Online Tracking and Reacquisition Using Co-trained Generative and Discriminative Trackers

Qian Yu, Thang Ba Dinh, and Gérard Medioni

University of Southern California, Los Angeles, CA 90089-0273, USA
{qianyu,thangdin,medioni}@usc.edu

Abstract. Visual tracking is a challenging problem, as an object may change its appearance due to viewpoint variations, illumination changes, and occlusion. Also, an object may leave the field of view and then reappear. In order to track and reacquire an unknown object with limited labeling data, we propose to learn these changes online and build a model that describes all seen appearance while tracking. To address this semi-supervised learning problem, we propose a co-training based approach to continuously label incoming data and online update a hybrid discriminative generative model. The generative model uses a number of low dimension linear subspaces to describe the appearance of the object. In order to reacquire an object, the generative model encodes all the appearance variations that have been seen. A discriminative classifier is implemented as an online support vector machine, which is trained to focus on recent appearance variations. The online co-training of this hybrid approach accounts for appearance changes and allows reacquisition of an object after total occlusion. We demonstrate that under challenging situations, this method has strong reacquisition ability and robustness to distracters in background.

1 Introduction

Object tracking is challenging [1] due to appearance changes, which can be caused by varying viewpoints and illumination conditions. Appearance can also change relative to background due to the emergence of clutter and distracters. Also, an object may leave the field of view (or be occluded) and reappear. To address these difficulties, we aim to track an arbitrary object with limited initialization (labeled data) and learn an appearance model on-the-fly, which can then be used to reacquire the object when it reappears.

This tracking problem can be formulated in two different ways: generative and discriminative. Generative tracking methods learn a model to represent the appearance of an object. Tracking is then expressed as finding the most similar object appearance to the model. Several examples of generative tracking algorithms are Eigentracking [2], WSL tracking [3] and IVT [5]. To adapt to appearance changes, the object model is often updated online, as in [5]. Due to the fact that the appearance variations are highly non-linear, multiple subspaces [6] and non-linear manifold learning methods [7] have been proposed. Note that traditional generative tracking methods are trained based on object appearance without considering background information.

D. Forsyth, P. Torr, and A. Zisserman (Eds.): ECCV 2008, Part II, LNCS 5303, pp. 678–691, 2008.

Instead of building a model to describe the appearance of an object, discriminative tracking methods aim to find a decision boundary that can best separate the object from the background. Recently, many discriminative trackers are proposed [9,10,11] and demonstrate strong robustness to avoid distracters in the background. Support Vector Tracking (SVT)[8] integrates an offline trained Support Vector Machine (SVM) classifier into an optic-flow-based tracker. In order to update the decision boundary according to new samples and background, discriminative tracking methods with online learning are proposed in [10,9]. In [10], a confidence map is built by finding the most discriminative RGB color combination in each frame. However, a limited color feature pool restricts the discriminative power of this method. In [9], Avidan proposes to use an ensemble of online learned weak classifiers to label a pixel as belonging to either the object or the background. To accommodate object appearance changes, at every frame, new weak classifiers replace part of old ones that do not perform well or have existed longer than a fixed number of frames. Both methods [9,10] use features at the pixel level and rely on a mode seeking process (mean shift) to find the best estimate on a confidence map, which restricts the reacquisition ability of these methods. Oza and Russell [12] proposed an online boosting algorithm, which is applied to the visual tracking problem [13,14]. Due to the large number of features, either an offline feature selection procedure or an offline trained seed classifier is usually required in practice. Thus, for tracking methods based on online boosting, it is difficult to generalize to arbitrary object types

It has been shown that discriminative classifiers often outperform generative models [15] if enough training data is available. However, generative methods often have better generalization performance when the size of training data is small. Specifically, a simple generative classifier (naive Bayes) outperforms its discriminative counterpart (logistic regression) when the amount of labeled training data is small [16]. Recently, hybrid discriminative generative methods have opened a promising direction to benefit from both types of methods. Several hybrid methods [17,18,19,15] have been proposed in many application domains [17,18,19]. Most of them imbue generative methods with the discriminative power via "discriminative training" of a generative model. These methods train a model by optimizing a convex combination of the generative and discriminative log likelihood functions. Due to the asymmetry in training data, "discriminative training" of a generative model requires a parameter to govern the trade-off between generative and discriminative. Theoretical discussions in [15] show that an improper hybrid of discriminative generative model generates even worse performance than pure generative or discriminative methods.

We propose to use co-training to combine generative and discriminative models. Here, the online learning an appearance model of an arbitrary object with limited labeled data is treated as a semi-supervised problem. The co-training approach proposed by Blum and Mitchell [20] is a principled semi-supervise training method. The basic idea is to train two classifiers on two conditionally independent views of the same data (with a small number of exemplars) and then use the prediction from each classifier to enlarge the training set of the other. It is proved that co-training can find an accurate decision boundary, starting from a small quantity of labeled data as long as the two feature sets are independent [20]. Empirical results [21] show that co-training works well

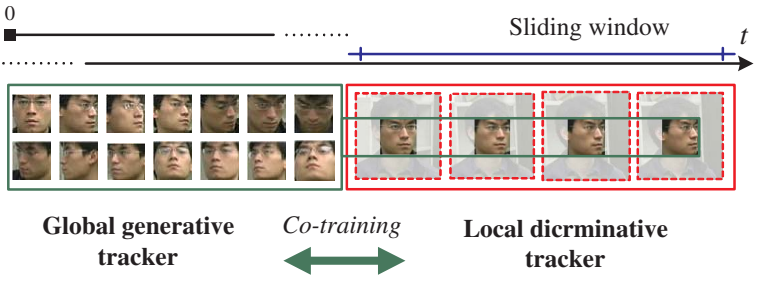


Fig. 1. Online co-training a generative tracker and a discriminative tracker with different life span (the area bounded by dashed red boxes indicates the background)

even in the case where the independence is not perfectly satisfied. In [22], the features used for classification are derived from PCA bases, which are obtained offline from training samples, and co-training is used to improve an offline learned object detector. More recently, Tang *et al.* [23] proposes to use co-training to online train two SVM trackers with color histogram features and HOG features. This method uses an incremental/decremental SVM solver[25] to focus on recent appearance variations without representing the global object appearance.

In order to represent the global object appearance, we propose to use a generative model, which contains a number of low dimension subspaces. This generative model encodes all the appearance variations that have been seen in a compact way. An online subspace updating algorithm is proposed to modify the subspaces adaptively. The descriptive power of the generative model increases as new samples are added. For the discriminative classifier, we use an incrementally learned SVM classifier [24] with histogram of gradient (HOG) [26] features. In practice, we find the number of support vectors grows quite fast when the appearance of object and background changes. Moreover, the adaption of the discriminative model to new appearance changes becomes more and more slow as samples are accumulated. To address these problems, we decrementally train the SVM to focus on recent appearance variations within a sliding window. The training data flow is shown in Figure 1. The image patches bounded with green boxes are samples used in the generative model. They contain all the object appearance variations since tracking starts. The red bounding boxes indicate the positive and negative training samples within the sliding window used in the discriminative classifier. The main advantage of this method is that it collaboratively combines the generative and discriminative models with complementary views (features) of the training data and it encodes the global object appearance variations, thus can handle reacquisition. Experiments show that our method has strong reacquisition ability and is robust to distracters in background clutter.

The rest of this paper is organized as follows. The overview of the co-training framework is presented in Section 2. The details of the generative tracker and the discriminative tracker are presented in Section 3 and Section 4. The experiments are shown in Section 5, followed by conclusions and future work.

2 Bayesian Inference with Co-training

We formulate the visual tracking problem as a state estimate problem in a similar way as [5,27]. Given a sequence of observed image regions $O_t = (o_1, \ldots, o_t)$ over time t, the goal of visual tracking is to estimate the hidden state s_t. In our case, the hidden state refers to an object's 2D position, scale and rotation. Assuming a Markovian state transition, the posterior can be formulated as a recursive equation

$$p(s_t|O_t) \propto p(o_t|s_t) \int p(s_t|s_{t-1})p(s_{t-1}|O_{t-1})ds_{t-1} \tag{1}$$

where $p(o_t|s_t)$ and $p(s_t|s_{t-1})$ are the observation model and state transition model respectively. $p(s_{t-1}|O_{t-1})$, which is represented as a set of particles and weights, is the posterior distribution given all the observations up to time $t-1$. The recursive inference in Eq.1 is implemented with resampling and importance sampling processes[27]. In our approach, the transition of the hidden state is assumed to be a Gaussian distribution as, $p(\mathbf{s}_t|\mathbf{s}_{t-1}) = \mathcal{N}(\mathbf{s}_t; \mathbf{s}_{t-1}, \Psi_t)$, where Ψ_t is a time variant diagonal covariance matrix. In this recursive inference formulation, $p(o_t|s_t)$ is the crucial part for finding the ideal posterior distribution. $p(o_t|s_t)$ measures the likelihood of observing o_t given one state of the object. Besides the 2D position, our state variables encode an object's rotation and scale. This reduces the appearance variations caused by such motion at the price of that more particles are needed to represent the distribution.

Our measurement of one observation comes from two independent models. One is the generative model, which is based on online constructed multi-subspaces. The other is the discriminative model, which is online trained with HOG features. The features used by these two models, namely intensity pattern and local gradient features, are complementary. After limited initialization, these two models are co-trained with sequential unlabeled data. It is worth noting that co-training is not a classification framework [21], but an automatic way to train with unlabeled data. In our approach, each model makes the decision based on its own knowledge and this information is used to train the other model. The final decision is made by the combined hybrid model. Due to the independence between the two observers, our observation model $p(o_t|s_t)$ can be expressed as a product of two likelihood functions from the generative $\mathcal{M}$ model and the discriminative model $\mathcal{C}$, $p(o_t|s_t) \propto p_{\mathcal{M}}(o_t|s_t)p_{\mathcal{C}}(o_t|s_t)$.

We adaptively adjust Ψ_t in the state transition model according to the tracking result at time t. If neither of the models accepts the new unlabeled data, we increase the covariance matrix Ψ_t and the number of samples. The extreme condition is that a very flat transition distribution is close to scanning the whole state space uniformly. This Bayesian formulation is very proper for the object tracking and reacquisition problem. By adaptively resampling, we can cover a large search region efficiently. Also, as partial appearance variations are compensated by the motion state, our method can deal with object rotation and scale changes. The key step in the co-training algorithm is to incrementally update both the generative and discriminative trackers, which are discussed in Section 3 and Section 4 respectively.

3 Generative Tracker with Multiple Linear Subspaces

The global appearance of one object under different viewpoints and illumination conditions is known to lie on a low dimension manifold. However, such a global appearance manifold is highly non-linear. In [7], a non-linear mapping from the embedding space to the input space is offline learned for tracking a specific object. Although the appearance manifold is globally non-linear, the local appearance variations can still be approximated as a linear subspace. Thus, we propose to incrementally learn a set of low dimension linear subspaces to represent the global appearance manifold. A multi-subspace representation is used in [6], where a fixed number of subspaces are offline built and are online updated with new samples.

Let $\mathcal{M} = \{\Omega_1, ..., \Omega_L\}$ represent the appearance manifold of one object and $\Omega_l, l \in [1, ..., L]$ denote the local sub-manifold. An appearance instance $\mathbf{x}$ is a d-dimension image vector. Let $\Omega_l = (\hat{\mathbf{x}}_\mathbf{l}, U_l, \Lambda_l, n_l)$ denote one sub-manifold, where $\hat{\mathbf{x}}_l, U_l, \Lambda_l$ and n_l represent the mean vector, eigenvectors, eigenvalues and the size (number of samples) of the subspace respectively. For simplicity, we omit the subscript when this causes no confusion. Here, $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)$ with sorted eigenvalues of the subspace, $\lambda_1 \geq \lambda_2 \cdots \geq \lambda_n$. A η-truncation is usually used to truncate the subspaces, namely $m = \arg\min_i (\sum_i \lambda_i / \text{tr}(\Lambda) \geq \eta)$. From a statistical point of view, a subspace with m eigenbases can be regarded as a m-dimensional Gaussian distribution. Suppose Ω is a subspace with the first m eigenvectors, the projection of $\mathbf{x}$ on Ω is $\mathbf{y} = (y_1, \ldots, y_m)^T = U^T(\mathbf{x} - \hat{\mathbf{x}})$. Then, the likelihood of $\mathbf{x}$ can be expressed [28] as

$$p(\mathbf{x}|\Omega) = \left[\frac{\exp\left(-\frac{1}{2}\sum_{i=1}^{m} \frac{y_i^2}{\lambda_i}\right)}{(2\pi)^{m/2} \prod_{i=1}^{m} \lambda_i^{1/2}} \right] \cdot \left[\frac{\exp\left(-\frac{\varepsilon^2(\mathbf{x})}{2\rho}\right)}{(2\pi\rho)^{(d-m)/2}} \right] \tag{2}$$

where $\varepsilon(\mathbf{x}) = |\mathbf{x} - UU^T\mathbf{x}|$ is the projection error, namely L_2 distance between the sample $\mathbf{x}$ and its projection on the subspace. The parameter $\rho = \frac{1}{d-m}\sum_{i=m+1}^{d} \lambda_i$ [28] or uses the $\frac{1}{2}\lambda_{m+1}$ as a rough approximation. By using Eq.2, we can evaluate the confidence of a sample from one subspace. As our generative model contains multiple subspaces (each subspace can be regarded as a hyper-ellipsoid), we maintain the neighborhood according to L_2 distance between the mean vectors of subspaces. To evaluate the confidence of one sample from such a generative model, we use the maximum confidence of the K-nearest (we use $K = 4$ in experiments) neighboring subspaces.

3.1 Online Subspace Learning

Given that samples are given in a sequential way, we aim to learn the low dimension linear subspaces incrementally. A new subspace is created with d_0 dimension, namely $d_0 + 1$ sequential samples form a new subspace. Local smoothness is guaranteed by a small d_0. A new subspace is created and added into the subspace pool. In order to represent a large number of sequential samples, we use a fixed number subspaces: if the number of subspaces exceeds a predetermined maximum, the most similar two subspaces are merged. The outline of the online subspace learning algorithm is shown in

Algorithm 1. Online Subspace learning algorithm

Input: ($\mathcal{I}$, $\mathcal{M}$, d_0,L)
$\mathcal{I} = \{I_1, \cdots, I_n, \cdots\}$: a sequence of samples $\mathcal{M} = \emptyset$: the appearance manifold
d_0: the initial dimension for each subspace L: the maximum number of subspaces
Output: $\mathcal{M} = (\Omega_1, \cdots \Omega_L)$: multi-local linear subspaces
while $\mathcal{I} \neq \emptyset$ **do**
  fetch $d_0 + 1$ samples and form a new subspace
  $\Omega_n \leftarrow (I_i, \cdots, I_{i+d_0})$
  if there exists an empty subspace **then**
    Add Ω_n to $\mathcal{M}$
  else
    $(p, q)^* = \arg\max Sim(\Omega_p, \Omega_q), p, q \in [1, \cdots, L], p \neq q$
    $\Omega_m = \Omega_p \cup \Omega_q$ and replace Ω_p and Ω_q with Ω_m
  end if
end while

Algorithm 1. In order to maintain the local property of the subspaces, merging only happens between neighboring subspaces. Merging of two subspaces and measuring the similarity between two subspaces are two critical steps in this algorithm.

Several methods have been proposed to incrementally update the eigenspaces. Only the method proposed by Hall *et al.* [29] takes into account the change of the mean of a subspace. This approach provides an exact solution to update an eigenspace and does not require storing original samples. Similar method was also used in [5,6] to update a subspace given new samples. We summarize Hall's method in [29] by using scatter matrixes to simplify the representation. Suppose there are two subspaces $\Omega_1 = (\mathbf{x}_1, U_1, \Lambda_1, N)$ and $\Omega_2 = (\mathbf{x}_2, U_2, \Lambda_2, M)$, which we are trying to merge to a new subspace $\Omega = (\bar{\mathbf{x}}, U, \Lambda, M + N)$. If the dimension of Ω_1 and Ω_2 are p and q, the dimension r of the merged subspace Ω satisfies: $\max(p, q) \leq r \leq p + q + 1$. The vector connecting the centers of the two subspaces does not necessarily belong to either subspace. This vector causes the additional one in the upper bound of r.

It is easy to verify that the scatter matrix S of the merged subspace Ω satisfies, $S = S_1 + S_2 + \frac{MN}{M+N}(\mathbf{x}_1 - \mathbf{x}_2)(\mathbf{x}_1 - \mathbf{x}_2)^T$. We aim to find a sufficient orthogonal spanning of S. Let $h_1(\mathbf{x})$ denote the residual vector of a vector $\mathbf{x}$ on Ω_1, $h_1(\mathbf{x}) = \mathbf{x} - U_1 U_1^T \mathbf{x}$. Note that $h_1(\mathbf{x})$ is orthogonal to U_1, *i.e.* $h(\mathbf{x})' U = 0$. Now, $U' = [U_1, \mathbf{v}]$ is a set of orthogonal bases to span the merged space, where $\mathbf{v} = GS\left(h_1\left(U_2, (\mathbf{x}_2 - \mathbf{x}_1)\right)\right)$ and $GS(\cdot)$ denote the Gram-Schmidt process.

Given the sufficient orthogonal bases, we can obtain the SVD decomposition of S.

$$U'^T S U' = \begin{bmatrix} \Lambda_1 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} G\Lambda_2 G^T & G\Lambda_2 \Gamma^T \\ \Gamma\Lambda_2 G^T & \Gamma\Lambda_2 \Gamma^T \end{bmatrix} + \frac{MN}{M+N} \begin{bmatrix} gg^T & g\gamma^T \\ \gamma g^T & \gamma\gamma^T \end{bmatrix} = R\Lambda R^T \quad (3)$$

where $G = U_1^T(\mathbf{x}_2 - \mathbf{x}_1)$, $\Gamma = \mathbf{v}^T U_2$, $g = U_1^T(\mathbf{x}_2 - \mathbf{x}_1)$ and $\gamma = U'(\mathbf{x}_2 - \mathbf{x}_1)$. Now, the eigenvalue of the merged subspace is Λ in Eq.3 and the eigenvector U is simply $U'R$. Note that incrementally updating a subspace with one observation as in [6] is one special case of merging two subspaces using Eq.3.

3.2 Subspace Distance

The other critical step in Algorithm 1 is to determine the similarity between two subspaces. We use two factors to measure the similarity between two neighboring subspaces Ω_1, Ω_2, the canonical angles (principal angles) and the data-compactness.

Suppose the dimensions of two subspaces are $p, q, p \geq q$, then there are q canonical angles between the two subspaces. A numerical stable algorithm [30] computes the angles between all pairs of orthonormal vectors of the two subspaces as, $\cos\theta_k = \sigma_k(U_1^T U_2), k = 1, \cdots, q$, where $\sigma_k(\cdot)$ is the k^{th} sorted eigenvalue computed by SVD. The consistency of two neighboring subspaces can be represented as follows.

$$Sim_1(\Omega_1, \Omega_2) = \prod_{k=q-d_0+1}^{q} \sigma_k(U_1^T U_2) \quad k = 1, \cdots, q \tag{4}$$

As the dimensionality of subspaces is larger than d_0, the initial dimension, we select the d_0 largest principal angles, which approximately measure the angle between two local subspaces. In a 3D space, the largest canonical angle between two 2D subspaces is equivalent to the angle between the two planes. In this case, we prefer to merge 2D patches with a small plane-to-plane angle. Note that the merge only happens between neighbor subspaces. The neighborhood is defined according to the mean vector L_2 distance. Merging subspaces with a small principal angle can avoid destroying the local structure of the appearance manifold.

The other factor to consider is data-compactness, which measures how much extra dimensionality is incurred by a merge operation. Suppose the dimension of two subspaces Ω_1, Ω_2 is $p, q, p \geq q$, the sorted eigenavalues of original merged subspace are $\Lambda_r = (\lambda_1, \ldots, \lambda_r), r = p + q + 1$. The similarity based on data-compactness is defined as

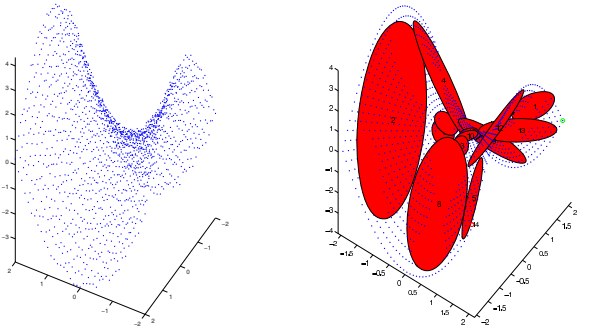
$$Sim_2(\Omega_1, \Omega_2) = \sum\nolimits_{i=1}^{p} \lambda_i / \sum\nolimits_{i=1}^{r} \lambda_i \tag{5}$$

If Sim_2 is close to one, this indicates the merge operation does not incur any new dimension; on the contrary, if Sim_2 is small, this indicates the variations in Ω_1 and Ω_2 cannot use common eigenvectors to represent it. Combining the two factors in Eq.4 and Eq.5, the final similarity between two subspaces is defined in Eq.6.

$$Sim(\Omega_1, \Omega_2) = Sim_1(\Omega_1, \Omega_2) + w_d Sim_2(\Omega_1, \Omega_2) \tag{6}$$

where w_d is the weight to balance these two factors. We use $w_d = 0.2$ in experiments.

One example of online trained local subspaces is shown in Figure 2. A saddle-like 3D surface (shown in Figure 2(a) is generated with Gaussian noise. The 3D points are input to Algorithm 1 sequentially. The final subspaces are shown in Figure 2(b) with $L = 15, \eta = 0.995$. The initial dimension d_0 is one. Although the online built subspaces depend on the order of the samples, a compact representation of the samples can always be created as long as the data are input with local smoothness. The subspace updating operation dominates the complexity of the generative tracker. Merging two subspaces with dimension p, q requires Golub-Reinsch SVD $O(r^3)$ of $r = p + q + 1$ dimension square matrix. Since the dimension of each subspace is low, the total complexity is quite low. The low dimensionality of the local subspaces can guarantee both the local property and efficient computation.



(a) 3D saddle surface (b) final subspaces built

Fig. 2. 3D example of incremental updating subspaces

4 Discriminative Tracker Using Online SVM

It has often been argued that SVM has better generalization performance than other discriminative methods on a small training set. For the discriminative model, we adopt an incremental SVM algorithm, *LASVM* [24], to train a classifier between object and background. SVM [31] is able to form the optimal separating function, which reduces to a linear combination of kernels on the training data, $f(\mathbf{x}) = \sum_j \alpha_j y_j K(\mathbf{x}_j, \mathbf{x}) + b$, with training samples $\mathbf{x_i}$ and corresponding label $y_i = \pm 1$.

In practice, this is achieved by maximizing the dual objective function $\max_\alpha W(\alpha)$ with $W(\alpha) = \sum_i \alpha_i y_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j K(x_i, x_j)$, subject to

$$\sum\nolimits_i \alpha_i = 0, A_i \leq \alpha_i \leq B_i, \tag{7}$$

where $A_i = \min(0, Cy_i), B_i \max(0, Cy_i)$. Here, α is a vector of weights on y_i. A SVM solver can be regarded as updating α along some direction to maximize $W(\alpha)$. Let $g = (g_1, \ldots, g_n)$ denote the gradient of $W(\alpha)$

$$g_k = \frac{\partial W(\alpha)}{\partial \alpha_k} = y_k - \sum_i \alpha_i K(x_i, x_k) = y_k - \hat{y}(x_k) + b \tag{8}$$

LASVM suggests that optimization is faster when the search direction mostly contains zero coefficients. *LASVM* uses the search directions whose coefficients are all zero except for a single +1 and a single -1. The two non-zero coefficients, are called *τ-violating pair* (i, j) if $\alpha_i < B_i, \alpha_j > A_j$, and $g_i - g_j > \tau$, where τ is a small positive value. and *LASVM* selects the τ-violating pair (i, j) that maximizes the directional gradient $g_i - g_j$.

The LASVM algorithm contains two procedures named PROCESS and REPROCESS [24]. When a new sample x_k arrives, PROCESS forms a *τ-violating pair* (i, j), which contains x_k and another existing support vector, and updates the weights of this pair. Following PROCESS, REPROCESS selects a *τ-violating pair* from the set of support

vectors and updates their weights. The new sample x_k may become a new support vector through PROCESS, while another support vector may need to switch out by REPROCESS. Both PROCESS and REPROCESS select τ-*violating pair* with the largest gradient. The complexity of such a selection grows linearly with the number of vectors. A finishing step, which runs REPROCESS multiple times to further remove as many τ-*violating pairs* as possible, is performed after online process. For tracking, the intermediate classifier is useful, hence we run this finishing step every 10 frames. Note that, since we do not need to look at the ignored vectors for incremental learning, τ-*violating pair* is only selected from the set of support vectors. For online tracking problem, many appearance variations and limited training samples may degrade the generalization ability of SVM. Also, in experiments, we find that the number of support vectors grows fast when the appearance of object and background changes. Thus, we propose to decrementally train the SVM and focus on recent appearance variations within a sliding window. REPROCESS in LASVM can be used to achieve the "unlearning" of old samples. For decremental learning, removing ignored vectors (when ignored vectors move out of the sliding window) will not change the decision boundary. However, the removal of a support vector will affect the decision boundary and some ignored vectors may become support vectors. In order to remove one support vector, we first zero its coefficient and put its coefficient into the closest vector to keep the constraint in Eq.7. We then apply REPROCESS multiple times to select τ-violating pairs in set of both ignored and support vectors and update the weights. The cost of decremental learning is that we need to store all samples within a sliding window.

5 Experiments

For both generative and discriminative models, we use image vectors of size 32×32 (for face) or 32×64 (for human and vehicle). For the generative model, η is set to 0.95-0.99 and the maximum number of subspaces is set to 5-10. The initial subspace dimension is 4, which is very low compared to the input space. Thus, every 5 frames, we form a new subspace, which is then inserted into the subspace pool. For the discriminative model, we use *LASVM* with R-HOG feature vectors, which are created from 16×16 blocks containing 8×8 cells. The strike size is 4 to allow overlapping HOG descriptors. Each cell has 9 bins oriented histogram; hence, we have 36-bin oriented histogram for a block. For a 32×64 window, the vector size is 2340. We use the linear kernel function in SVM. The number of support vectors varies between 50-150 for different sequences. We use a sliding window of 30 frames. We manually label the first 10 frames as the initialization for the two trackers. The Bayesian inference framework generates 600 particles. The co-trained hybrid model is implemented in C++. The combined tracker runs at around 2 fps on a P4 2.8GHz dual core PC. All testing sequences are 320×240 graylevel images.

During co-training, each learner labels the unlabeled data on which it makes a confident prediction based on its own knowledge. For this purpose, a threshold is needed for each learner. For the generative model, we set a threshold based on the log likelihood in Eq.2. To be more conservative, we use a second criteria: we find several local optima in the posterior distribution and if ratio ρ between the second optimum and the global

optimum is small enough ($\rho \leq 0.7$), we accept the global optimum as a positive sample and all other samples that far enough from the global optimum are negative samples. For the discriminative model, due to the very limited training data, the positive and negative training data are usually well separated. Thus, we cannot adopt the way used in [21] to select the threshold. Instead, we select the confidence threshold so that at most 80% positive samples' confidence is above that threshold. This threshold is updated every 30 frames. The positive and negative samples labeled by the generative model will not be added to the discriminative model unless they are close to the decision boundary. To express the SVM confidence as a probability, we use the method in [32] to fit a sigmoid function that is updated every 30 frames.

5.1 Comparative Analysis

We compare our co-trained tracker with two generative methods, including (G1) IVT [5] and our multiple linear subspaces (G2) algorithm and three discriminative methods, including online selection of discriminative color (D1) [10], our online SVM method (D2) and ensemble tracking (E.T) [9]. G1 uses a single 15D linear subspace and updates it incrementally. Note that D1 does not consider tracking with large scale change and rotation. G1, G2, D2 and the co-trained tracker use the same parameters in CONDENSATION algorithm, but G1, G2 and D2 use self-learning to update their models.

We compare these methods with challenging data sets, which contain image sequences of various types of object, including face (seq1-seq2), human (seq3-seq5) and vehicle (seq6). The challenging conditions include significant illumination changes (seq1), abrupt camera motion and significant motion blur (seq2-seq5), viewpoint changes and/or pose variations (seq3-seq6), and also occlusions (seq4-seq6). To compare the robustness under the challenging conditions, we show how many frames these methods can track the objects before tracking failure, *i.e.* after this frame a tracker cannot recover without re-initialization. Table 1 shows the comparison between different methods. The number of frames and the number of frames where occlusion happens in each sequence are also shown in Table 1. The comparison demonstrates that the co-trained tracker performs more robustly than other methods. Note that D1 requires color information, thus it cannot process some sequences, which are indicated with "n/a". The visual results are shown in Figure 4, where the tracked objects and part of negative

Table 1. Comparison of different methods G1:IVT [5], G2: incremental learning multiple subspaces, D1: online selection of discriminative color features [10], D2: online SVM, E.T: ensemble tracking [9]. D1 uses color information, which is not available for Seq1 and Seq6.

	Frames	Occlusion	G1	G2	D1*	D2	E.T	Ours
Seq1	761	0	17	261	n/a	491	94	**759**
Seq2	313	0	75	282	313	214	44	**313**
Seq3	140	0	11	15	6	89	22	**140**
Seq4	338	93	33	70	8	72	118	**240**
Seq5	184	30	50	50	50	50	53	**154**
Seq6	945	143	163	506	n/a	54	10	**802**

(a) the proposed method

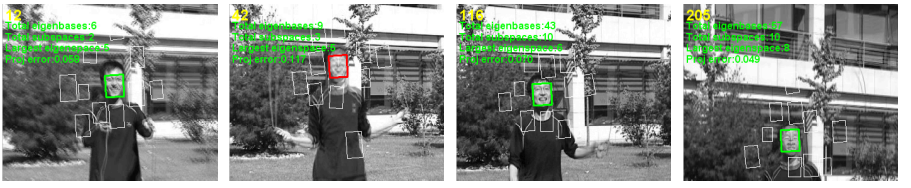
(b) ensemble tracking

Fig. 3. Tracking and reacquisition with long occlusion and clutter background

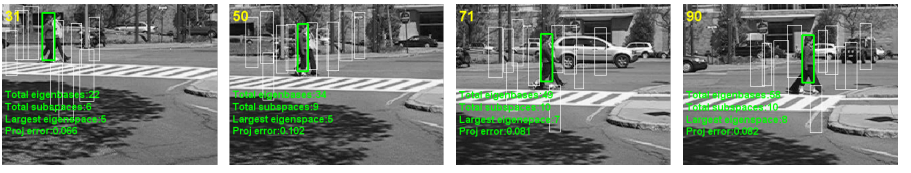
samples are bounded with green boxes and white boxes respectively. The red box indicates none of the models is updated in this frame. In experiments, we frequently find that the co-trained tracker has better self-awareness of current tracking performance and can safely enlarge the search range (by changing the diffusion dynamics) without being confused by distracters in the background. Also, the co-trained tracker successfully avoids drifting caused by varying viewpoints and illumination changes.

We compare with two other methods to demonstrate our method's reacquisition ability. One is ensemble tracking [9]. The other one is the online SVM tracker without decremental learning. In the experiments, we find that ensemble tracking can only reacquire the object after a short occlusion. Also, ensemble tracking cannot deal with rotation and scale change, which restricts its reacquisition ability. Without decremental learning, the number of support vectors increases to several hundreds quickly when the background becomes cluttered and the object appearance changes. This makes the online SVM tracker very slow. Meanwhile, the performance of the tracker goes down, exhibiting as drifting and being trapped by distracters.

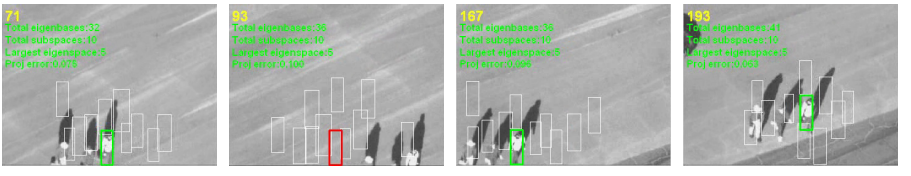
We also compare our generative tracker G2 with G1 and another generative method G3, in indoor environments with few distracters. G3 uses 5 key 10D subspaces (corresponding to front, left profile, right profile, up and down), which are trained offline by manually labelling 150 samples into 5 categories; our generative method G2 uses at most 10 subspaces and each new subspace starts from 4-dimensions. The indoor sequence exhibits significant head pose variations, shown in Figure 5(b). We calculate projection errors of each method in Figure 5(a) (average of multiple runs) within 1000 frames before the other two trackers start to drift. Offline 5-key subspaces method shows large projection errors at some frames where the poses are not covered in offline samples. G1 is not able to adapt rapidly to new appearance variations after running a long sequence. Our generative method G2 can promptly adapt to appearance variations and show smaller projection errors consistently, though each subspace in our generative track has much smaller dimensionality than the other two methods. As we can see, the



(a) Tracking and reacquisition with abrupt motion and blur

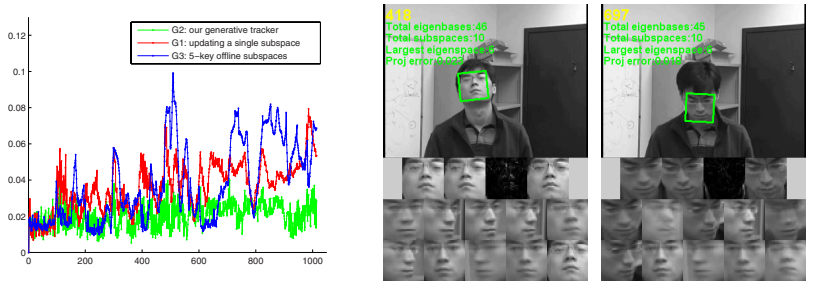


(b) Tracking human with clutter background and distracters



(c) Tracking and reacquisition with long leaving out of field of view

Fig. 4. Tracking various type of objects in outdoor environments



(a) Comparison of projection errors (b) Tracking with pose variations

Fig. 5. Comparison of generative methods in indoor environments

online trained subspaces approximately represent the poses that have been observed, though we do not train offline in different poses.

6 Summary and Future Work

We have proposed a co-training framework to combine one global generative tracker and one local discriminate tracker. Our generative tracker builds a compact representation of the complete appearance of an object by online learning a number of local linear

subspaces. The discriminative tracker adopts the online SVM algorithm to focus on local appearance. By co-training, the two trackers can train each other on-the-fly with limited initialization. Though no offline training data is used, our method has strong reacquisition ability and robustness against the distracters in background. Extensive experiments demonstrate that our method can handle different challenging situations.

Our method relies on smoothness in appearance changes and cannot deal with abrupt appearance changes. Also, our method cannot explicitly handle the partial occlusion problem. Partial occlusions are often regarded as non-object by our method. This is a safe strategy to avoid updating the model with wrong appearance instances. It is worth noting that our method addresses online building appearance model for unknown types of objects. In the future, we expect to combine this method with offline learning for tracking one particular type of object. For tracking a particular type of objects, the robustness of tracking can be significantly improved by combining online training with offline training, as shown in [33] and [34]. We also expect to introduce collaborative part-based trackers under this co-training framework to deal with partial occlusions.

Acknowledgements. This research was funded, in part, by MURI-ARO W911NF-06-1-0094. The second author is also supported by Vietnam Education Foundation. We thank Dr. David Ross, Dr. Shai Avidan and Dr. Antoine Bordes for sharing the data and implementations of methods in [5], [9] and [24] respectively. We also thank Yuan Li for providing part of the data.

References

1. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. ACM Comput. Surv. (2006)
2. Black, M.J., Jepson, A.D.: Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. IJCV 26, 63–84 (1998)
3. Jepson, A.D., Fleet, D.J., El-Maraghi, T.F.: Robust online appearance models for visual tracking. In: CVPR (2001)
4. Isard, M., Blake, A.: Contour tracking by stochastic propagation of conditional density. In: ECCV, pp. 343–356 (1996)
5. Lim, J., Ross, D., Lin, R., Yang, M.: Incremental learning for visual tracking. In: NIPS, pp. 793–800 (2004)
6. Lee, K.C., Kriegman, D.: Online learning of probabilistic appearance manifolds for video-based recognition and tracking. In: CVPR, pp. 852–859 (2005)
7. Elgammal, A.: Learning to track: Conceptual manifold map for closed-form tracking. In: CVPR, pp. 724–730 (2005)
8. Avidan, S.: Support vector tracking. PAMI (2004)
9. Avidan, S.: Ensemble tracking. In: CVPR, vol. 2, pp. 494–501 (2005)
10. Collins, R.T., Liu, Y., Leordeanu, M.: Online selection of discriminative tracking features. In: PAMI, vol. 27, pp. 1631–1643 (2005)
11. Nguyen, H.T., Smeulders, A.W.: Robust tracking using foreground-background texture discrimination. IJCV 2006 (2006)
12. Oza, N.C., Russell, S.: Online bagging and boosting. In: International Workshop on Artificial Intelligence and Statistics (2001)
13. Grabner, M., Grabner, H., Bischof, H.: Learning features for tracking. In: CVPR (2007)
14. Liu, X., Yu, T.: Gradient feature selection for online boosting. In: ICCV (2007)

15. Lasserre, J.A., Bishop, C.M., Minka, T.P.: Principled hybrids of generative and discriminative models. In: CVPR (2006)
16. Ng, A.Y., Jordan, M.I.: On discriminative vs. generative classifiers: a comparison of logistic regression and naive bayes. In: NIPS (2001)
17. Raina, R., Shen, Y., Ng, A.Y., McCallum, A.: Classification with hybrid generative/discriminative models. In: NIPS (2003)
18. Lin, R.S., Ross, D., Lim, J., Yang, M.H.: Adaptive discriminative generative model and its applications. In: NIPS (2004)
19. Yang, M., Wu, Y.: Tracking non-stationary appearances and dynamic feature selection. In: CVPR (2005)
20. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: COLT (1998)
21. Levin, A., Viola, P., Freund, Y.: Unsupervised improvement of visual detectors using co-training. In: ICCV (2003)
22. Javed, O., Ali, S., Shah, M.: Online detection and classification of moving objects using progressively improving detectors. In: CVPR (2005)
23. Tang, F., Brennan, S., Zhao, Q., Tao, H.: Co-tracking using semi-supervised support vector machines. In: ICCV (2007)
24. Bordes, A., Ertekin, S., Weston, J., Bottou, L.: Fast kernel classifiers with online and active learning. In: JMLR 2005, pp. 1579–1619 (2005)
25. Cauwenberghs, G., Poggio, T.: Incremental and decremental support vector machine learning. In: NIPS (2001)
26. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR, pp. 886–893 (2005)
27. Isard, M., Blake, A.: Condensation - conditional density propagation for visual tracking. IJCV, 5–28 (1998)
28. Moghaddam, B., Pentland, A.: Probabilistic visual learning for object representation. PAMI 19, 696–710 (1997)
29. Hall, P., Marshall, D., Martin, R.: Merging and splitting eigenspace models. In: IEEE PAMI, pp. 1042–1049 (2000)
30. Bjoerck, A., Golub, G.H.: Numerical methods for computing angles between linear subspaces. Mathematics of computation, 579–594 (1973)
31. Vapnik, V.N.: Statistical Learning Theory. John Wiley Sons, Chichester (1998)
32. Platt, J.C.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In: Advances in Large Margin Classifiers (1999)
33. Li, Y., Ai, H., Yamashita, T., Lao, S., Kawade, M.: Tracking in Low Frame Rate Video: A Cascade Particle Filter with Discriminative Observers of Different Lifespans. In: IEEE CVPR (2007)
34. Kim, M., Kumar, S., Pavlovic, V., Rowley, H.: Face Tracking and Recognition with Visual Constraints in Real-World Videos. In: IEEE CVPR (2008)