Real-Time Expression Recognition System Using Active Appearance Model and EFM

Kyoung-Sic Cho, Yong-Guk Kim, and Yang-Bok Lee

Scholl of Computer Engineering Sejong University, Seoul, Korea
nopnse@sju.ac.kr, ykim@sejong.ac.kr, yangbok@empal.com

Abstract. We present a continuous facial expression recognition system based on Active Appearance Model (AAM) and Enhanced Fisher-Discriminant Model (EFM). AAM has been widely used in face tracking, face recognition, and object recognition tasks. In this study, we have implemented an independent AAM using Inverse Compositional Image Alignment method, which is a useful for the real-time system, because of its fast performance. The evaluation of this system carried out with the standard Cohn-Kanade facial expression database.

Keywords: Active Appearance Model(AAM.), Cohn-Kanade database, Enhanced Fisher-Discriminant Model(EFM).

1 Introduction

Facial expression recognition is one of the crucial ways to infer human emotion. Facial emotions are basically categorized into six facial expressions (surprise, fear, sadness, anger, disgust, happiness). The process flow of the present system is shown in Figure 1. Facial expression images are captured from a web camera. Active appearance model includes various shape and appearance parameters. With these images, AAM instance is generated, and its emotion is classified by EFM.

We first review the AAM in section 2. Then, the process of EFM classifier is described in Section 3. The performance of the system is evaluated using Cohn-Kanade facial expression database in Section 4. Finally, we will summarize our work in Section 5.

2 Active Appearance Model

AAM is firstly proposed in [2]. Typical applications of AAM are for modeling and recognizing the faces. And yet, AAM is also effective in modeling other object. In addition, its model can be transferred to other application. AAM created once for face recognition, for example, can be useful for facial expression recognition or tracking faces [1]. The main purpose of AAM is building a new model instance by finding the best matched parameters between input images and the model with a fitting algorithm. The fitting algorithm, which is a non-linear mode, iterates itself until the parameters of both shape and appearance satisfy some particular values. For instance, when the parameter of shape is measured, we can fit an input image on the coordinate frame of

Y. Wang, Y. Cheung, and H. Liu (Eds.): CIS 2006, LNAI 4456, pp. 1078–1084, 2007.

the model. After such match, the error between the instance of the model and pixels within the shape of input image could be acquired. This error would be applied for the fitting algorithm in order to update the parameters. Iterating this process would make optimized parameters through fitting algorithm. We adopted an Inverse Compositional Image Alignment, one of fitting algorithm used this system will be illustrated in section 2.2.

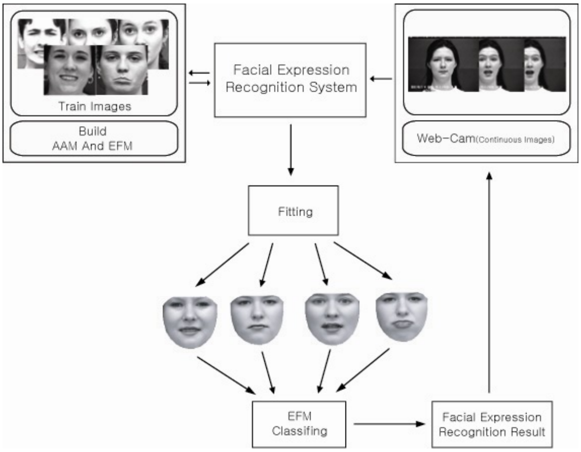

Fig. 1. Facial expression recognition system

2.1 Model Instance

Firstly, the shape of the AAM is created by combining the vectors which is made by marking on images manually.

$$\mathbf{s} = \mathbf{s}_0 + \sum_{i=1}^{n} p_i \mathbf{s}_i \tag{1}$$

In equation (1), p_i means the parameters of shape, s_0 indicates a base shape, and s_i represents shape vectors. The eigenvectors for the shape can be obtained by using Principal Component Analysis (PCA). They are the n eigenvectors corresponding to the n largest eigenvalues. Before applied PCA, AAM usually use Procrustes analysis in order to normalize Landmark points marked manually [1].

Secondly, the appearance of the AAM is defined by a pixel in a base mesh. Like the shape, appearance is also generated by the linear combination of the pixel intensity.

$$A(\mathbf{x}) = A_0(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i A_i(\mathbf{x}) \tag{2}$$

λ_i indicates the appearance parameters, A_i represents the appearance vectors, and A_0 is a base appearance. After finding both the shape parameters and the appearance parameters, the AAM instance is generated by locating each pixel of appearance to the inner side of the current shape with piecewise affine warp. A model instance is indicated as equation (3).

$$M(\mathbf{W}(\mathbf{x};\mathbf{p})) = A(\mathbf{x}) \tag{3}$$

The parameters of both shape and appearance are obtained by a fitting algorithm. Figure 2 shows the process of creating the model instance.

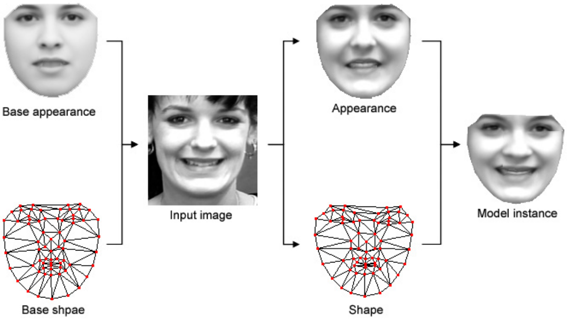

Fig. 2. Generation of an AAM instance

2.2 Inverse Compositional Image Alignment

The aim of the image alignment is to find a location of a fixed template image on an input image. Lucas and Kanade first proposed the Image Alignment Method [3]. Their algorithm is to locally align on the fixed template on an image, shown by equation (4).

$$\sum_{\mathbf{x}} \left\lfloor A_0(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p}))^2 \right\rfloor \tag{4}$$

The parameter p plays a role in minimizing errors between the fixed template $A_0(\mathrm{x})$ and the input image $I(\mathrm{x})$. In $\mathrm{x} = (x, y)^{\mathrm{T}}$ is the coordinator of pixel. The parameter p was linear, although $I(\mathrm{x})$ was non-linear. Thus, this formula is a non-linear optimization problem. To solve this problem linearly, Lucas-Kanade algorithm assumed that p was already known. Then, the algorithm increases the difference of p repeatedly, marked $\Delta\mathrm{p}$ in (5).

$$\sum_{\mathbf{x}} \left\lfloor A_0(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p}+\Delta\mathbf{p}))^2 \right\rfloor \tag{5}$$

The way of increasing p is that adding $\Delta\mathrm{p}$ to p. When changing the value of p, the performance of Lucas-Kanade algorithm is very slow because three parameters, Jacobian, gradient image, and Hessian Matrix have to be computed over and over. To improve this performance, Forwards Compositional Image Alignment method is introduced. In this algorithm, p is changed by combining the $\mathrm{W}(\mathrm{x};\mathrm{p})$ and $\mathrm{W}(\mathrm{x};\Delta\mathrm{p})$ as mentioned in equation (6).

$$\sum_{\mathbf{x}} \left\lfloor A_0(\mathbf{x}) - I(\mathbf{W}(\mathbf{W}(\mathbf{x};\Delta\mathbf{p});\mathbf{p}))^2 \right\rfloor \tag{6}$$

In Forwards Compositional Image Alignment method, we do not need to compute Jacobian every times, since the algorithm can compute Jacobian in $(\mathrm{x};0)$. In this paper, we adopt the Inverse Compositional Image Alignment method, in which we exchange the position of the input image and the template image. The ICIA can be formulated as an equation (7).

$$\sum_{\mathbf{x}} \left\lfloor I(\mathbf{W}(\mathbf{x};\mathbf{p})) - A_0(\mathbf{W}(\mathbf{x};\Delta\mathbf{p}))^2 \right\rfloor \tag{7}$$

The main advantage of this method is that the speed of updating parameters p can be very fast, since Jacobian and Gradient Image are calculated at $A_0(\mathrm{x})$. Once those values are acquired at the initial stage, we can use those values whenever updating a new warp parameter. Figure 3 shows three images of tracking a face using this method.

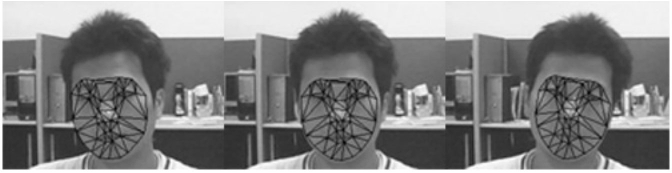

Fig. 3. Real-time face tracking using ICIA

3 EFM Algorithm

Let Y be a random vector representing the lower dimensional feature. Let $w_1, w_2, ..., w_L$ and $N_1, N_2, ..., N_L$ denote the classes and the number of images within each class, respectively. Let $M_1, M_2, ..., M_L$ and M be the mean of the classes and the grand mean. The within-and between-class covariance matrices Σw and Σb are defined as follows:

$$\Sigma w = \sum_{i=1}^{L} P(w_i) E\{(Y - M_i)(Y - M_i)^t \mid w_i\} \tag{8}$$

$$\Sigma b = \sum_{i=1}^{L} P(w_i)(M_i - M)(M_i - M)^t \tag{9}$$

The EFM firstly diagonalizes the within-class covariance matrix Σw.

$$\Sigma w \Xi = \Xi \Gamma \text{ and } \Xi^t \Xi = I, \tag{10}$$

$$\Gamma^{-1/2} \Xi^t \Sigma w \Xi \Gamma^{-1/2} = I, \tag{11}$$

where Ξ, Γ are the eigenvector and the diagonal eigenvalue matrices of Σw, respectively. The EFM then proceeds to compute the between-class covariance matrix as follows:

$$\Gamma^{-1/2} \Xi^t \Sigma b \Xi \Gamma^{-1/2} = K_b, \tag{12}$$

The EFM diagonalizes the new between-class covariance matrix K_b.

$$K_b \Theta = \Theta \Delta \text{ and } \Theta^t \Theta = I, \tag{13}$$

where Θ, Δ are the eigenvector and the diagonal eigenvalue matrices of K_b, respectively. The overall transformation matrix of the EFM is finally defined as follows:

$$T = \Xi \Gamma^{-1/2} \Theta \tag{14}$$

4 Experiments and Performance

Facial expressions used, for the present study, were 4: neutral, sadness, happiness, and surprise. AAM model was established using 498 face images, and each image marked by 68 landmark points. For 54 neutral, 50 sadness, 50 happiness, and 50 surprise images, the EFM model was set up. Experiments were divided into two evaluations: one was an evaluation about how correctly each model classified images of facial expressions, and another was a test about how exactly the system analyzed sequential images including various facial expressions.

4.1 Performance Evaluation

For the performance evaluation, we employed 5 Cross Validation method, in which we took 1/5 facial expression images as a test set, and then used the rest as the EFM train set. Thus, the number of tests was five times, and each result is shown in Table 1 and 2. The result shows that the worst cases were for the neutral and sadness facial expressions. In the confusion matrix, note that the error was large between the neutral and sadness cases. It seems that it is rather difficult to distinguish between two subtle facial expressions.

Table 1. Recognition Result

	Test Image No.	Success No.	Rate(%)
Neutral	54	46	85.1
Sadness	50	41	82.0
Happiness	50	46	92.0
Surprise	50	45	90.0
Total	204	178	87.2

Table 2. Confusion Matrix

	Neutral	Sadness	Happiness	Surprise
Neutral	46	6	0	2
Sadness	8	41	0	1
Happiness	2	2	46	0
Surprise	5	0	0	45

4.2 Continuous Expression Image Analysis

Since the Cohn-Kanade database consists of sequential images of a facial expression, we are able to test our system for the continuous facial expression recognition cases using it. For instance, Figure 4 shows how the happiness expression evolves from the neutral one, in particular, between image sequence 4 and 7. Here, the horizontal axis represents the sequence number of the images, whereas the vertical axis indicates EFM distance between the facial expression ranks. After the sequence 7, the happiness expression becomes a dominant one with a large distance from the other expressions. The system can process 15 frames per second.

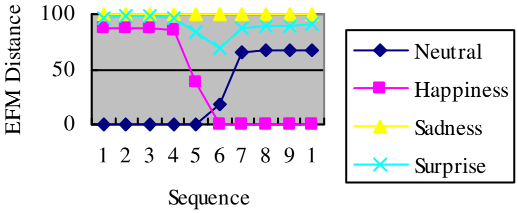

Fig. 5. Recognition of the surprise facial expression

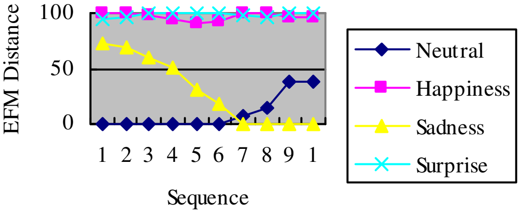

Fig. 6. Recognition of the sadness facial expression

Figure 6 shows that the sadness expression makes a distance from two expressions (happiness and surprise), as time goes by from sequence 2 to 6. And the neutral expression drifts from the bottom to upward direction. However, note that the distance between the neutral expression and the sadness one is not very far even at sequence 10, indicating that two expressions are, in a way, similar. This observation confirms the result of Table 2, in which the system appears to be confusing between two expressions.

5 Conclusions and Discussion

In this paper, we describe how real-time facial expression recognition system using AAM and EFM was implemented, and conducted two tests in order to examine the performance of the system. The result suggests that the system carried out facial expression recognition very well and it also operates in continuous facial expression recognition task. It is known that facial expression tends to accompany with head motion, we are working on a project that the head tracker is combined with the present facial expression tracker.

Acknowledgement

This work was supported by Seoul City Cluster Project.

References

1. Matthews, I., Baker, S.: Active Appearance Models revisited International Journal of Computer vision, 135–164 (2004)
2. Edwards, G.J., Taylor, C.J., Cootes, T.F.: Interpreting Face Images using Active Appearance Models. In: Proc. International Conference on Automatic Face and Gesture Recognition, pp. 300–305 (1998)
3. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of the International Joint Conference on Artificial Intelligence, pp. 674–679 (1981)