VISUAL TRACKING VIA EFFICIENT KERNEL DISCRIMINANT SUBSPACE LEARNING

Chunhua Shen, Anton van den Hengel, Michael J. Brooks

[1]School of Computer Science, University of Adelaide, Australia [2] CSSIP, Australia
email: chhshen@cs.adelaide.edu.au

ABSTRACT

Robustly tracking moving objects in video sequences is one of the key problems in computer vision. In this paper we introduce a computationally efficient nonlinear kernel learning strategy to find a discriminative model which distinguishes the tracked object from the background. Principal Component Analysis and Linear Discriminant Analysis have been applied to this problem with some success. These techniques are limited, however, by the fact that they are capable only of identifying linear subspaces within the data. Kernel based methods, in contrast, are able to extract nonlinear subspaces, and thus represent more complex characteristics of the tracked object and background. This is a particular advantage when tracking deformable objects and where appearance changes due to the unstable illumination and pose occur. An efficient approximation to Kernel Discriminant Analysis using QR decomposition proposed by Xiong *et al.* [1] makes possible real-time updating of the optimal nonlinear subspace. We present a tracking method based on this result and show promising experimental results on real videos undergoing large pose and illumination changes.

1. INTRODUCTION

Robust appearance-based tracking of targets in video has attracted extensive research effort due to its wide range of potential applications, including, for example, video surveillance and human-machine interaction. One of the major problems affecting the performance of visual tracking systems has been the lack of suitable appearance models. For particle filtering based tracking algorithms, the equivalent problem is to find sufficiently robust likelihood models.

Due to the nature of visual tracking, occlusions and variations of the environment's illumination, the targets' pose *etc.* are inevitable. These dynamic components make adaptive observation models imperative. While the use of linear subspace models such as Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) algorithms can improve the tracker's reliability [2–4], we argue that nonlinear kernel methods such as Kernel Discriminant Analysis (KDA) are more suitable for separating the target from the background. The additional flexibility of KDA comes at a computational cost. The application of the method of Xiong et al. [1] reduces the computational burden of estimating a nonlinear subspace to the extent that the extra discriminatory power is well justified for tracking.

We present here a probabilistic Monte Carlo sampling based tracking algorithm. Instead of using colour (*e.g.* [5]) or contour features (*e.g.* [6]) to represent the target for tracking, we adaptively learn a subspace to represent the tracked object (as in [2–4]). Knowing the states $\mathbf{x}_{1:t-1}$ of the tracked object from time 1 to $t-1$ and under the Markovian assumption, we sample the states from the prior $p(\mathbf{x}_t|\mathbf{x}_{t-1})$. After computing the likelihood $p(\mathbf{z}_t|\mathbf{x}_t)$[1] for each sample, we find a *Maximum A Posteriori* (MAP) estimate for the current state $\mathbf{x}_t$.

Related work. It can be difficult to maintain reliable tracking performance over longer sequences with fixed target and background models because it is not possible to reflect the changes in appearance that inevitably occur. Many solutions to this problem have been proposed. Wu *et al.* have suggested switching observation models when tracking object contours [7]. The main limitation of their approach is that the switching process is determined *a priori* using a finite state machine. For colour feature based tracking, in [8], a method is proposed for evaluating multiple colour features during tracking, and for selecting the set of most discriminative features to improve tracking performances. Jepson *et al.* propose an elaborate mixture appearance model with an online EM updating algorithm in which the changes in image observations can be robustly adapted [9].

Object tracking with subspace models has been well studied [2]. Recently this topic has received renewed attention. In [3] an incremental PCA based tracker is presented. The method learns eigenspace representations online to reflect appearance changes of the observations. In [4], instead of using PCA (as in [3]), probabilistic LDA models are proposed to separate the target from the background. Compared with PCA models, LDA is a discriminative method and it utilises the background appearance as negative training data, thereby boosting the tracking performance.

However the abovementioned subspace methods are all linear and based on second-order statistics of the image set. Kernel subspace methods provide a more powerful image model that takes higher-order statistics into account. Better results obtained by kernel subspace methods have been reported over conventional subspace methods in various vision applications (*e.g.* [10–12]). One of the most relevant work to ours is in [12] where the authors explore the application of KDA to frontal face detection and promising experimental results are reported.

Replacing LDA with KDA for visual tracking requires that the added computational burden not be so large as to prevent real-time performance. The computational complexity of the conventional KDA is $\mathcal{O}(\ell^2 d + \ell^3)$ where ℓ is the training data size, d is the dimension of the data, and c is the number of classes [13]. This makes conventional KDA infeasible for real-time applications with

[1]$\mathbf{x}$ and $\mathbf{z}$ are states and observations respectively.

large training data. Recently Xiong *et al.* have developed a novel KDA algorithm based on QR decomposition [1], which is much more efficient while maintaining classification accuracy. Its complexity is only $\mathcal{O}(\ell dc)$. By contrast, the complexity of PCA is $\mathcal{O}(\ell^2 d)$. We adopt this efficient KDA to update the learned subspace every several frames while tracking.

Our approach. The method we propose in this paper bears some similarity to the work in [4]. However, we use efficient Kernel Discriminant Analysis which has proven better than conventional LDA as visual representations in recognition and detection. Schematically, our tracking algorithm is quite simple. The key of the algorithm is to learn subspace representations with efficient KDA. At time t, given a set of positive and negative (background) examples, a discriminative analysis is performed to obtain subspace representations for the target object. The positive examples might be the successful tracking results for $t = 1 \dots t-1$ or some previously collected training data. From the viewpoint classification, the estimated subspace maximises the margin between the target object class and the background class. Of course we might also be able to cast this learning as a probabilistic process, similar to [4, 14]. The details are presented in the following section.

The remainder of this paper is structured as follows. In Section 2 we present the details of the tracking algorithm. We shall focus on how to update the nonlinear subspace. In Section 3 we show the robustness of the algorithm through real video tracking under large pose and lighting variations. In the last section, we draw concluding remarks as well as possible future research issues.

2. PROBABILISTIC TRACKING ALGORITHM VIA KERNEL DISCRIMINANT LEARNING

In this section, we describe the specific components of our tracking framework. We first introduce the probabilistic tracking model. Then, for the sake of the completeness, we briefly review the KDA algorithm. The subspace update strategy is also discussed.

The entire tracking algorithm is a simplified variant of the well-developed Bayesian filtering tracking algorithm [6]. At each time frame t, we have the observations $\mathbf{z}_t$, together with the previous states $\mathbf{x}_{1:t-1}$, the task is to infer the latent state $\mathbf{x}_t$. Usually it is difficult to accurately estimate the system dynamics model $p(\mathbf{x}_t|\mathbf{x}_{1:t-1})$. Therefore in this work a Gaussian random walk is adopted, *i.e.*

$$p(\mathbf{x}_t|\mathbf{x}_{1:t-1}) = \mathcal{N}(\mathbf{x}_t; \mathbf{x}_{t-1}, \boldsymbol{\Sigma}_t) \tag{1}$$

where $\mathcal{N}(\cdot; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes a Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. This distribution is actually the *prior* for the inference at time t. According to Bayes' rule, $p(\mathbf{x}_t|\mathbf{z}_t, \mathbf{x}_{1:t-1}) \propto p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{1:t-1})$.

Following the Monte Carlo sampling philosophy, we draw N particles $\mathbf{x}_t^{(n)}$ ($n = 1 \dots N$) from the Gaussian prior. Each particle represents the possible position, posture and size of the target. We then evaluate each particle's likelihood $p(\mathbf{z}_t|\mathbf{x}_t^{(n)})$. Instead of gaining a Minimum Mean Square Error (MMSE) estimate of the posterior as in the particle filters, we compute an MAP estimate $\mathbf{x}_t^\star = \arg\max_{\mathbf{x}_t^{(n)}} p(\mathbf{z}|\mathbf{x}_t^{(n)})$. After the state $\mathbf{x}_t^\star$ is determined, the corresponding observation $\mathbf{z}_t^\star$ will be regarded as a positive example of the target. Those particles with large likelihood but away from $\mathbf{x}_t^\star$ will also be stored as a negative example. This training data selection strategy is identical as in [4]. The major differences are the subspace learning and updating strategies.

Kernel Discriminant Learning. PCA and LDA are limited by their nature to linear projections of the data while KDA is nonlinear discriminating tool using kernel-based techniques. PCA and LDA can be seen as assuming the noise in the data exhibits a Gaussian distribution, which is very restrictive. Let $\mathbf{A} \in \mathbb{R}^{d\times\ell}$ denote the training data matrix which contains ℓ column vectors $\mathbf{a}_i \in \mathbb{R}^d$. The size of the i-th class is denoted as ℓ_i. We can relax the Gaussian assumption by an appropriate nonlinear mapping from the original space to the feature space $\phi : \mathbb{R}^d \to \mathbb{R}^D$, $\mathbf{A} \to \phi(\mathbf{A})$ such that the mapped data follows a Gaussian in the high-dimensional feature space.[2] By employing the *kernel trick*, we can avoid the the explicit knowledge of the nonlinear mapping. Specifically, in order to calculate the inner product directly in the input space, a Mercer kernel $k(\mathbf{a}_i, \mathbf{a}_j) = \phi(\mathbf{a}_i)^\top \phi(\mathbf{a}_j)$ is introduced. In this paper, a Gaussian RBF kernel (up to an irrelevant multiplicative constant)

$$k(\mathbf{a}_i, \mathbf{a}_j) = \exp\left(-\frac{\|\mathbf{a}_i - \mathbf{a}_j\|^2}{2\sigma^2}\right)$$

is used, where σ is the bandwidth parameter and $\|\cdot\|$ denotes L_2 norm.

AS in the linear case, the KDA is written as an optimisation problem [13]

$$(\mathbf{P}^\phi)^\star = \arg\max_{\mathbf{P}^\phi \in \mathbb{R}^D} \left(\frac{(\mathbf{P}^\phi)^\top \mathbf{S}_b^\phi \mathbf{P}^\phi}{(\mathbf{P}^\phi)^\top \mathbf{S}_w^\phi \mathbf{P}^\phi}\right). \tag{2}$$

The difference is that both of the between- and within-scatter matrices $\mathbf{S}_b^\phi$, $\mathbf{S}_w^\phi$ are calculated in the feature space. And the projection $\mathbf{P}^\phi$ lies in the feature space as well. Eventually we arrive at an equivalent eigen-decomposition problem. See [13] for an explanation of how the kernel trick may be applied to reformat the optimisation. Two main drawbacks of the algorithm in [13] exist. Namely, it is unstable for sample training data applications and the computational load is large.

The method presented here employs an improved QR decomposition based KDA which is developed in [1]. The algorithm is divided into two stages. The first stage maximises the separation between different classes in the feature space via QR decomposition. At the second stage, the within-class distance is then be minimised. The core of the algorithm is given in the Appendix. Observing that the centroid of each class in the input space maps very close to the centroid in the feature space, the complexity of the KDA algorithm is reduced to $\mathcal{O}(\ell dc)$ [1]. See [1] for technical details and the computational complexity analysis.

Appearance update. In order to capture the change in appearance of foreground and background, both the training images' projections and the subspace representation need to be updated after a reasonable tracking duration. Suppose that at frame t, the tracker replaces the old training data with the positive and negative examples collected in the past $t-1$ frames. Thus at most 2ℓ training data need to be stored by the tracker. Then the KDA learning process is re-run to learn a new subspace.

[2]Typically $D \gg d$, and D could be infinite.

- **Initialisation:**
  Initialise the tracker. For face tracking, a face detector might be applied. Collect ℓ training data and learn a kernel subspace with KDA *a priori*. Select a template appearance $\mathbf{a}^{+\star}$ and calculate its kernel projection $(\mathbf{P}^{\phi})^{\top}\phi(\mathbf{a}^{+\star})$ using Equation (6). Set $t = 1$.
- **Sampling:**
  Sample N particles $\mathbf{x}_t^{(n)}, n = 1 \cdots N$ from the prior $p(\mathbf{x}_t|\mathbf{x}_{t-1})$. $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ is a Gaussian expressed in Equation (1).
- **State inference:**
  Estimate the state $\mathbf{x}_t^{\star}$ using Equations (4) and (5).
- **Subspace and appearance update:**
  For the interval of F frames, collect positive observations $\{\mathbf{a}_1^+, \ldots, \mathbf{a}_F^+\}$ as well as negative observations $\{\mathbf{a}_1^-, \ldots, \mathbf{a}_F^-\}$. Replace the $2F$ most "old" training data in the total ℓ training examples. Run Kernel Discriminant Analysis to update the subspace. Re-select the template appearance image.
- Set $t = t + 1$. Go to the **Sampling** step to process the next frame.

Figure 1: The KDA based tracking algorithm.

The template update process requires the determination of the subspace which best encapsulates the ℓ_+ positive training examples $\{\mathbf{a}_1^+, \ldots, \mathbf{a}_{\ell_+}^+\}$ from the previous tracking results. A simple treatment is to select the image from the training set according to the following criterion in the feature space:

$$\mathbf{a}^{+\star} = \arg\min_{i\in[1,\ell_+]} \left(\sum_{f=1}^{\ell_+} \|(\mathbf{P}^{\phi})^{\top}\phi(\mathbf{a}_f^+) - (\mathbf{P}^{\phi})^{\top}\phi(\mathbf{a}_i^+)\|^2 \right). \tag{3}$$

This criterion ensures the minimal total projection error between $\mathbf{a}^{+\star}$ and the positive training examples. While Equation (3) is computationally expensive, we can use an approximation in the input space. Towards this goal we seek $\mathbf{a}^{+*}$ which is closest to the geometric centroid of the positive training examples. We know that the centroid of each class in the input space maps close to the centroid in the feature space [1]. The $\mathbf{a}^{+\star}$ which satisfies the criterion (3) is also closest to the centroid in the feature space. Therefore $\mathbf{a}^{+*}$ is an approximation of the optimal $\mathbf{a}^{+\star}$.

This update process can be run at every F frames. The values for F and ℓ affect the efficiency and robustness of the tracking algorithm. Theoretically the smaller the value of F, the better the tracking performance, although at the cost of being more computationally demanding. It is always a trade off between the efficiency and robustness. The selection of an optimal ℓ is not straightforward. The value of ℓ should not be so large as to cause the use of too many observations from past frames. Too many examples irrelevant to the current observation will increase the probability of contaminating the current appearance model with "outliers".

Tracking algorithm. We represent the location of tracked object by a state vector $\mathbf{x} = (x, y, s_x, s_y, \theta)$, which determines a rectangular window in image space. x and y denote the centroid of the rectangle, s_x, s_y the width and length, and θ the orientation angle. The image content of each window, determined by a sample particle $\mathbf{x}_t^{(n)}$, is down-sampled to an image of fixed size in order to perform the subspace learning with the image observation. The kernel projection $(\mathbf{P}^{\phi})^{\top}\phi(\mathbf{z}_t^{(n)})$ is obtained using Equation (6). The L_2 distance in the projected feature space is

$$d_t^{(n)} = \|(\mathbf{P}^{\phi})^{\top}\phi(\mathbf{z}_t^{(n)}) - (\mathbf{P}^{\phi})^{\top}\phi(\mathbf{a}^{+\star})\|. \tag{4}$$

The likelihood probability is modelled as a Gaussian process as in [5], *i.e.*, $p(\mathbf{z}|\mathbf{x}_t^{(n)}) \propto \exp\left(-\rho \cdot (d_t^{(n)})^2\right)$, where ρ is a positive constant. It is easy to verify that

$$\mathbf{x}_t^{\star} = \arg\max_{\mathbf{x}_t^{(n)}} p(\mathbf{z}|\mathbf{x}_t^{(n)}) = \arg\min_{\mathbf{x}_t^{(n)}} d_t^{(n)}. \tag{5}$$

A disadvantage of this proposed algorithm is that the dimension of the discriminant space constructed by the KDA is limited to 2 for two-category classification. Thus the discrimination power is somewhat deteriorated. A higher dimensional discriminant space is obtained by a naïve remedy. We simply introduce multiple non-target classes. A more elegant modification to the KDA algorithm for solving this problem in the application of face detection has been proposed in [15].

To summarise, the KDA based tracking algorithm is depicted in Figure 1.

3. EVALUATION

Due to the space limit, we report an experiment on real video sequences to test the proposed tracker's capability on accurately tracking the object position and updating the appearance model.

Several frames from a real video sequence taken in an office environment are shown in Figure 2[3]. The resolution of the video is 320×240 and it is sampled at 30 frames per second. We use 500 particles in the experiment. The covariance matrix of the Gaussian distribution for the prior transition is determined empirically. Two free parameters involved in the KDA algorithm are σ for RBF kernel and δ for regularisation. We choose $\sigma = 230$ and $\delta = 0.10$.

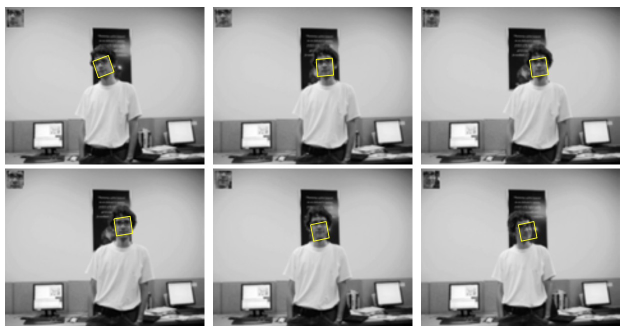

Figure 2: Tracking a human face undergoing large pose variations. The frame number are $3, 30, 40, 50, 61, 200$ respectively.

Our tracking algorithm can successfully track the human face undergoing large pose variations. The object location in the first

[2]For clarity we still use the symbol $\mathbf{z}_t$ to represent the re-sampled image observation.

[3]This video is courtesy of David Ross.

frame is manually initialised and the KDA learning is based on several frames at the beginning of tracking and then it is updated. We update the subspace every 10 frames. We see that the variation of the tracked object can be captured by the updated KDA subspace learning, while the fixed subspace tracker can drift easily during the tracking.

4. CONCLUSION

We have presented a tracking algorithm based on kernel discriminant subspace learning. The core of the method is an efficient QR decomposition based KDA algorithm which is suitable for real-time tracking applications. This efficient KDA allows the flexibility of nonlinear subspace learning without the prohibitive computational burden usually associated with such methods. Preliminary tests show that the algorithm performs well but more extensive testing is necessary.

Appendix

The basic procedure of the QR decomposition based KDA is as follows.

- Compute the centroid matrix $\mathbf{C}^{\phi}$ in the feature space, which contains the geometric centres of each class.
- Construct the kernel matrix $\mathbf{K}$, where $\mathbf{K}_{ij} = k(\mathbf{a}_i, \mathbf{a}_j)$. Then $(\mathbf{C}^{\phi})^{\top}(\mathbf{C}^{\phi}) = \mathbf{M}^{\top}\mathbf{K}\mathbf{M}$. The i-th column of $\mathbf{M}$ is defined as $(0, \cdots, 0, \frac{1}{\ell_i}, \cdots, \frac{1}{\ell_i}, 0, \cdots, 0)^{\top}$.
- Calculate Cholesky decomposition of $(\mathbf{C}^{\phi})^{\top}(\mathbf{C}^{\phi})$ to obtain $\mathbf{Q}^{\phi}, \mathbf{R}^{\phi}$.
- Construct
$$\mathbf{B}^{\phi} = (\mathbf{Q}^{\phi})^{\top}(\mathbf{S}_b^{\phi})(\mathbf{Q}^{\phi}),$$
$$\mathbf{T}^{\phi} = (\mathbf{Q}^{\phi})^{\top}(\mathbf{S}_t^{\phi})(\mathbf{Q}^{\phi}).$$
$\mathbf{S}_t$ is the total scatter matrix in the feature space. Note that both $\mathbf{B}^{\phi}$ $\mathbf{T}^{\phi}$ need to be calculated by the kernel trick. Refer to [1] for details.
- Calculate c eigenvectors $\boldsymbol{\lambda}_i$ $(i = 1 \cdots c)$ of $(\mathbf{T}^{\phi} + \delta\mathbf{I})^{-1}\mathbf{B}^{\phi}$ where $\mathbf{I}$ is an identity matrix and δ is a positive constant. c is number of classes.
- Define $\boldsymbol{\Lambda}^{\phi} = [\boldsymbol{\lambda}_1, \cdots, \boldsymbol{\lambda}_c]$ according to the order of decreasing corresponding eigenvalues. The final transformation matrix is $\mathbf{P}^{\phi} = \mathbf{C}^{\phi}(\mathbf{R}^{\phi})^{-1}\boldsymbol{\Lambda}^{\phi}$. Then given an input vector $\mathbf{z}$ in the original space, the projection is calculated with the kernel trick,
$$(\mathbf{P}^{\phi})^{\top}\phi(\mathbf{z}) = (\boldsymbol{\Lambda}^{\phi})^{\top}((\mathbf{R}^{\phi})^{-1})^{\top}\mathbf{M}^{\top}\mathbf{K}^{z}, \qquad (6)$$
where $\mathbf{K}^{z} \in \mathbb{R}^{\ell}$ and $\mathbf{K}_i^{z} = k(\mathbf{a}_i, \mathbf{z})$.

References

[1] T. Xiong, J. Ye, Q. Li, V. Cherkassky, and R. Janardan, “Efficient kernel discriminant analysis via QR decomposition,” in *Advances in Neural Information Processing Systems*. 2004, The MIT Press.

[2] M. J. Black and A. D. Jepson, “EigenTracking: Robust matching and tracking of articulated objects using a view-based representation,” *International Journal of Computer Vision*, vol. 26, no. 1, pp. 63–84, 1998.

[3] J. Lim, D. Ross, R.-S. Lin, and M.-H. Yang, “Incremental learning for visual tracking,” in *Advances in Neural Information Processing Systems*. 2004, The MIT Press.

[4] R.-S. Lin, D. Ross, J. Lim, and M.-H. Yang, “Adaptive discriminative generative model and its applications,” in *Advances in Neural Information Processing Systems*. 2004, The MIT Press.

[5] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet, “Color-based probabilistic tracking,” in *7th European Conference on Computer Vision*, Copenhagen, Denmark, 2002, vol. 2350 of *Lecture Notes in Computer Science*, pp. 661–675, Springer.

[6] M. Isard and A. Blake, “CONDENSATION – Conditional density propagation for visual tracking,” *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[7] Y. Wu, G. Hua, and T. Yu, “Switching observation models for contour tracking in clutter,” in *IEEE Conference on Computer Vision and Pattern Recognition*, Madison, WI, USA, 2003, vol. 1, pp. 295–302.

[8] R. T. Collins and Y. Liu, “On-line selection of discriminative tracking features,” in *IEEE International Conference on Computer Vision*, Nice, France, 2003, vol. 1.

[9] A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi, “Robust online appearance models for visual tracking,” in *IEEE Conference on Computer Vision and Pattern Recognition*, Kauai, 2001, vol. 1, pp. 415–422.

[10] J. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, “Face recognition using kernel direct discriminant analysis algorithms,” *IEEE Transactions on Neural Networks*, vol. 14, no. 1, pp. 117–126, January 2003.

[11] Y. Li, S. Gong, and H. Liddell, “Recognising trajectories of facial identities using kernel discriminant analysis,” *Image and Vision Computing*, vol. 21, no. 13-14, pp. 1077–1086, December 2003.

[12] Y. Feng and P. Shi, “Face detection based on kernel fisher discriminant analysis,” in *IEEE International Conference on Automatic Face and Gesture Recognition*, Seoul, Korea, 2004.

[13] G. Baudat and F. Anouar, “Generalized discriminant analysis using a kernel approach,” *Neural Computation*, vol. 12, no. 10, pp. 2385–2404, 2000.

[14] M. E. Tipping and C. M. Bishop, “Probabilistic principal component analysis,” *Journal of the Royal Statistical Society, Series B*, vol. 21, no. 3, pp. 611–622, 1999.

[15] Takio Kurita and T. Taguchi, “A modification of kernel-based fisher discriminant analysis for face detection,” in *IEEE International Conference on Automatic Face and Gesture Recognition*, Washinton D.C., 2002.