Tracking Non-stationary Appearances and Dynamic Feature Selection

Ming Yang Ying Wu
ECE Department, Northwestern University
2145 Sheridan Road, Evanston, IL 60208
{mya671,yingwu}@ece.northwestern.edu

Abstract

Since the appearance changes of the target jeopardize visual measurements and often lead to tracking failure in practice, trackers need to be adaptive to non-stationary appearances or to dynamically select features to track. However, this idea is threatened by the risk of adaptation drift that roots in its ill-posed nature, unless good constraints are imposed. Different from most existing adaptation schemes, we enforce three novel constraints for the optimal adaptation: (1) negative data, (2) bottom-up pair-wise data constraints, and (3) adaptation dynamics. Substantializing the general adaptation problem as a subspace adaptation problem, this paper gives a closed-form solution as well as a practical iterative algorithm. Extensive experiments have shown that the proposed approach can largely alleviate adaptation drift and achieve better tracking results.

1 Introduction

Tracking targets in video is a fundamental research problem in video analysis and is important for a large variety applications including video surveillance and vision-based interfaces. Recent years have witnessed a steady advance in both theory and practice, e.g., the sampling-based methods [8, 14] and the kernel-based methods [4, 6]. Although many tracking tasks can be successfully handled by these techniques, the real situations in practice, e.g., long duration tracking and unconstrained environments, still pose enormous challenges to these techniques. One common difficulty arisen from these real situations is the non-stationary visual characteristics of the target due to the view or illumination changes, leading to the changes in target appearances over time. Such appearance changes can jeopardize the visual measurements and lead to tracking failure.

Two general approaches can be taken to overcome this challenge. One of them is to use the visual invariants of the targets. But in general finding invariants itself is very difficult, if not impossible, although learning methods can be employed [1, 2, 5, 12]. Another approach is to adapt the tracker to the changes, for example, by updating the appearance models [9, 7, 13, 14], or selecting best visual features [3]. Unlike the invariants-based methods that require off-line learning of the visual measurement models (the appearance models), the adaptation-based methods trend to be more flexible, since the measurement models are adaptive or the features used for tracking can be adaptively selected.

In the practice of most existing adaptation-based methods, it is not uncommon to observe the adaptation drift, i.e., the appearance models adapt to other image regions than the target of interest and lead to tracking failure. Many *ad hoc* remedies have been proposed to alleviate the drift, e.g., by enforcing the similarity to the initial model [3, 7] which confines the range of possible adaptations. However, in general, such a phenomenon of adaptation drift is not accidental but widely exists in a large variety of adaptation schemes, and poses a threat to adaptation-based visual trackers. Thus, it worths an in-depth investigation to facilitate careful designs of the adaptation schemes.

In most existing adaptive tracking methods, the model at the current time instant is updated by the new data that are closest to the model at previous time step, with a hidden assumption that the best model (or feature) up to time $t-1$ is also the best for time t. Unfortunately, this assumption may not universally hold. As a result, when this assumption becomes invalid, the data closest to the model at time $t-1$ may actually be far from the right model at time t, and thus deviating the adaptation and failing the tracker.

The nature of the adaptive tracking problem lies in a chicken-and-egg dilemma: the right data at time t are found by the right model at time t, while the right model can only be adapted by using the right data at time t. If no constraints are enforced, any new data can lead to a valid and stable adaptation, since the adapted model trends to best fit the new data. Therefore, in order to make this problem well-posed, good constraints need to be introduced, and they should be reasonable and allow a wide range of adaptation.

In this paper, we substantialize the general adaptation problem as a subspace adaptation problem in non-stationary appearance tracking, where the target visual appearances at

a short time interval are represented as a linear subspace. We analyze the ill-posed adaptive tracking problem in this setting. In our approach, we enforce three novel constraints for the optimal adaptation: (1) negative data that are easily available, (2) pair-wise data constraints that are used to identify positive data from bottom-up, and (3) adaptation dynamics that smooth the updating process. In this paper, we give a closed-form solution to this subspace tracking problem and also provide a practical iterative algorithm. Our method not only estimates the motion parameters of the target, but also keeps track the appearance subspaces.

2 Related Work

Visual appearance is critical for tracking, since the target is tracked or detected based on the matching between the observed visual evidence (or measurements) and the visual appearance model. We generally call the model that stipulates the matching as the *observation model* or the *measurement model*. The visual appearances of an object shall bear a manifold of the image space. Depending on the features used to describe the target and on the variances of the appearances, such a manifold can be quite nonlinear and complex. Therefore, the complexity in the appearances largely determines the degrees of difficulty of the tracking task.

We can roughly categorize the observation models in various tracking algorithms into three classes: (1) with fixed appearance templates, (2) with known appearance manifolds; (3) with adaptive appearance manifolds on-the-fly.

In the observation models with fixed appearance templates, the motion parameters to be estimated (denoted by $\mathbf{x}$) are the only variables that affect the appearance changes. We denote the image observations as $\mathbf{z}$ and the hypothesized one as $\hat{\mathbf{z}}(\mathbf{x})$. Then the observation model needs to measure the similarity of $\mathbf{z}$ and $\hat{\mathbf{z}}(\mathbf{x})$, or the likelihood $p(\mathbf{z}|\mathbf{x}) = p(\mathbf{z}|\hat{\mathbf{z}}(\mathbf{x}))$. If $\mathbf{z}$ is a vector, i.e., $\mathbf{z} \in \mathbb{R}^m$, this class of observation models is concerned with the distance between two vectors. The image observations $\mathbf{z}$ can be edges [8], color histograms [4], etc. Most tracking algorithms employ this type of observation models.

There are cases where the motion parameters of interest are not the only contribution to the appearance changes, but there can be many other factors. We denote it by $\hat{\mathbf{z}}(\mathbf{x}, \theta)$. For example, the illumination also affects the appearance [5] (e.g., in tracking a face), or the non-rigidity of the target changes the appearances (e.g., in tracking a walking person), but we may not be interested in recovering too many delicate non-rigid motion parameters. Thus, there are uncertainties in the appearances model itself, and the observation model needs to integrate all these uncertainties, i.e.,

$$p(\mathbf{z}|\mathbf{x}) = \int_\theta p(\mathbf{z}|\mathbf{x}, \theta)p(\theta|\mathbf{x})d\theta = \int_\theta p(\mathbf{z}|\hat{\mathbf{z}}(\mathbf{x}, \theta))p(\theta|\mathbf{x})d\theta.$$

In other words, given a motion hypothesis $\mathbf{x}$, its hypothesized observation $\hat{\mathbf{z}}(\mathbf{x})$ is no longer a vector, but a manifold in $\mathbb{R}^m$, and the observation model needs to calculate the distance of the evidence $\mathbf{z}$ to this manifold. Depending on the free parameters θ, such a manifold can be as simple as a linear subspace [2, 5], or as complex as a highly non-linear one [1, 12]. The second class of observation models assumes a known manifold, which can be learned from training data off-line in advance.

Although the appearance manifolds exist, in most cases, they are quite complex, and the learning task itself is challenging enough. In addition, in real applications, we may not have the luxury of being able to learn the manifolds of arbitrary objects for two reasons: we may not be able to collect enough training data, and the applications may not allow the off-line processing. Thus, we need to recover and update the appearance manifolds online during the tracking [9, 13, 14]. In general, we make a reasonable assumption that the manifold at a short time interval is linear [7, 10]. The method of online feature selection, e.g., in [3], can also be categorized in this class, since the selected features span a subspace. In these methods, model drift is one of the common and fundamental challenges.

This paper studies the problem of adaptive appearance models. The differences from the existing work include an in-depth analysis of the adaptation drift, and a novel solution that alleviates the drift by enforcing both bottom-up and top-down constraints.

3 The Nature of the Problem

Although the appearance manifold of a target can be quite complex and nonlinear, it is reasonable to assume the linearity over a short time interval. In this paper, we assume the appearances (or visual features) $\mathbf{z} \in \mathbb{R}^m$ lie in a linear subspace $\mathcal{L}$ spanned by r linearly independent columns of a linear transform $\mathbf{A} \in \mathbb{R}^{m \times r}$, i.e., $\mathbf{z}$ is a linear combination of the columns of $\mathbf{A}$. We write $\mathbf{z} = \mathbf{A}\mathbf{y}$.

The projection of $\mathbf{z}$ to the subspace $\mathbb{R}^r$ is given by the least square solution of $\mathbf{z} = \mathbf{A}\mathbf{y}$, i.e.,

$$\mathbf{y} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{z} = \mathbf{A}^\dagger\mathbf{z},$$

where $\mathbf{A}^\dagger = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$ is the pseudo-inverse of $\mathbf{A}$. The reconstruction of the projection in $\mathbb{R}^m$ is given by:

$$\bar{\mathbf{z}} = \mathbf{A}\mathbf{A}^\dagger\mathbf{z} = \mathbf{P}\mathbf{z},$$

where $\mathbf{P} = \mathbf{A}\mathbf{A}^\dagger \in \mathbb{R}^{m \times m}$ is called the *projection matrix*. Unlike the orthonormal basis, the projection matrix is unique for a subspace. We can decompose the Hilbert space $\mathbb{R}^m$ into two orthogonal subspaces: a r-dimensional subspace characterized by $\mathbf{P}$ and its $(m - r)$-dimensional orthogonal complement characterized by $\mathbf{P}^\perp = \mathbf{I} - \mathbf{P}$.

Therefore, the subspace $\mathcal{L}$ delineated by a random vector process $\{\mathbf{z}\}$ is given by the following optimization problem:

$$\mathbf{P}^* = \arg\min_P \mathtt{E}(||\mathbf{z} - \mathbf{P}\mathbf{z}||^2) = \arg\min_P \mathtt{E}(||\mathbf{P}^\perp\mathbf{z}||^2).$$

It is easy to prove that the optimal subspace is spanned by the r principal components of the data covariance matrix. This problem is well-posed since the samples from $\{\mathbf{z}\}$ are given, thus the covariance matrix is known.

However, in the tracking scenario, the problem becomes:

$$\{\mathbf{P}_t^*, \mathbf{x}_t^*\} = \arg\min_{P_t, x_t} \mathtt{E}(||\mathbf{P}_t^{\perp}\mathbf{z}(\mathbf{x}_t)||^2), \quad (1)$$

where $\mathbf{x}_t$ is the motion parameters to be tracked. In this setting, we are facing a dilemma: if $\{\mathbf{x}\}$ can not be determined, then neither can $\mathbf{P}$, and vice versa. Namely, given any tracking result, good or bad, we can always find an optimal subspace that can best explain this particular result.

Unlike some other chicken-and-egg problems, this problem is even worse since no constraints on either $\mathbf{P}$ or $\{\mathbf{x}\}$ are imposed. Therefore, this problem is ill-posed and the formulation allows arbitrary subspace adaptations.

From the analysis above, it is clear that constraints need to be added to make this problem well-posed. A commonly used constraint is the "smoothness" of the adaptation, i.e., the updated model should not deviate much from the previous one, and most existing methods [3, 7, 9, 10] solve this dilemma in the following manner:

$$\begin{cases} \mathbf{x}_t^* &= \arg\min_{x_t} \mathtt{E}(||\mathbf{P}_{t-1}^{\perp}\mathbf{z}(\mathbf{x}_t)||^2) \\ \mathbf{P}_t^* &= \arg\min_{P_t} \mathtt{E}(||\mathbf{P}_t^{\perp}\mathbf{z}(\mathbf{x}_t^*)||^2). \end{cases}$$

In this adaptation scheme, at time t, the data that are the closest to the subspace at the previous time instant are found first, and then are used to update the subspace. This approach is valid only if the following assumption holds: the optimal subspace at $t-1$ is also optimal for time t. In reality, this assumption may not necessarily be true, since a data point that is the closest to the subspace $\mathcal{L}_t$ may not be the closest to $\mathcal{L}_{t-1}$. Thus, we often observe that the model adaptation can not keep up with the real changes and the model gradually drifts away. When the data found based on $\mathbf{P}_{t-1}$ in fact deviate from $\mathbf{P}_t$ significantly, the adaptation is catastrophic. Although this approach makes the original ill-posed problem in Eq. 1 well-posed, it is prone to drift and thus not robust.

4 Our Solution

From the analysis in Section 3, it is clear that we need more constraints than the adaptation dynamics constraint alone. In the tracking problem, at time t before the target is detected, all the observation data are unlabelled data, i.e., we can not tell if or not a certain observation should be associated (or classified) to the target appearance subspace. The adaptation dynamics constraint is a top-down constraint, which does not provide much supervised information to the data at time t. Therefore, to make the adaptation more robust, we need to also identify and employ bottom-up data-driven constraints, beside the smoothness constraint.

In this paper, we propose to integrate the following three constraints:

- *Adaptation smoothness constraints.* The smoothness constraints are essential for the tracking process, since the process of the data at time t should take advantage of the subspace at time $t-1$. There are many ways to represent and use this type of constraints. The most common scheme as indicated in Section 3 enforces a very strong smoothness constraint. In our approach, we treat the constraint as a penalty which can be balanced with other types of constraints. The penalty is proportional to the distance of two subspaces, i.e., the Frobenius norm of the difference of the two projection matrices $||\mathbf{P}_t - \mathbf{P}_{t-1}||_F^2$;
- *Negative data constraints.* At the current time t, although it is difficult to obtain the positive data (i.e., the visual observations that are truly produced by the target), negative data are everywhere. In fact, positive data are very rare in all the set of possible observation data. The negative data may help to constrain the target appearance subspaces. We denote the positive data at time t by $\mathbf{z}_t^+$, and the negative data by $\mathbf{z}_t^-$;
- *Pair-wise data constraints.* Given a pair of data points, it is relatively easier to determine if or not they belong to the same class. Such pair-wise data constraints are also widely available. A large number pair-wise constraints may lead to a rough clustering of the data. Based on the smoothness constraints, we can determine a set of *possible positive* data to constrain the subspace updating. The detailed is in Section 4.4.

4.1 Formulation

When processing the current frame t, the following are assumed to be known: (1) the projection matrix of the previous appearance subspace $\mathbf{P}_{t-1}$, (2) a set of negative data collected from the current image frame, $\{\mathbf{z}_t^-\}$, (3) a set of possible positive data identified based on the pair-wise constraints, $\{\mathbf{z}_t^+\}$, (4) previous negative covariance matrix $\mathbf{C}_{t-1}^-$ and positive covariance matrix $\mathbf{C}_{t-1}^+$.

An optimal subspace should have the following properties. The negative data should be far from their projections onto this subspace; the positive data should be close to their projections, and this subspace should be close to the previous one. Therefore, we form an optimization problem to solve for the optimal subspace at current time t:

$$\min_{A_t} J_0(\mathbf{A}_t) = \min_{A_t}\{\mathtt{E}(||\mathbf{z}_t^+ - \mathbf{P}_t\mathbf{z}_t^+||^2) + \mathtt{E}(||\mathbf{P}_t\mathbf{z}_t^-||^2) + \alpha||\mathbf{P}_t - \mathbf{P}_{t-1}||_F^2\}, \quad (2)$$

where $\mathbf{P}_t = \mathbf{A}_t\mathbf{A}_t^{\dagger}$ is the projection matrix and $\alpha > 0$ is a weighting factor. We denote by $\mathbf{C}_t^+ = \mathtt{E}(\mathbf{z}_t^+\mathbf{z}_t^{+T})$, and

$\mathbf{C}_t^- = \mathtt{E}(\mathbf{z}_t^- \mathbf{z}_t^{-T})$. It is easy to show Eq. 2 is equivalent to the following:

$$\min_{A_t} J_1(\mathbf{A}_t) = \min_{A_t}\{\mathtt{tr}(\mathbf{P}_t\mathbf{C}_t^-) - \mathtt{tr}(\mathbf{P}_t\mathbf{C}_t^+) + \alpha||\mathbf{P}_t - \mathbf{P}_{t-1}||_F^2\}, \quad (3)$$

where $\mathtt{tr}(\cdot)$ denotes the trace of a matrix.

4.2 An Closed-form Solution

Theorem 1 *The solution to the problem in Eq. 3 is given by* $\mathbf{P}_t = \mathbf{U}\mathbf{U}^T$*, where* $\mathbf{U}$ *is constituted by the* r *eigenvectors that corresponds to the* r *smallest eigenvalues of a symmetric matrix*

$$\hat{\mathbf{C}} = \mathbf{C}_t^- - \mathbf{C}_t^+ + \alpha\mathbf{I} - \alpha\mathbf{P}_{t-1}.$$

The proof of this theorem is given in the Appendix. Please note that the solution to $\mathbf{A}_t$ is not unique, but the projection matrix $\mathbf{P}_t$ is. If we require $\mathbf{A}_t$ spanned by r orthogonal vectors, then $\mathbf{A}_t = \mathbf{U}$. Please also note the eigenvalues of $\hat{\mathbf{C}}$ may be negative.

By considering the data in previous time instants, we can use a forgetting factor $\beta < 1$, which can down-weight the influence of the data from previous times. This is equivalent to the use of exponentially-weighted sliding window over time. Thus, we can write:

$$\mathbf{C}_t = \sum_{k=1}^{t} \beta^{t-k}\mathtt{E}(\mathbf{z}_k\mathbf{z}_k^T) = \beta\mathbf{C}_{t-1} + \mathtt{E}(\mathbf{z}_t\mathbf{z}_t^T).$$

This way, we can update both $\mathbf{C}_t^+$ and $\mathbf{C}_t^-$.

4.3 An Iterative Algorithm

Section 4.2 gives a closed-form solution to the subspace, but this solution involves the eigenvalue decomposition of a $m \times m$ matrix $\hat{\mathbf{C}}$, where m is the dimension of the visual observation vectors and thus can be quite large. To achieve a less demanding computation, we develop an iterative algorithm in this section, by formulating another optimization problem as:

$$\begin{aligned}\min_U J_2(\mathbf{U}) = \min_U\{&\mathtt{E}(||\mathbf{z}_t^+ - \mathbf{U}\mathbf{U}^T\mathbf{z}_t^+||^2) \\ &+\mathtt{E}(||\mathbf{U}\mathbf{U}^T\mathbf{z}_t^-||^2) + \alpha||\mathbf{U}\mathbf{U}^T - \mathbf{P}_{t-1}||_F^2\} \\ s.t. \quad &\mathbf{U}^T\mathbf{U} = \mathbf{I}, \end{aligned} \quad (4)$$

where $\mathbf{U} \in \mathbb{R}^{m\times r}$ is constituted by r orthonormal columns. The gradient of J_2 is given by:

$$\nabla J_2(\mathbf{U}) = \frac{\partial J_2(\mathbf{U})}{\partial \mathbf{U}} \propto (\mathbf{C}_t^- - \mathbf{C}_t^+ + \alpha\mathbf{I} - \alpha\mathbf{P}_{t-1})\mathbf{U}. \quad (5)$$

To find the optimal solution of $\mathbf{U}$, we can use the gradient descent iterations:

$$\mathbf{U}^k \longleftarrow \mathbf{U}^{k-1} - \mu\nabla J_2(\mathbf{U}^{k-1}), \quad (6)$$

during which the columns of $\mathbf{U}^k$ need to be orthogonalized after each update.

To speed up the iteration, we can also perform an approximation. When the subspace is to be updated by the positive data $\mathbf{z}_t^+$, the PAST algorithm [15] can be adopted for fast updating. When the updating is directed by the negative data $\mathbf{z}_t^-$, we can use the gradient-based method in Eq. 5.

4.4 Pair-wise Constraints

Although the target can not be detected directly, the low level image features which distinguish the target object from its neighborhood may give some hints about the target. Here we employ a graph cut algorithm [11] to roughly cluster some sample appearances collected within the predicted target regions. Then we may be able to find possible positive data and negative data from bottom-up.

Suppose the predicated region for the target is a rectangle region centered at (u, v) with width w and height h. We draw uniform samples (i.e., 15×15 image patches) to cover a rectangle region $(u\pm w, v\pm h)$. For each sample patch, the kernel-weighted [4] hue histogram $\mathbf{h}$ with 64 bins is calculated. The affinity matrix, obtained based on the similarity of all pairs of these histograms, is:

$$\mathbf{S} = [S_{ij}], \quad \text{where } S_{ij} = \mathtt{exp}\left\{\frac{(\rho(\mathbf{h}_i, \mathbf{h}_j) - \mu)^2}{2\sigma^2}\right\}, \quad (7)$$

where $\rho(\cdot)$ is the Bhattacharya coefficient, μ is the mean of all coefficients, σ is their standard deviation. These sample patches can be grouped into $3-5$ clusters by the eigenvalue decomposition of the affinity matrix.

It is not necessary to have a perfect clustering, as observed in our experiments. The image region delineated by the cluster with the minimum mean L^2 distance to the previous target subspace indicates the possible locations that the target may present. In practice, we can simply treat its geometry centroid as the possible location of the target and the corresponding appearance vector as the possible positive data $\mathbf{z}_t^+$.

4.5 Selecting Negative Data

The negative data should be selected carefully. Because if the negative data are too far from the target, the data point may already lie in the orthogonal complement of the target subspace, then minimizing the projections of the negative data may not help. In addition, if the negative data are too close to the target, they may lie partly in the target subspace such that the estimated target subspace is pushed away from its true place. Our selection of negative data is heuristic based on the clustering in Section. 4.4: in the image regions spanned by all the negative clusters, we find the locations whose appearances (or features) are close to the previous target manifold, and treat these appearance data as negative data $\mathbf{z}_t^-$ in order to distinguish the target from the negative

clusters. This heuristics works well in our experiments, but a more principled selection deserves more investigation.

5 Experiments

5.1 Setup and Comparison Baseline

In our experiments, the motion $\mathbf{x} = \{u, v, s\}$, where (u, v) is the location of the target and s is its scale. The corresponding appearance region is normalized to a 20×20 window and rasterized to a feature vector $\mathbf{z} \in \mathbb{R}^{400}$. Since the target appearances during tracking may become totally different from the first frame, the remedy of always including the initial appearance in the model [3, 7] does not apply.

For comparison, we implemented a subspace updating tracker similar to the method in [7], where the nearest appearance observation $\mathbf{z}_i$ to the previous target subspace $\mathbf{P}_{t-1}$ is used to update the orthonormal basis of the subspace by using Gram-Schmidt and dropping the oldest basis. We refer to this method *Nearest Updating*. The method is referred as *Nearest+Negative* when the positive data are collected by the nearest scheme and the negative data are used in updating the same way as in our approach. In all these methods, the adaptation applies every 4 frames.

5.2 Impact of the Positive and Negative Data

In this quantitative study, we show that the use of negative and possible positive data do help. We have manually annotated a video with 300 frames, in which a head presents a 180^o out-of-plane rotation, and collect the ground truth appearance data for each frame (denoted by $\mathbf{z}_t^*$). The comparison is based on the L^2 distance of the ground truth data $\mathbf{z}_t^*$ to the subspaces estimated by various methods. A smaller distance implies a better method.

As shown in Figure 1(a), the distance curve for the *Nearest+Negative* scheme is slightly lower than that for *Nearest Updating*, showing negative data can help to keep the adaptation away from the wrong subspaces. We also observed in our experiments that the negative data themselves may not be able to precisely drive the adaptation to the right places.

We compare the proposed with *Nearest+Negative* in Figure 1(b), in which the curve of our approach is apparently lower than that of *Nearest Updating*. This verifies that the possible positive data from bottom-up do help.

These two comparisons validate that the proposed approach are more capable of following the changes of the non-stationary appearances. Some sample frames are shown in Figure 2, where the top row is the results of the proposed method, the middle row shows the location of the possible positive cluster and possible positive data is shown at the top-left corner of each frame, and the bottom row shows the results of *Nearest Updating* and the nearest data is shown at the top-left corner as well. The details can be viewed in the submitted video `head180.avi`.

5.3 Impact of the Clustering Procedure

In this experiment, we compare our method with *Nearest Updating* in the situation of partial occlusion. We need to track a face, but the partial occlusion makes it difficult when the person drinks and the face moves behind a computer.

Figure 3. Clustering performance in `face.avi`: top row shows the drift process of *Nearest Updating* around frame 272; middle row lists 6 positive data at frame 272; bottom row lists 6 positive data at frame 284.

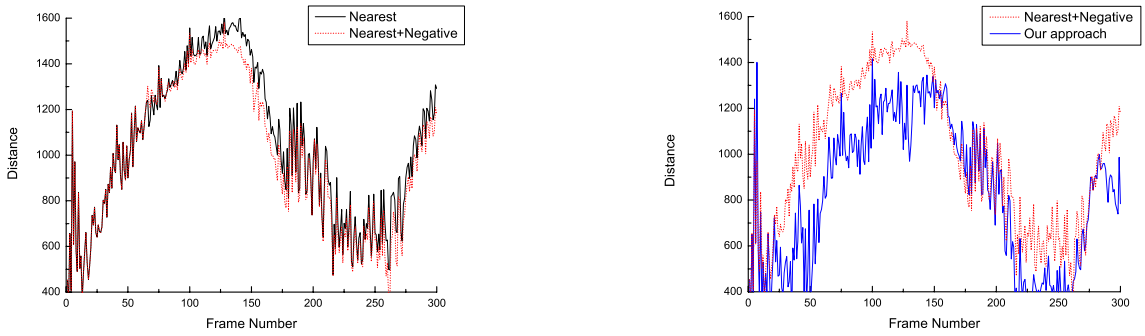
When the face moves slowly behind the computer, *Nearest Updating* drifts and erroneously adapts to a more stable appearance, i.e., a back portion of the computer. In Figure 3, the top row illustrates this drift process in detail.

The middle row in Figure 3 presents 6 appearance data from the possible positive cluster in our method at the 272-th frame. Obviously, some of them are not faces, since the clustering is quite rough. But our heuristic of selecting the centroid of the cluster does help and leads to a correct adaptation. Similarly, the bottom row shows the situation of our method at the 284-th frame. As the person moves upward, our method correctly follows the face.

This also shows that a rough clustering is sufficient for our method which is more robust than *Nearest Updating*. Some sample frames are shown in Figure 4, where the top row is our method and the bottom row is that of *Nearest Updating*. The details of this demo can be viewed in the video `face.avi`.

5.4 Tracking a Head with 360^o Rotations

Figure 5 shows the results of tracking a head presenting 360^o out-of-plane rotation (The demo is in `head360.avi`). The appearances of different views of the head are significantly different, which makes the tracking difficult and also challenges the adaptation. Our experiment shows that *Nearest Updating* tends to stick to the past appearances and thus reducing the likelihood of including new appearances. For example, when the front face gradually disappears, this scheme is unable to adapt to the hairs to track the back head. In all of our experiments, this scheme loses track when the head fades away. On the contrary, since the bottom-up information (i.e., the negative and possible positive data)



(a) *Nearest Updating* vs. *Nearest+Negative Updating*

(b) *Nearest+Negative Updating* vs. Our approach

Figure 1. Comparison of the distances of the ground truth data to the updated subspaces given by three schemes.

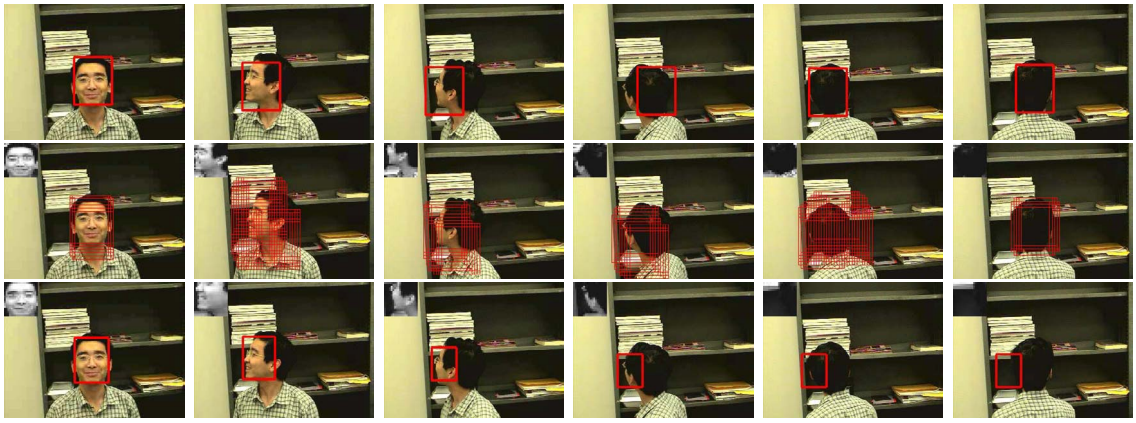


Figure 2. Tracking a head with 180^o rotation [`head180.avi`]. (top) our method, (middle) clustering, (bottom) *Nearest Updating*.

hints the emerging appearances, it can successfully track the head, although the bottom-up processing is quite rough.

5.5 Tracking a Head in Real Environments

Figure 6 shows the results of the experiment of tracking the head of a person walking in a real environment. The appearance of the head undergoes large changes, and there is also scale changes. Our result shows that *Nearest Updating* drifts to the background when the appearances of the black hair that the subspace has initially learned almost disappears. This happens when the person moves towards the camera. On the contrary, the proposed method can work comfortably and stably in the case. The details can be viewed in `walking.avi`.

5.6 Tracking with In-plane Rotations

In general, 2D in-plane rotation also induces significant changes to the target appearance. In this experiment, the black background is similar to the panel of the watch such that the adaptation in *Nearest Updating* deviates from the true subspace and it drifts rapidly. On the contrary, although the proposed method is also distracted at frame 444, it is able to recover quickly thanks to the help from the pairwise constraints. Sample frames are shown in Figure 7 and details in `watch.avi`.

5.7 Discussions

All the above experiments have validated the proposed approach. When the target model experiences drastic changes, we can explain the reason why the methods sharing the same nature as *Nearest Updating* deteriorates in two aspects. First, these methods trend to adhere to the old model as much as possible and are reluctant to include the changes. When the model changes completely or the original features disappear, the updated model will drift away from the true one eventually. Second, when the drift starts, there is no mechanism in these methods to force them back, thus the drift is unstable and catastrophic.

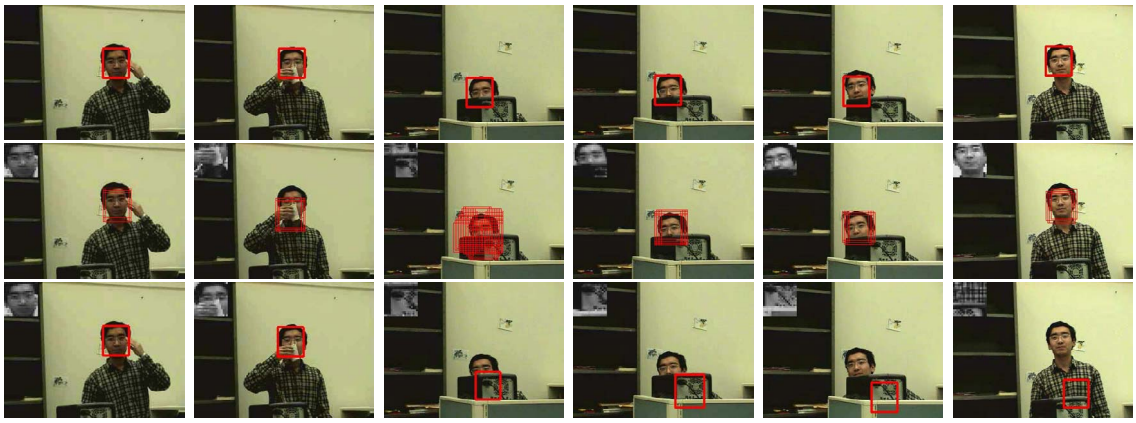


Figure 4. Tracking partial occlusion target [`face.avi`]. (top) our method, (middle) clustering, (bottom) *Nearest Updating*.

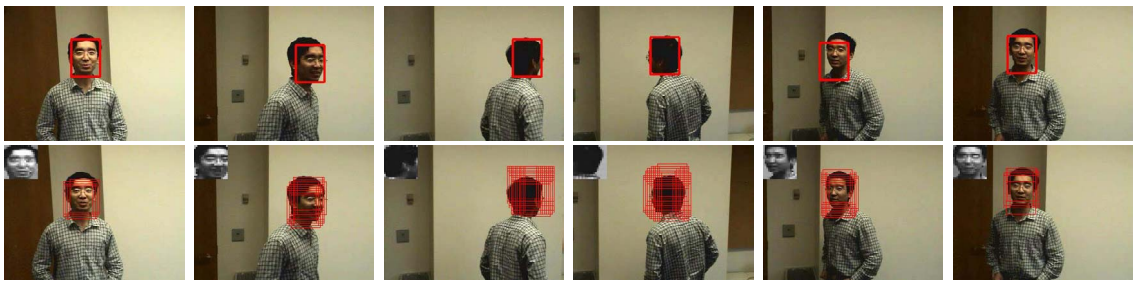


Figure 5. Tracking a head with 360^o out-of-plane rotation [`head360.avi`]. (top) our method, (bottom) clustering.

On the contrary, since our method utilizes the information from bottom-up, it can be thought as feedbacks that makes our method stable and avoids catastrophic drift to a large extent. As a result, the proposed method can be more robust and stable to cope with the adaptation drift.

6 Conclusions

We present an analysis of the adaptive tracking problem. If no constraints imposed, this problem is ill-posed. Different from the commonly used nearest updating scheme, we propose to impose both top-down smoothness constraints and the bottom-up data-driven constraints from current observances. Our method is based on the minimization balancing three factors: (1) distance of positive data to the subspace, (2) the projections of the negative data, and (3) the smoothness of two consecutive subspaces. The proposed method can largely alleviate the risk of adaptation drift and thus achieving better tracking performance.

Our further study will include the investigation of the situation when the smoothness constraint and bottom-up information are contradicted, and the best way of balancing and fusing these three types of constraints.

Appendix

Lemma 1 *The solution of the following problem:*

$$\min_A \mathtt{tr}(\mathbf{A}^T\mathbf{C}\mathbf{A}), \quad s.t., \quad \mathbf{A}^T\mathbf{A} = I, \tag{8}$$

where $\mathbf{A} \in \mathbb{R}^{m\times r}$, *and* $\mathbf{C} = \mathbf{Z}\mathbf{Z}^T \in \mathbb{R}^{m\times m}$, *is give by the eigenvectors that corresponds to the* r *smallest eigenvalues of* $\mathbf{C}$.

Proof: It is easy to figure it out. Actually this is the same as the proof of PCA.

Based on the Lemma, the proof of Theorem 1 is given by the following: Performing SVD on $\mathbf{A}_t$, we have $\mathbf{A}_t = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, where $\mathbf{U} \in \mathbb{R}^{m\times r}$, $\mathbf{\Sigma} \in \mathbb{R}^{r\times r}$, $\mathbf{V} \in \mathbb{R}^{r\times r}$. It is easy to see: $\mathbf{P}_t = \mathbf{U}\mathbf{U}^T$. Then the optimization problem in Eq. 3 is equivalent to:

$$\begin{aligned}\min_U J_3(\mathbf{U}) = \min_U \{&\mathtt{tr}(\mathbf{U}^T\mathbf{C}_t^-\mathbf{U}) - \mathtt{tr}(\mathbf{U}^T\mathbf{C}_t^+\mathbf{U}) \\ &+\alpha||\mathbf{U}\mathbf{U}^T - \mathbf{P}_{t-1}||_F^2\}, \quad s.t. \quad \mathbf{U}^T\mathbf{U} = \mathbf{I}.\end{aligned} \tag{9}$$

The Lagrangian is given by:

$$L(\mathbf{U}) = J_3(\mathbf{U}) + \lambda(\mathbf{U}^T\mathbf{U} - \mathbf{I}).$$

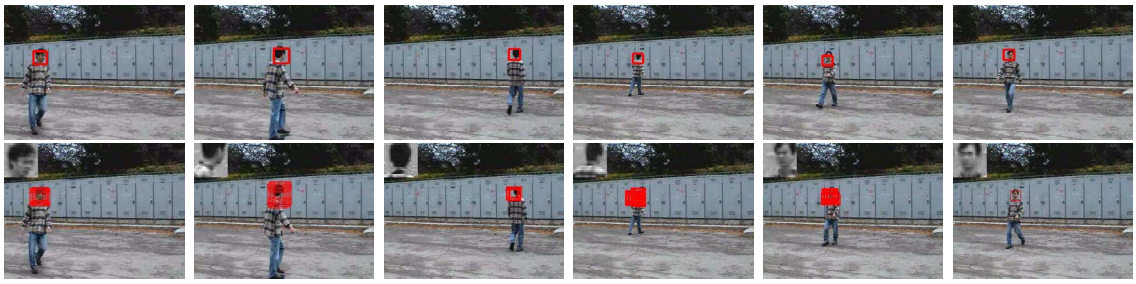


Figure 6. Tracking a head [`walking.avi`]. (top) our method, (bottom) clustering.

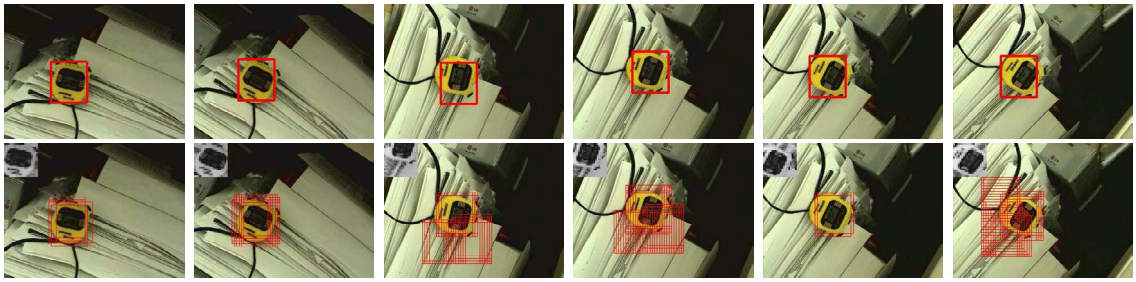


Figure 7. Tracking a watch with in-plane rotations [`watch.avi`]. (top) our method, (bottom) clustering.

Let $\mathbf{U} = [\mathbf{e}_1, \ldots, \mathbf{e}_r]$, and we have:

$$\begin{aligned} \frac{\partial L}{\partial \mathbf{e}} &= 2(\mathbf{C}_t^- - \mathbf{C}_t^+)\mathbf{e} + 2\alpha(\mathbf{e}\mathbf{e}^T - \mathbf{P}_{t-1})\mathbf{e} + 2\lambda\mathbf{e} \\ &= 2(\mathbf{C}_t^- - \mathbf{C}_t^+ + \alpha\mathbf{I} - \alpha\mathbf{P}_{t-1})\mathbf{e} + 2\lambda\mathbf{e}. \end{aligned}$$

Thus, $\mathbf{e}$ is an eigenvector of $\hat{\mathbf{C}} = \mathbf{C}_t^- - \mathbf{C}_t^+ + \alpha\mathbf{I} - \alpha\mathbf{P}_{t-1}$. The minimization problem is solved by finding the r eigenvectors that correspond to the r smallest eigenvalues of $\hat{\mathbf{C}}$. **Q.E.D.**

Acknowledgments

This work was supported in part by NSF Grant IIS-0347877, IIS-0308222 and Northwestern faculty startup funds.

References

[1] S. Avidan. Support vector tracking. *IEEE Trans. Pattern Anal. Machine Intell.*, 26(8):1064–1072, Aug. 2004.

[2] M. J. Black and A. D.Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. In *ECCV'96*, pages 329–342, Apr. 1996.

[3] R. T. Collins and Y. Liu. On-line selection of discriminative tracking features. In *ICCV'03*, volume 2, pages 346–352, Nice, France, Oct. 13-16, 2003.

[4] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *CVPR'00*, volume 2, pages 142–149, Hilton Head Island, South Carolina, June 13-15, 2000.

[5] G. Hager and P. Belhumeur. Real-time tracking of image regions with changes in geoemtry and illumination. In *CVPR'96*, pages 403–410, San Francisco, June 18-20, 1996.

[6] G. Hager, M. Dewan, and C. Stewart. Multiple kernel tracking with SSD. In *CVPR'04*, volume 1, pages 790–797, Washington, DC, Jun.27-Jul.2 2004.

[7] J. Ho, K.-C. Lee, M.-H. Yang, and D. Kriegman. Visual tracking using learned linear subspace. In *CVPR'04*, volume 1, pages 782–789, Jun.27-Jul.2 2004.

[8] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *ECCV'96*, pages 343–356, Cambridge, UK, 1996.

[9] A. Jepson, D. Fleet, and T. El-Maraghi. Robust online appearance models for visual tracking. In *CVPR'01*, volume 1, pages 415–422, Hawaii, Dec. 8-14, 2001.

[10] D. Ross, J. Lim, and M.-H. Yang. Adaptive probabilistic visual tracking with incremental subspace update. In *ECCV'04*, volume 1, pages 215–227, Prague, Czech Republic, May 2004.

[11] J. Shi and J. Malik. Normalized cuts and image segmentation. In *CVPR'97*, pages 731–737, June 17-19, 1997.

[12] K. Toyama and A. Blake. Probabilistic tracking in a metric space. In *ICCV'01*, volume 2, pages 50–57, Vancouver, Canada, July 7-14, 2001.

[13] J. Vermaak, P. Perez, M. Gangnet, and A. Blake. Towards improved observation models for visual tracking: Selective adaptation. In *ECCV'02*, volume 1, pages 645–660, Copenhagen, Denmark, May 2002.

[14] Y. Wu and T. S. Huang. Robust visual tracking by co-inference learning. In *ICCV'01*, volume 2, pages 26–33, Vancouver, Canada, July 7-14, 2001.

[15] B. Yang. Projection approximation subspace tracking. *IEEE Trans. Signal Processing*, 43(1):95–107, Jan. 1995.