Multiple Faces Tracking Using Motion Prediction and IPCA in Particle Filters

Sukwon Choi and Daijin Kim

Intelligent Multimedia Laboratory, Dept. of Computer Science & Engineering,
Pohang University of Science and Technology (POSTECH), Pohang, Korea
{capriso,dkim}@postech.ac.kr

Abstract. We propose an efficient real-time face tracking system that can track fast moving face and cope with the illumination changes. To achieve these goals, we use the active appearance model(AAM) to represent the face image due to its simplicity and flexibility and take the particle filter framework to track the face image due to its robustness. We modify the particle filter framework as follows. To track fast moving face, we predict the motions using motion history and motion estimation, hence we can reduce the required number of particles. For observation model, we use active appearance model(AAM) to obtain an accurate face region, and update the model using incremental principle component analysis(IPCA). Occlusion handling scheme incorporates motion history to handle the moving face with occlusion. We have expanded our application to multiple faces tracking system. Experimental results present the robustness and effectiveness of the proposed system.

1 Introduction

Many research groups in the computer vision society have been interested in the topics related to the human face such as face detection, face recognition, face tracking, etc. Among them, face tracking is an important work that can be applied to several applications such as human-computer interaction, facial expression recognition, and robotics, etc.

There are several works that is related to the face tracking. Bayesian filters such as Kalman filter [1] and particle filter [2], [3], [4] are the most popular techniques for face tracking. The Kalman filter assumes that the state transition model obeys the Gaussian function. On the other hand, the particle filter assumes arbitrary model. The particle filter shows better performance on face tracking because the dynamics of the moving face obeys the non-linear/non-Gaussian function. Particle filter consists of observation model and state transition model. The observation model has to measure the likelihood of each particle. For this purpose, we need an appearance model. There are several researches about face modeling algorithm. Turk *et al.* [5] use PCA for eigenface analysis. Jepson *et al.* [6] introduce an online learning algorithm, namely online appearance model (OAM), which assumes that each pixel can be explained with mixture components, and Zhou *et al.* [7] modify it. However, these approaches cannot cope

with the facial variations such as illumination or expression. Cootes *et al.* [8] introduce AAM to represent the face using eigenface approach. The drawback of AAM is that all variations cannot be trained during training phase. Hamlaoui *et al.* [9] expand Zhou's work and use AAM as their appearance model, but there is no scheme to update the observation model.

The state transition model describes the dynamics of the moving objects between two frames. There are two ways to approximate the motion model: using a trained motion model by learning from video examples and using a fixed constant-velocity model. However, these approaches do not work well when the objects are moving very fast. The image registration technique introduced by Lucas and Kanade [10] is another approach that can be used for object tracking. However, their method is a gradient-based approach, which often traps into the local minimum. Zhou *et al.* [7] use adaptive velocity model to track the object effectively. However, if the particles are insufficient or the variance is relatively small, fast moving face cannot be tracked.

We propose to use IPCA [11], [12], [13] and motion prediction model to track the fast moving faces under the illumination change. This tracking system is expanded to multiple faces tracking for real-time system.

This paper is organized as follows: We review the particle filter in Section 2. In Section 3 and Section 4, we explain the observation model and the state transition model. We present how to handle the occlusion while the face is moving in Section 5. Section 6 shows how we expanded our system for multiple faces tracking. Next, the experimental results are presented in Section 7 and the conclusions are drawn in Section 8.

2 Particle Filter

The particle filter tries to estimate the states $\{\theta_1, \dots, \theta_t\}$ recursively using sampling technique. To estimate the states, the particle filter approximates the posterior distribution $p(\theta_t|Y_{1:t})$ with a set of samples $\{\theta_t^{(1)}, \dots, \theta_t^{(P)}\}$ and a noisy observation $\{Y_1, \dots, Y_t\}$. The particle filter consists of two components, observation model and state transition model. They can be defined as:

$$\begin{aligned} \text{State Transition Model} : \theta_t &= F_t(\theta_{t-1}, U_t), \\ \text{Observation Model} &: Y_t = H_t(\theta_t, V_t). \end{aligned} \tag{1}$$

The transition function F_t approximates the dynamics of the object being tracked using the previous state θ_{t-1} and the system noise U_t . The measurement function H_t models a relationship among the noisy observation Y_t , the hidden state θ_t , and the observation noise V_t . We can characterize transition probability $p(\theta_t|\theta_{t-1})$ with the state transition model, and likelihood $p(Y_t|\theta_t)$ with the observation model. We use maximum a posteriori (MAP) estimate to get the state estimate $\hat{\theta}_t$.

3 Observation Model

The observation model in the particle filter finds the relationship between the observed data and the state. We use AAM to represent the face image due to its simplicity and flexibility. We design the observation likelihood using the warped image and the reconstructed image of AAM. To make the observation model cope with the illumination change, we use IPCA to update the AAM basis vectors.

3.1 AAM-based Observation Model

We use the AAM for our observation model. Let us define the reconstructed image at time t as $\hat{I}_t$, and the warped image at time t as $\tilde{I}_t$. The AAM tries to minimize the AAM error, which is the distance between the reconstructed image and the warped image. Thus, we approximate the AAM error using Mahalanobis distance between $\hat{I}_{t-1}$ and $\tilde{I}_t$. To prevent the observation likelihood of a good particle from being spoiled by a few outliers, we use the robust statistics [14] to decrease the weight of outliers. The observation likelihood looks as :

$$p(I_t|\theta_t) \propto \exp \left(- \sum_{l=1}^N \rho \left(\frac{\tilde{I}_t(l) - \hat{I}_{t-1}(l)}{\sigma_R(l)} \right) \right), \quad (2)$$

$$\text{where } \rho(x) = \begin{cases} \frac{1}{2}x^2, & \text{if } |x| < \xi \\ \xi|x| - \frac{1}{2}\xi^2, & \text{if } |x| \geq \xi \end{cases}, \quad (3)$$

where N is the number of pixels in the template image, l is the pixel index, the σ_R is the standard deviation of reconstruction image, and ξ is a threshold that determines whether the pixel is an outlier or not.

We select the best particle, which has the largest observation likelihood, as the MAP estimate.

3.2 Update Observation Model

After estimating the state, we perform AAM fitting to get the accurate face region and then we can get the 2D global pose parameter $\hat{\theta}_t$. Then, we need to update the observation model to cope with the illumination change. Before we update the observation model, we need to check the goodness of fitting to determine whether we need to update the observation model or not. The AAM error is not appropriate for measuring the goodness of fitting, because the ill-fitted result and the illumination changes both increases the AAM error. For this purpose, we use the number of outliers using OAM. A pixel is declared as an outlier if the normalized pixel value, which is normalized by the mean and the variance of the OAM component, is larger than certain threshold ξ . We get the number of outliers for each component and take the average number of outliers, $N_{outlier}$. If $N_{outlier}$ is smaller than certain threshold N_0 , then we can update the AAM basis vectors using IPCA with new input image $\tilde{I}_t$.

4 State Transition Model

The state transition model describes the dynamics of moving object. In our system, we use motion history and motion estimation to predict the location of the face with low computation.

4.1 Motion Prediction Model Using Motion History

We implemented the tracking system of Zhou *et al.* [7] and evaluated the tracking performance and operating speed. Fig. 1-(a) shows the tracking performance when 20 particles are used to track the fast moving face, where the vertical axis denotes the pixel displacement between two consecutive frames and the circles denote the cases of failed tracking. We can see that the their tracking algorithm often fails when the pixel displacement between two consecutive frames is greater than 20. Fig. 1-(b) shows the average operating speed when the number of particles is changed. The number of particles should be less than 30 to guarantee the 15 frame per second. To realize the real-time face tracking, we propose the adaptive state transition model using the motion prediction model using the motion history.

Fig. 1. Tracking experimentation results of Zhou *et al.*[7] algorithm

We assume that the dynamics of moving face obeys the constant acceleration model. Then, the velocity $\mathbf{v}_{t-1}$ and the acceleration $\mathbf{a}_{t-1}$ at the time $t - 1$ is obtained as:

$$\mathbf{v}_{t-1} = \hat{\boldsymbol{\theta}}_{t-1} - \hat{\boldsymbol{\theta}}_{t-2}, \quad (4)$$

$$\mathbf{a}_{t-1} = \mathbf{v}_{t-1} - \mathbf{v}_{t-2}, \quad (5)$$

where $\boldsymbol{\theta}_{t-1}$ is the motion state that is the global 2D pose parameter of the AAM fitted image at the time $t - 1$. The motion velocity at the current time is predicted by the motion velocity and the acceleration at the previous time. Then, the velocity of moving face at the time t and the effective velocity $\bar{\mathbf{v}}$ between the two consecutive frames can be obtained as:

$$\tilde{\mathbf{v}}_t = \mathbf{v}_{t-1} + \mathbf{a}_{t-1}, \quad (6)$$

$$\bar{\mathbf{v}}_t = \frac{\mathbf{v}_{t-1} + \tilde{\mathbf{v}}_t}{2}. \quad (7)$$

In real situation, the actual motion of moving face does not obey the constant velocity model. So, the effective velocity obtained from the motion history model may be overestimated or underestimated. To overcome this problem, we suggest to use the motion prediction model that performs the motion estimation technique [15] around the estimated motion state $\bar{\theta}_t = \hat{\theta}_{t-1} + \bar{v}_t$.

In the motion estimation technique, each macroblock in the current frame is compared to a macroblock in the reference frame to find the best matching macroblock. To achieve this goal, block matching criterion such as the mean absolute error(MAE) or the mean squared error(MSE) are usually used. The computational load of finding the best matching macroblock is proportional to the size of macroblock.

To reduce the computation, we use the face detector based on Adaboost [16]. We can obtain the predicted motion state of the face θ'_t by applying motion estimation technique around $\bar{\theta}_t$. If the face detector fails, we use θ_t as the predicted motion state of the face as:

$$\tilde{\theta}_t = \begin{cases} \bar{\theta}_t, & \text{if the face is detected,} \\ \theta'_t, & \text{if the face is not detected.} \end{cases} \quad (8)$$

The particles are distributed around the predicted motion state of the face $\tilde{\theta}_t$ as in the following state transition model.

$$\theta_t^{(p)} = \tilde{\theta}_t + U_t^{(p)}, \quad (9)$$

where U_t is the system noise that follows Gaussian distribution $N(0, \sigma_t^2)$.

Optionally, we can include the adaptive velocity model into our state transition model. Since we can obtain the predicted motion state of the face using motion estimation technique, only a small number of particles is required. So, we can apply the adaptive velocity model without suffering from the heavy computational load. The particles are generated as:

$$\theta_t^{(p)} = \tilde{\theta}_t + V_{LS}\{I_t; \tilde{\theta}_t\} + U_t^{(p)}, \quad (10)$$

where $V_{LS}\{I_t; \tilde{\theta}_t\}$ is a function that computes the adaptive velocity at the starting point θ_t in the input image I_t .

4.2 Noise Variance and Number of Particles

We adaptively changes the noise variance and the number of particles. They are proportional to the Mahalanobis distance, D_t , between $\hat{I}_t$ and $\tilde{I}_t$ at $\hat{\theta}_t$.

$$\sigma_t = \sigma_0 \times \frac{D_t}{D_0}, \quad P_t = P_0 \times \frac{D_t}{D_0}. \quad (11)$$

We restrict the range of standard deviation as $[\sigma_{min}, \sigma_{max}]$ and the number of particles as $[P_{min}, P_{max}]$ to ensure certain degree of frame rate and performance.

5 Occlusion Handling

Our system detects occlusion using the number of outliers. If the face is occluded, the number of outliers $N_{outlier}$ is larger than certain threshold, N_0 , because the occlusion makes large number of outliers. When the occlusion is declared, we will get a bad tracking result if we use adaptive velocity model or face detector. Hence, we stop these methods and use only motion history with maximizing the number of particles and the variance. We can generate a new set of particles based on the Eq. (12).

$$\theta_t^{(p)} = \bar{\theta}_t + U_t^{(p)}. \quad (12)$$

6 Multiple Faces Tracking

We presented single face tracking algorithm so far. We have expanded our single face tracking algorithm to multiple faces tracking.

To check a new incoming face, we periodically invoke face detector. We use face detector based on Adaboost [16] learning. If we find a new face, then we detect eyes in the face region. With geometric information of eyes' position, we initialize the parameters of AAM, and perform AAM fitting. The result of the AAM fitting is used to initialize a new tracker. The overall procedure of multiple faces tracking algorithm is not quite different from the single tracking case. We use motion prediction model, and perform AAM fitting on the MAP estimate, then update the observation model if possible, for each tracker.

7 Experimental Results

We implemented the proposed system in Windows C++ environment. Our system processes at least 15 frames per second for single face tracking in Pentium 4 CPU with 3.0 GHz and 2 GB RAM with Logitech Web Camera. All the experiments in this section uses the parameters in the following manner. We captured the test sequences of size 320×240 with 15 fps. The size of AAM template image is 45×47 , and we used 5 shape/appearance basis vectors each. We set ξ as 1.7 to declare outlier, and N_0/N as 0.18 to declare occlusion. We performed 3 experiments to prove the robustness of the proposed face tracking system. In the first experiment, we showed that the proposed system is robust and effective in terms of the moving speed of faces, the required number of samples, and the tracking performance. The second experiment showed the performance of the tracking system under the occlusions. In the last experiment, we showed that our system can also handle the multiple faces tracking problem.

7.1 Tracking Fast Moving Face

We captured the test sequences with high/medium/low speed of moving face. We compared the face tracking system that uses the motion prediction model and the adaptive velocity model.

Fig. 2. The pixel displacement of the face in test sequences with high/medium/low speed of moving face

We obtained the pixel displacement of the face for each sequence as in Fig. 2. Tracking is performed using 20 particles. The horizontal axis denotes the frame number, and the vertical axis denotes the pixel displacement of the face. To measure the tracking accuracy, we compute the error between the estimated location and the ground truth of the face as $e_g = \sqrt{(q_x - g_x)^2 + (q_y - g_y)^2}$, where q_x, q_y are the horizontal/vertical translations of the estimated location of the face, and g_x, g_y are the translations of the ground truth. Fig. 3 compares the graph of e_g from each velocity models. As you can see, e_g of the motion prediction model shows lower value when the face moves fast. Fig. 4 shows the result of

Fig. 3. Tracking accuracy under the various speed of moving face

the face tracking using the test sequence with high speed of moving face. The motion prediction model successfully tracks the face, while the adaptive velocity model fails to track the face.

To compare the required number of particles, we performed another tracking experiment using the range of the number of particles as $[10, 60]$ and use 20 particles at initialization. Fig. 5 shows the required number of particles and the AAM error of the tracking systems with the two velocity models.

As shown in Fig. 5, we need fewer particles with the motion prediction model due to its better tracking performance, so we can save computations.

7.2 Occlusion Handling

We present the performance of our occlusion handling method. As shown in Fig. 6, the moving face is heavily occluded. On the upper right corner of each

Fig. 4. 1st row : original test sequence, 2nd row : tracking result of the adaptive velocity model, 3rd row : tracking result of the motion prediction model, 1-5 column : frame 155, 158, 161, 164, 167

Fig. 5. The required number of particles and the AAM error

Fig. 6. Occlusion handling, 1st row : frame 43, 56, 65, 2nd row : frame 277, 281, 285, 3rd row : frame 461, 463, 465

Fig. 7. Tracking results, 1st row : frame 190, 200, 210, 2nd row : frame 230, 330, 350

image, we can see small template image that displays outliers with white pixels. Our occlusion detection scheme detected outliers and occlusion successfully.

7.3 Multiple Faces Tracking

In this section, we present our multiple faces tracking system. Fig. 7 shows that our system is expanded to a multiple faces tracking system. We can see that scale, rotation, and translation of three faces are changing, but our system shows good tracking performance.

8 Conclusion

We presented a face tracking system that can track the fast moving face. The proposed system takes IPCA and motion prediction model to make the observation model and the state transition model adaptive. With these approaches, the face is successfully tracked using a few particles and small variance. We have found that these approaches generate particles more efficiently, and saved unnecessary computational load, and improved tracking performance. In the case of occlusion, we include only motion history to approximate the dynamics of the occluded face.

Our system has implemented and tested in real-time environment. The experimental results of the proposed face tracking system show the robustness and effectiveness in terms of the moving speed of faces, the required number of samples, the tracking performance, and occlusion handling. Also, we have applied our system and found that the system can handle the multiple faces properly.

Acknowledgement

This work was partially supported by the Korea Science and Engineering Foundation (KOSEF) through the Biometrics Engineering Research Center (BERC) at Yonsei University. Also, it was partially supported by the Intelligent Robotics Development Program, one of the 21st Century Frontier R&D Programs funded by the Ministry of Commerce, Industry and Energy of Korea.

References

1. Azarbayejani, A., Pentland, A.: Recursive estimation of motion, structure and focal length. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17, 562–575 (1995)
2. Doucet, A., Godsill, S.J., Andrieu, C.: On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing* 10(3), 197–209 (2000)
3. Arulampalam, S., Maskell, S., Gordon, N., Clapp, T.: A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing* 50(2), 174–189 (2002)

4. Okuma, K., Taleghani, A., de Freitas, N., Little, J., Lowe, D.: A boosted particle filter: Multitarget detection and tracking. In: *Proceedings of European Conference on Computer Vision*, pp. 28–39 (2004)
5. Turk, M., Pentland, A.: Eigenfaces for recognition. *Journal of Cognitive Neuroscience* 3(1), 72–86 (1991)
6. Jepson, A.D., Fleet, D.J., El-Maraghi, T.: Robust online appearance model for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25(10), 1296–1311 (2003)
7. Zhou, S., Chellappa, R., Moghaddam, B.: Visual tracking and recognition using appearance-adaptive models in particle filters. *IEEE Transactions on Image Processing* 13(11), 1491–1506 (2004)
8. Cootes, T.F., Edwards, G.J., Taylor, C.J.: Active appearance models. In: *Proceedings of 5th European Conference on Computer Vision*, pp. 484–498 (1998)
9. Hamlaoui, S., Davoine, F.: Facial action tracking using particle filters and active appearance models. In: *Joint sOc-EUSAI conference*, pp. 165–169 (2005)
10. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, pp. 674–679 (1981)
11. Hall, P., Marshall, D., Martin, R.: Incremental eigenanalysis for classification. In: *Proceedings of British Machine Vision Conference*, pp. 286–295 (1998)
12. Artac, M., Jogan, M., Leonardis, A.: Incremental PCA for on-line visual learning and recognition. In: *International Conference on Pattern Recognition*, pp. 781–784 (2002)
13. Ross, D.A., Lim, J., Yang, M.-H.: Adaptive probabilistic visual tracking with incremental subspace update. In: *Proceedings of 8th European Conference on Computer Vision*, vol. 2, pp. 470–482 (2004)
14. Huber, P.J.: *Robust statistics*. John Wiley, Chichester (1982)
15. Bhaskaran, V., Konstantinides, K.: *Image and video compression standards*. Kluwer Academic Publishers, Dordrecht (1997)
16. Froba, B., Ernst, A.: Face detection with the modified census transform. In: *Sixth IEEE International Conference on Automatic Face and Gesture Recognition*. IEEE Computer Society Press, Los Alamitos (2004)