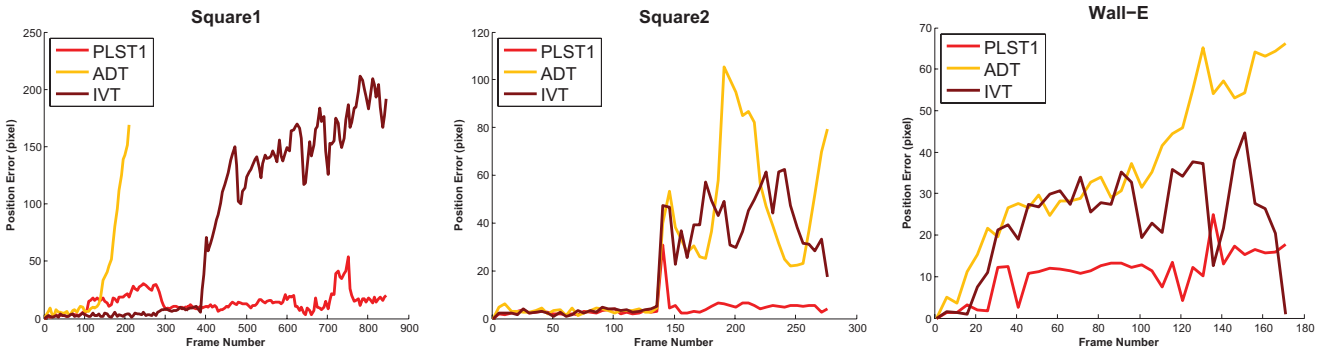

Fig. 18. Error plots of the PLS1, ADT, and IVT methods.

errors, the IVT, ATT, TLD trackers and our algorithm perform better than the other methods although all these trackers lose track of the target object for a number of frames. Since our method estimates the affine motion parameters of the target object, it can deal with scale change well.

Discussion: To isolate the merits of PLS over PCA and FLD, we develop an algorithm (referred as PLST1) which only learns one appearance model with PLS analysis for object tracking. We choose the data mean used in the IVT method as our positive sample, and select negative samples in the current frame for appearance modeling. All the other components of the PLST1 algorithm is the same as the IVT method. That is, there is no two-stage tracking mechanism in PLST1. We evaluate the PLST1 method on the *square1*, *square2* and *Wall-E* sequences, and the error plots are shown in Fig. 18. Although the PLST1 method does not perform as well as our proposed tracking method, it still performs better than the ADT and IVT methods. It demonstrates that the PLS method is more suitable for object tracking than the FLD and PCA methods, and the two-stage tracking mechanism can further improve the tracking performance. Furthermore, the above results shown in Fig. 5, Table I and II also verify that the PCA method does not essentially improve tracking performance compared to the adaptive template method (ATT).

VI. Conclusion

We present a tracking algorithm in which an object is represented by multiple appearance models learned online using partial least squares analysis. The proposed algorithm utilizes an adaptive discriminative representation to account for the nonlinear appearance change of an object over time. To reduce tracking drift, a two-stage particle filtering method is presented which makes use of both the static appearance information obtained at the outset and image observations acquired online. Compared with state-of-the-art tracking methods, the proposed algorithm achieves favorable performance with higher success rates and lower tracking errors.

References

[1] S. Avidan, "Support vector tracking," in *Proc. IEEE Comput. Vis. Pattern Recognit. Conf.*, vol. 1. Jerusalem, Israel, Apr. 2001, pp. 184–191.

[2] S. Avidan, "Ensemble tracking," in *Proc. IEEE Comput. Vis. Pattern Recognit. Conf.*, vol. 2. Cambridge, MA, Jun. 2005, pp. 494–501.

[3] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Proc. IEEE Comput. Vis. Pattern Recognit. Conf.*, San Diego, CA, Jun. 2009, pp. 983–990.

[4] S. Birchfield, "Elliptical head tracking using intensity gradients and color histograms," in *Proc. IEEE Comput. Vis. Pattern Recognit. Conf.*, Santa Barbara, CA, Jun. 1998, pp. 232–237.

[5] M. J. Black, D. J. Fleet, and Y. Yacoob, "A framework for modeling appearance change in image sequences," in *Proc. IEEE Comput. Vis. Pattern Recognit. Conf.*, Bombay, India, Jan. 1998, pp. 660–667.

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

[7] R. T. Collins, Y. Liu, and M. Leordeanu, "Online selection of discriminative tracking features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1631–1643, Oct. 2005.

[8] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 5, pp. 564–575, May 2003.

[9] M. Everingham, L. V. Gool, C. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2010.

[10] H. Grabner and H. Bischof, "On-line boosting and vision," in *Proc. IEEE Comput. Vis. Pattern Recognit. Conf.*, Jun. 2006, pp. 260–267.

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

[12] G. D. Hager and P. N. Belhumeur, "Efficient region tracking with parametric models of geometry and illumination," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 10, pp. 1025–1039, Oct. 1998.

[13] J. Ho, K.-C. Lee, M.-H. Yang, and D. Kriegman, "Visual tracking using learned linear subspaces," in *Proc. IEEE Comput. Vis. Pattern Recognit. Conf.*, vol. 1. Jul. 2004, pp. 782–789.

[14] A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi, "Robust online appearance models for visual tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 10, pp. 1296–1311, Oct. 2003.

[15] Z. Kalal, J. Matas, and K. Mikolajczyk, "P-N learning: Bootstrapping binary classifiers by structural constraints," in *Proc. IEEE Comput. Vis. Pattern Recognit. Conf.*, San Francisco, CA, Jun. 2010, pp. 49–56.

[16] J. Kwon and K. Lee, "Visual tracking decomposition," in *Proc. IEEE Comput. Vis. Pattern Recognit. Conf.*, San Francisco, CA, Jun. 2010, pp. 1269–1276.

[17] X. Li, W. Hu, Z. Zhang, X. Zhang, and G. Luo, "Robust visual tracking based on incremental tensor subspace learning," in *Proc. IEEE Comput. Vis. Pattern Recognit. Conf.*, Rio de Janeiro, Brazil, Oct. 2007, pp. 1–8.

[18] R.-S. Lin, D. A. Ross, J. Lim, and M.-H. Yang, "Adaptive discriminative generative model and its applications," in *Proc. Adv. Neural Inf. Process. Syst. Conf.*, 2004, pp. 801–808.

[19] 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.

[20] X. Mei and H. Ling, "Robust visual tracking using ℓ_1 minimization," in *Proc. IEEE Int. Comput. Vis. Conf.*, Sep. 2009, pp. 1436–1443.

[21] D. Nguyen and D. Rocke, "Tumor classification by partial least squares using microarray gene expression data," *Bioinformatics*, vol. 18, no. 1, pp. 39–50, 2002.

[22] R. Rosipal and N. Kramer, "Overview and recent advances in partial least squares," in *Latent Structures Feature Selection*. New York: Springer-Verlag, 2006, pp. 34–51.

[23] R. Rosipal and L. Trejo, "Kernel partial least squares regression in reproducing kernel Hilbert space," *J. Mach. Learn. Res.*, vol. 2, pp. 97–123, Mar. 2002.
[24] D. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *Int. J. Comput. Vis.*, vol. 77, nos. 1–3, pp. 125–141, 2008.
[25] H. Saigo, N. Kramer, and K. Tsuda, "Partial least squares regression for graph mining," in *Proc. ACM SIGKDD Int. Knowl. Discovery Data Mining Conf.*, 2008, pp. 578–586.
[26] W. Schwartz, A. Kembhavi, D. Harwood, and L. Davis, "Human detection using partial least squares analysis," in *Proc. IEEE Int. Comput. Vis. Conf.*, Oct. 2009, pp. 24–31.
[27] O. Williams, A. Blake, and R. Cipolla, "Sparse bayesian learning for efficient visual tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1292–1304, Aug. 2005.
[28] H. Wold, "Partial least squares," in *Encyclopedia of Statistical Science*, vol. 6, S. Kotz and N. L. Johnson, Eds. New York: Wiley, 1985, pp. 581–591.

Qing Wang (S'10) received the B.S. degree in automation from Northwestern Polytechnical University, China, in 2007. He is currently pursuing the Ph.D. degree with the Department of Automation, Tsinghua University, Beijing, China.

His current research interests include computer vision and machine learning.

Feng Chen (M'06) received the B.S. and M.S. degrees in automation from Saint-Petersburg Polytechnic University, St-Petersburg, Russia, in 1994 and 1996, respectively, and the Ph.D. degree in automation from Tsinghua University, Beijing, China, in 2000.

He is a Professor with Tsinghua University. His current research interests include probabilistic graphical models and computer vision.

Wenli Xu received the B.S. degree in electrical engineering and the M.S. degree in automatic control engineering from Tsinghua University, Beijing, China, in 1970 and 1980, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Colorado, Boulder, in 1990.

He is a Professor with Tsinghua University. His current research interests include automatic control, intelligent robots, and computer vision.

Ming-Hsuan Yang (SM'06) received the Ph.D. degree in computer science from the University of Illinois at Urbana-Champaign, Urbana, in 2000.

He is an Assistant Professor of electrical engineering and computer science with the University of California, Merced (UC Merced). Prior to joining UC Merced, he was a Senior Research Scientist with Honda Research Institute, Mountain View, CA. He has co-authored the book *Face Detection and Gesture Recognition for Human-Computer Interaction* (Kluwer Academic, 2001) and edited special issues on face recognition for *Computer Vision and Image Understanding* and the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE.

Dr. Yang was an Associate Editor of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE from 2007 to 2011 and is an Associate Editor of the *Image and Vision Computing*. He was the recipient of the National Science Foundation CAREER Award in 2012, the Campus-Wide Senate Award for Distinguished Early Career Research at UC Merced in 2011, and the Google Faculty Award in 2009. He is a Senior Member of the Association for Computing Machinery.