PROBABILISTIC VISUAL TRACKING VIA ROBUST TEMPLATE MATCHING AND INCREMENTAL SUBSPACE UPDATE

Xue Mei, Shaohua Kevin Zhou† and Fatih Porikli‡

University of Maryland, College Park, MD 20742
†Siemens Corporate Research, Princeton, NJ 08540
‡Mitsubishi Electric Research Labs, Cambridge, MA 02139

ABSTRACT

In this paper, we present a probabilistic algorithm for visual tracking that incorporates robust template matching and incremental subspace update. There are two template matching methods used in the tracker: one is robust to small perturbation and the other to background clutter. Each method yields a probability of matching. Further, the templates are modeled using mixed probabilities and updated once the templates in the library cannot capture the variation of object appearance. We also model the tracking history using a nonlinear subspace that is described by probabilistic kernel principal components analysis, which provides a third probability. The most-recent tracking result is added to the nonlinear subspace incrementally. This update is performed efficiently by augmenting the kernel Gram matrix with one row and one column. The product of the three probabilities is defined as the observation likelihood used in a particle filter to derive the tracking result. Experimental results demonstrate the efficiency and effectiveness of the proposed algorithm.

1. INTRODUCTION

Visual tracking is a critical task in many computer vision applications such as surveillance, robotics, human computer interaction, vehicle tracking and medical imaging, etc. The challenges in designing a robust visual tracking algorithm are caused by the presence of noise, occlusion, background clutter and the dynamic motion of target objects. A variety of tracking algorithms have been proposed to overcome these difficulties.

Early works used the sum of squared difference (SSD) as a cost function in the tracking problem [1]. A gradient descent algorithm was most commonly used to find the minimum. Subsequently, more robust similarity measures have been applied and the mean-shift algorithm or other optimization techniques utilized to find the optimal solution [2]. The graph-cut algorithm was also used for tracking by computing an illumination invariant optical flow field [3]. Subspace representations were successfully used for tracking by finding the minimum distance from the tracking object to the subspace spanned by the training data or previous tracking results [4, 5]. In [6, 7], tracking was implemented by maximizing the difference between the foreground and background. Recently, Porikli *et al* [8] propose an algorithm using a covariance based object description that fuses different types of features and modalities to successfully track non-rigid objects.

Stochastic tracking approaches reduce to an estimation problem, e.g., estimating the state parameter of a time series state space model. The problem is formulated in probabilistic terms. Early works used Kalman filter to provide optimal solutions for the linear Gaussian model. The particle filter, also known as the sequential Monte Carlo method [9], is the most popular approach which recursively approximates the posterior distribution of the state parameter. It has been developed in the computer vision community and applied to tracking problems under the name Condensation [10].

1.1. Algorithm Overview

This paper proposes a robust and adaptive appearance model for tracking noisy and complex objects. For each frame, S samples of the state parameter $\{x_1, x_2, \cdots, x_S\}$ are drawn from a Gaussian distribution, conditioned on the previous state. State samples x_i correspond to the windows W_i at various locations of different sizes and orientations in the image. The cropped windows, which are tracking result candidates, are compared with the templates to give the probabilities of template matching. Two template matching methods are suggested. One computes the Image Euclidean distance that is robust to small perturbation; the other computes the Image Weighted distance that is robust to background clutter. Both distances assign a weight based on the difference between pixel intensities. A third probability p_{KPCA} is provided by the probabilistic kernel Principal Component Analysis (PCA) that models a nonlinear subspace of the tracking history. The window which has the maximum product of the three probabilities is set as the tracking result in the current frame. The combination of these separate tracking algorithms has been justified in [11]. After tracking, we update the models to adapt to the most-recent appearance change. For the two template matching methods, the template library is modeled using mixed probabilities. The template with the smallest weight is replaced by the most-recent tracking result when the probability is below some threshold. The weight for each mixture component is updated adaptively. For the nonlinear subspace modeling of tracking history, the most-recent tracking result is added by augmenting the kernel Gram matrix with one more row and column. The standard particle filter algorithm is then used: the samples are resampled to eliminate particles with small importance weights and concentrate on particles with large weights. Tracking then proceeds to the next frame to repeat the same procedure.

The remainder of the paper is organized as follows. In the next section, we review the particle filter algorithm. Section 3 details the two template matching algorithms and how to update the template library. Section 4 presents the probabilistic kernel PCA used for modeling the tracking history. Experimental results are reported in section 5. We conclude this paper with a short summary.

2. PARTICLE FILTER

The particle filter is a Bayesian sequential importance sampling technique for estimating posterior distribution of state variables characterizing a dynamic system. It consists of essentially two steps: pre-

diction and update. The predicting distribution of x_t given all available observations $z_{1:t-1} = \{z_1, z_2, \cdots, z_{t-1}\}$ up to time $t-1$, denoted by $p(x_t|z_{1:t-1})$, is recursively computed as

$$p(x_t|z_{1:t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|z_{1:t-1})dx_{t-1} \quad (1)$$

At time t, the observation z_t is available and the state vector is updated using the Bayes's rule

$$p(x_t|z_{1:t}) = \frac{p(z_t|x_t)p(x_t|z_{1:t-1})}{p(z_t|z_{1:t-1})} \quad (2)$$

where $p(z_t|x_t)$ denotes the observation likelihood.

In the particle filter, the posterior $p(x_t|z_{1:t})$ is approximated by a finite set of N samples $\{x_t^i\}_{i=1,\cdots,N}$ with importance weights w_t^i. The candidate samples x_t^i are drawn from an importance distribution $q(x_t|x_{1:t-1}, z_{1:t})$ and the weights of the samples are updated as

$$w_t^i = w_{t-1}^i \frac{p(z_t|x_t^i)p(x_t^i|x_{t-1}^i)}{q(x_t|x_{1:t-1}, z_{1:t})} \quad (3)$$

3. TEMPLATE MATCHING

Template tracking works by exacting a template from the first frame and finding the object of interest in successive frames. SSD was used to calculate the distance between the template and the image. It gives the same weight to each pixel and is not robust to small perturbation and background clutter. The template has a specific shape, ellipse or rectangle, which inevitably includes some background pixels if the object is not perfectly in that shape or the tracker does not locate the object precisely. If the background has similar appearance as the object due to the illumination change, low contrast and noise, the appearance model cannot distinguish it from the background and tracking is prone to drift away. A fixed appearance template is not sufficient to handle recently changes in the video, while a rapidly changing model is susceptible to drift. In the algorithm proposed in [12], the updated template is aligned with the first template to obtain the final update in order to eliminate drift. Here we propose a robust template matching algorithm that handles small perturbation, background clutter and updating the template wisely.

3.1. Image Euclidean Distance

Wang *et al.* [13] propose a new Euclidean distance for images, which is dubbed as Image Euclidean Distance(IMED). Unlike the traditional Euclidean distance, IMED takes into account the spatial relationships of pixels. Therefore, it is robust to small perturbation.

In [13], it has been shown SSD is not a good metric to measure the image distance and a good one should contain the information of pixel distances. If the metric coefficients depend properly on the pixel distances, the computed Euclidean distance is insensitive to small deformation.

Suppose that two images z and t are rasterized to form two vectors, $z = (z_1, z_2, \cdots, z_{MN})$, $t = (t_1, t_2, \cdots, t_{MN})$, where z is the tracking result and t is the template, then the IMED is given by

$$d_{IME}^2(z,t) = \frac{1}{2\pi} \sum_{i,j=1}^{MN} exp\{-|P_i - P_j|^2/2\}(z_i - t_i)(z_j - t_j). \quad (4)$$

The probability of the presence of the tracked object given the template is written as

$$p_{IME}(z|t) = exp\{-d_{IME}^2(z,t)\} \quad (5)$$

which shows that small distance gives high probability.

3.2. Image Weighted Distance

When we use a rectangle or ellipse to select the region of interest, we inevitably include some background in the region of interest. The background will contaminate the template and contribute to tracking failure. Inspired by [2], we propose an image weighted distance method to overcome this problem.

Suppose w and h are the width and height of the image, respectively. The weight for the pixel at location (x, y) is

$$w(x,y) = 1 - \frac{1}{2}\{(\frac{x - x_0}{w/2})^2 + (\frac{y - y_0}{h/2})^2\} \quad (6)$$

where x_0 and y_0 is the center of the template. The weights are smaller for pixels that are farther from the center. Using these weights increases the robustness of matching since the peripheral pixels are the least reliable, being often affected by occlusion, clutter or interference from the background. The weight function is a 2D Gaussian kernel.

The probability of the object being tracked given the template is

$$p_{IMW}(z|t) = exp\{-d_{IMW}^2(z,t)\} \quad (7)$$

where $d_{IMW}^2(z,t) = \sum_{i=1}^{MN} w_i(z_i - t_i)^2$.

3.3. Template Update

The object appearance remains the same only for a certain period of time, but eventually the template is no longer an accurate model of the object appearance. If we do not update the template, the template cannot capture the variations in object appearance due to illumination or pose variations. If we update the template too often, small errors are introduced each time the template is updated. The errors are accumulated and the tracker drifts from the object. We propose an effective template updating strategy which achieves a compromise.

We attempt to overcome the drift problem by introducing a library of templates and using mixed probabilities. The mixture of Gaussian distributions is used in [14] to simultaneously model and track feature sets. The tracking probability for template matching given the template library $t_{1:K}$, $p(z|t_{1:K})$, is written as

$$\begin{aligned} p(z|t_{1:K}) &= p_{IME}(z|t_{1:K})\, p_{IMW}(z|t_{1:K}) \\ &= \{\sum_{k=1}^{K} w_k * p_{IME}(z|t_k)\}\, \{\sum_{k=1}^{K} w_k * p_{IMW}(z|t_k)\} \end{aligned} \quad (8)$$

where K is the number of templates in the library and w_k is the weight assigned to the template t_k.

When the tracker starts working, the template is added to the library when the probability $p(z|t_{1:k})$ is below some threshold p_0. After the number of templates in the library reaches the maximum value K, the template with the smallest weight is replaced by the tracking result when $p(z|t_{1:k}) < p_0$. The weight of each model increases when the appearance of the tracking object and template is close enough and decreases vice versa. In general, newly-added templates are less reliable than the old ones. The variation in structure and appearance may be due to transient environmental effects. To model the effect of a particular template gaining "trust", its weight is increased each time $p(z|t_k)_{k=1,2,\cdots,K}$ exceeds some threshold. The increase is at the expense of the other templates. We apply the strategy suggested in [14] to update the weight of the mixture model and the weights are updated as

$$w_i^{f+1} = \begin{cases} (w_i^f + \alpha)\frac{1}{1+\alpha} & \text{if } p(z|t_i) > p_{i0} \\ w_i^f \frac{1}{1+\alpha} & \text{otherwise,} \end{cases} \quad (9)$$

where p_{i0} is the threshold for template t_i, α is the learning rate such that $\alpha \in (0,1)$, and f is the frame number. The value of α sets the rate at which component rankings are changed and outmoded representations are removed and "trust" in new representations is gained.

4. INCREMENTAL NONLINEAR SUBSPACE UPDATE

PCA uses a low dimensional space to approximate a high dimensional space, but it restricts itself to a linear setting, where high-order statistical information is discarded. Kernel PCA overcomes this disadvantage by using a 'kernel trick'. The essential idea of the kernel PCA is to avoid the direct evaluation of the required dot product in a high-dimensional feature space using the kernel function. Hence, no explicit nonlinear mapping function projecting the data from the original space to the feature space is needed. Since a nonlinear function is used, albeit in an implicit fashion, high-order statistical information is captured. Probabilistic PCA [15] gives a probabilistic output by decomposing the data space into two subspaces, a principle subspace and a residual subspace. Kernel PCA is implemented by mapping the data space to a higher dimensional space using a nonlinear function. We propose an approach that analyzes kernel principal components in a probabilistic manner by using probabilistic kernel PCA.

For each frame, we need to update the current eigenbasis with the tracking result. In [16], the eigenbasis is updated without storing the covariances or the previous training examples using the probabilistic PCA for tracking. We show that when we use probabilistic kernel PCA, the eigenbasis can be incrementally updated very efficiently by augmenting the kernel Gram matrix with one row and column.

Below, we review the probabilistic kernel PCA and give the details on updating the kernel Gram matrix.

4.1. Probabilistic Analysis of Kernel Principal Components

Suppose that $x_1, x_2, \cdots, x_N$ are the given training samples in the original data space $\mathbb{R}^q$. Kernel PCA operates in a higher dimensional feature space induced by a nonlinear mapping function $\phi : \mathbb{R}^q \rightarrow \mathbb{R}^f$, where $f > q$ and f could even be infinite.

Probabilistic analysis assumes that the data in the feature space follows a special factor analysis model which related an f-dimensional data $\phi(x)$ to a latent q-dimensional variable z as

$$\phi(x) = \mu + Wz + \epsilon \tag{10}$$

where $z \sim N(0, I_q), \epsilon \sim N(0, \sigma^2 I_f)$, and W is a $f \times q$ loading matrix. Therefore, $\phi(x) \sim N(\mu, \Sigma)$, where $\Sigma = WW^T + \sigma^2 I_f$.

The probability has the form

$$p(\phi(x)) = (2\pi)^{-d/2}|\Sigma|^{-1/2} exp\{-\frac{1}{2}(\phi(x)-\mu)^T\Sigma^{-1}(\phi(x)-\mu)\} \tag{11}$$

The maximum likelihood estimates for μ and W are given by

$$\mu = \frac{1}{N}\sum_{n=1}^{N}\phi(x_n), \quad W = U_q(\Lambda_q - \sigma^2 I_q)^{1/2}R \tag{12}$$

where R is any $q \times q$ orthogonal matrix, and U_q and Λ_q contain the top q eigenvectors and eigenvalues of the C matrix, where $C = N^{-1}\sum_{n=1}^{N}(\phi_n - \phi_0)(\phi_n - \phi_0)^T$, where $\phi_0 = \mu, \phi_n = \phi(x_n)$.

The MLE for σ^2 is approximated as

$$\sigma^2 \simeq \frac{1}{f-q}\{tr(K) - tr(\Lambda_q)\} \tag{13}$$

where the $(i,j)th$ entry of the kernel Gram matrix K can be calculated as follows:

$$K_{ij} = \phi(x_i)^T\phi(x_j) = k(x_i, x_j) \tag{14}$$

4.2. Incremental Update of Kernel Gram Matrix

The nonlinear space is divided into M nonlinear subspaces, each subspace modelled by probabilistic kernel PCA. The final probability of the tracking object z is defined as a mixture distribution:

$$p_{KPCA}(\phi(z)) = \sum_{m=1}^{M} w_m * p_m(\phi(z)) \tag{15}$$

where $p_m(\phi(z))$ is the probability of kernel PCA in the m^{th} subspace and w_m is the mixture weight. The most-recent tracking result is added to the subspace which has the maximum probability $p_m(\phi(z))$. In order to obtain the probability of tracking, we only need to update the kernel Gram matrix K. The kernel Gram matrix is updated as:

$$K^{f+1} = \begin{pmatrix} K^f & k(x,y) \\ k^T(x,y) & k(y,y) \end{pmatrix} \tag{16}$$

where f is the index of the frame and $x_1, x_2, \cdots, x_n$ are all the points in the subspace, y is the current tracking result. $k(x,y) = (k(x_1,y), k(x_2,y), \cdots, k(x_n,y))^T$. This requires a lot of storage space because one has to save all the previous tracking results to calculate the kernel Gram matrix. To overcome this problem, we set a limit on the number of samples in each cluster. The oldest sample is discarded to leave the space for the most-recent one. We follow the same strategy used for template matching to update the weights for the subspaces.

5. EXPERIMENTS

We conducted numerous experiments to test whether our proposed tracking algorithm performs well in terms of following the object position and updating the appearance model. The tracking area is described by a rectangular window modeled by a 5-dimensional state vector $X = [x_0, y_0, w, h, \theta]$, where (x_0, y_0) represents the centroid of the tracking window, (w, h) are the width and height of the tracking window, and θ is the 2D rotation angle of the tracking window. Currently the parameters are initialized manually.

In the first experiment, two infrared image sequences consisting of 720×480 color videos, recorded at 30 frames/second were used. The target-to-background contrast is very low and the noise level is high for the IR images. Figure 1 shows the results of our tracker(left column) and the tracker using SSD in the template matching(right column). The second tracker is not robust to background clutter and includes the black bar as foreground when the vehicle passes it. Figure 2 contains the results of our tracker(left column) and the tracker using a naive template update strategy which updates template at each frame(right column). The presence of dust and smoke adds difficulties to the tracking of the vehicle. The second tracker sticks to the light background and cannot recover. Our method avoids these possibilities and is effective under low contrast and noisy situation and is robust to the background clutter.

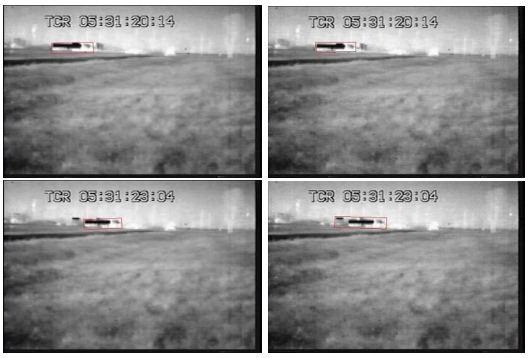

Fig. 1. Left column: Results of our tracker. Right column: Results of the tracker using SSD in the template matching

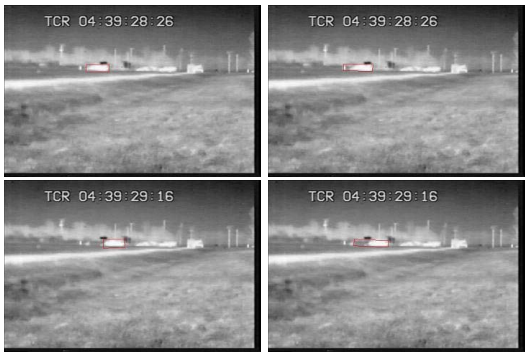

Fig. 2. Left column: Results of our tracker. Right column: the tracker using a naive template update strategy which updates template at each frame.

The third video sequence is from Carnegie Mellon University. The vehicle has a long shadow and turns a corner. Figure 3 shows the results of our tracker(top row) and the tracker without using subspace technique(bottom row). There is large scale change as the camera zooms in and out. The tracker without using subspace technique cannot handle the pose variation of the vehicle properly.

6. SUMMARY AND CONCLUSIONS

In this paper, we have presented an efficient and robust object tracking algorithm that uses template matching and incremental nonlinear subspace updating, based on the particle filter. We have demonstrated our algorithm for tracking various challenging sequences. Our algorithm requires no training data and can adaptively update the template and the subspace characterizing tracking history. It can be applied and play an important role on certain consumer electronics applications such as surveillance systems, user assisted metadata generation for consumer video, and so on.

7. REFERENCES

[1] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," *Int. J. Computer Vision.*, vol. 56, pp. 221–255, 2004.

[2] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, pp. 564–577, 2003.

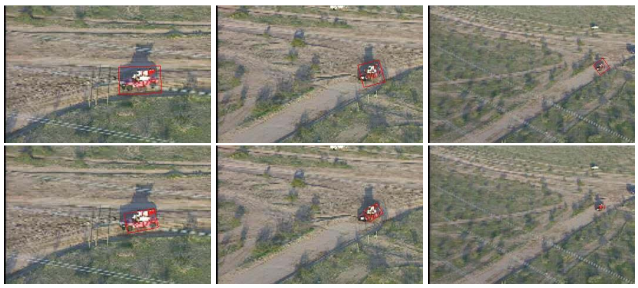

Fig. 3. Top row: Results of our tracker. Bottom row: Results of the tracker without using subspace technique.

[3] D. Freedman and M. W. Turek, "Illumination-invariant tracking via graph cuts," *In Proc. IEEE Conf. on Comp. Vision Patt. Recog.*, pp. 10–17, 2005.

[4] M. J. Black and A. D. Jepson, "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation," *Int. J. Computer Vision*, vol. 26, pp. 63–84, 1998.

[5] M.-H. Y. J. Ho, K.-C. Lee, and D. Kriegman, "Visual tracking using learned subspaces," *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, pp. 782–789, 2004.

[6] R. T. Collins and Y. Liu, "On-line selection of discriminative tracking features," *Proc. Int. Conf. Computer Vision*, pp. 346 – 352, 2003.

[7] S. Avidan, "Ensemble tracking," *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, pp. 494 – 501, 2005.

[8] F. Porikli, O. Tuzel, and P. Meer, "Covariance tracking using model update based on lie algebra," *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, pp. 728–735, 2006.

[9] A. Doucet, N. de Freitas, and N. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*, Springer-Verlag, New York, 2001.

[10] M. Isard and A. Blake, "Condensation - conditional density propagation for visual tracking," *Int. J. Computer Vision*, vol. 29, pp. 5–28, 1998.

[11] I. Leichter, M. Lindenbaum, and E. Rivlin, "A probabilistic framework for combining tracking algorithms," *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, pp. 445–451, 2004.

[12] I. Matthews, T. Ishikawa, and S. Baker, "The template update problem," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, pp. 810–815, 2004.

[13] L. Wang, Y. Zhang, and J. Feng, "On the euclidean distance of images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, pp. 1334–1339, 2005.

[14] N. Dowson and R. Bowden, "Simultaneous modeling and tracking (smat) of feature sets," *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, pp. 99–105, 2005.

[15] M. Tipping and C. Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society, Series B*, vol. 61, pp. 611–622, 1999.

[16] K.C. Lee and D. Kriegman, "Online learning of probabilistic appearance manifolds for video-based recognition and tracking," *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, pp. 852 – 859, 2005.