FACIAL EXPRESSION RECOGNITION USING AAM ALGORITHM

Thanh Nguyen Duc, Tan Nguyen Huu, Luy Nguyen Tan*

Division of Automatic Control, Ho Chi Minh University of Technology, Vietnam
*National key lab for Digital Control & System Engineering, Vietnam

ABSTRACT

Facial expression recognition is especially important in interaction between human and intelligent robots. Since the introduction of AAM model, there has been great change in detection accuracy. The main concern of this material is on facial expression recognition. In this paper, the recognition task is based on two methods, one of them is AAM combined with neural network, which gives better accuracy but lower speed, while the other is AAM combined with point correlation which is especially fast. Thus, it can be integrated to mobile robot platforms.
Keywords: Facial expression recognition, Active Appearance Models (AAM), Digital image processing

1. Introduction

Digital image processing is a task involved in capturing images from camera and analyzing the images, to extract the necessary information from the images. Facial expression recognition is not an exception of this rule. In this case, the information to be extracted is on special features of the face that relates to human feelings such as angry, normal happy, surprise.

W. Y. Zhao [1], B. Fasel [2] gave survey about facial expression recognition. Philip Michel & Rana El Kaliouby [3] used SVM (Support vector machines), the accuracy is over 60%. Ashish Kapoor [4] considered the movement of eyebrows. M.S. Barlett [5] combined Adaboost and SVM with the database DFAT-504 of Cohn & Kanade

In this article, AAM model [7] is used and combined with two other methods for recognitions. The first one is AAM combined with neural network, which would give us better accuracy, where as the second one is AAM combined with point correlation, which would give us an acceptable accuracy but a better speed.

2. Background theories on AAM

2.1. AAM Model

2.1.1.Building AAM model

Since AAM model was first introduced in [7], the use of this model has been increasing rapidly. An AAM model consists of two parts. AAM shape and AAM texture

2.1.1.1. AAM shape

According to [7], AAM shape is composed of the coordinates of v vertices make of the mesh:

$$s = [x_1, y_1, x_2, y_2, ..., x_v, y_v]^T \qquad (1.1)$$

Furthermore, AAM has linear shape, thus a shape vector s can be represented as:

$$s = s_0 + \sum_{i=1}^{N} p_i s_i = s_0 + P_s b_s \qquad (1.2)$$

Where s_0 is the base shape and $s_0, s_1, ..., s_N$ are orthogonal eigenvectors obtained from training shapes, $P_s = (s_1, s_2, ..., s_N)$ and $b_s = (p_1, p_2, ..., p_N)^T$

2.1.1.2. AAM texture

The AAM texture $A(x)$ is a vector defined as pixel intensities across objects x ϵ s_0, where $x = (x, y)^T$. Texture of AAM is also linear. Thus, it can be presented as:

$$A(x) = A_0(x) + \sum_{i=1}^{M} \lambda_i A_i(x) = A_0(x) + P_t b_t \quad (1.3)$$

$$\forall x \in s_0$$

Where $b_t = (\lambda_1, \lambda_2, ..., \lambda_M)^T$, $P_t = (t_1, t_2, ..., t_M)$ are orthogonal eigenvectors obtained from the training textures.

2.1.2.Fitting AAM model to an object

The goal of fitting AAM model is to find the best alignment to minimize the difference between the constant template $T(x)$ and an input image $I(x)$ with respect to warp parameters p. Let $x = (x, y)^T$ be the pixel coordinates and $W(x, p)$ denotes the set of parameterized allowed warps, where $p = (p_1, p_2, ..., p_N)^T$ is a vector of N parameters. The warp (x, p) takes the pixel x in the coordinate frame of a template T and map it to the sub-pixel location $W(x, p)$ in the coordinate of the input image-I. The Lucas-Kanade image alignment algorithm in [10] is to minimize:

$$\sum_x [I(W(x, p)) - T(x)]^2 \quad (1.4)$$

According to [10], to solve this, we assume that an estimation of p is known and then iteratively solve for the increment parameter Δp so that the following expression is minimized

$$A = \sum_x [I(W(x, p)) - T(W(x, \Delta p))]^2 \quad (1.5)$$

Taking Taylor expansion of T(W(x, Δp)) in terms of Δp at $\Delta p = 0$. This expression can be rewritten as

$$A = \sum_x [I(W(x, \Delta p)) - T(x) - \nabla T \frac{\partial W(x; p)}{\partial p}\Big|_{p=0} \Delta p]^2$$

(1.6)

Please note that, $W(x, 0) = x$ (because $p = 0$ does mean no changes at all)
The solution to minimize that expression can be easily found as followed:

$$\Delta p = H^{-1} \sum_x \left[\nabla T \frac{\partial W}{\partial p} \right]^T [I(W(x; p)) - T(x)]$$

(1.7)

Where H is the Hessian matrix.

$$H = \sum_x \left[\nabla T \frac{\partial W}{\partial p} \right]^T \left[\nabla T \frac{\partial W}{\partial p} \right] \quad (1.8)$$

Notice that the Jacobian matrix $\frac{\partial W}{\partial p}$ is calculated at $p = 0$ and the matrix T can be pre-computed. Thus, H can be pre-computed before every iteration. From these statements, according to [10], the fitting algorithm for AAM model can be summarized in the following steps

Table 1 Steps for fitting AAM model

Pre-computation	Iteration
•For every pixel x in the convex hull of AAM shape, obtain the intensity T(x) in the template image. •Calculate the gradient of T(x), which is ∇T. Calculate the Jacobian	•Start iteration at p = 0. •For each pixel x in the convex hull of the reference AAM shape, warp it to coordinate $W(x, p)$. Then, obtain the intensity $I(W(x, p))$ by

matrix $\frac{\partial W}{\partial p}$ at $(x,0)$. •Calculate the steepest decent image $\nabla T \frac{\partial W}{\partial p}$ •Calculate the Hessian matrix versus (1.8)	interpolation •Compute error image I(W(x, p)) – T(x). •Compute Δp using the pre-computed H and the formula (1.7) •If Δp is small enough, ends the iterations. Otherwise, update $\Delta p \leftarrow \Delta p + p$

3. Experimentation

3.1. Building and testing AAM model

In the training phase, we used more than 200 images of the model's face for 4 basic facial expressions which are normal, happy, surprise and angry taken in different lighting conditions. Each input image is marked with 66 points. The figure below shows some of the images that have been used in the training phase.

Fig. 1 Some input images

After performing Procustes' analysis [10] to eliminate the effect of similarity transforms on the input images (such as rotation, scaling, translation etc), we get normalized input shape vectors. Next, performing PCA analysis on these vectors, we get the base shape s_0 and other orthogonal shape eigenvectors.
Finally, performing PCA analysis on input image textures, after having warped them into base shape s_0, we got base texture $A_0(x)$ and other orthogonal texture eigenvectors.

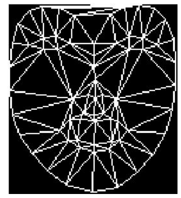

Fig. 2 Shape s_0

Fig. 3 Base texture $A_0(x)$

Figure 4 show us some examples of fitting the model face using AAM model. These images have not been previously used in training phase. The average number of iterations for each image is 5

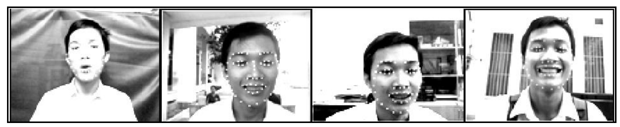

Fig. 4 Result of AAM face fitting

3.2. Facial Expression Recognition using AAM combined with neural network

3.2.1. Training neural network

In our experiment, we used about 30 images for each emotion. They are happy, normal, surprise, angry. For each image, we used the following procedure to extract the featuring vector.

1. Load the built AAM model.
2. Load the image need to extract the featuring vector together with its corresponding emotion vector E. The emotion vector E is a 4*1 vector, which belongs to one of the 4 following vectors $[0,0,0,1]^T$ (normal); $[0,0,1,0]^T$ (happy); $[0,1,0,0]^T$ (surprise); $[1,0,0,0]^T$ (angry).
3. Apply AAM model to track the original face (F) in the image and normalized it to another face (F') which is a texture defined on the base shape s_0.
4. Perform PCA analysis on F' using the textures $A_0(x), A_1(x), ..., A_M(x)$. We get

$$F^{'} = A_0 + v_1 A_1 + v_2 A_2 + ... + v_M A_M \quad (1.9)$$

5. The featuring vector for this image is

$$v = [v_1, v_2, ..., v_M]^T \quad (1.10)$$

Using the set of such (v, E) vectors, we trained the three-layer neural network, which has the following structures. Input layer: 62 neurons, which is also the number of eigen textures; hidden layer: 50 neurons; output layer: 4 neurons, which is also the number of emotions.

3.2.2.Testing neural network

For a testing input image, conducting the following procedures

1. Load the AAM model
2. Load the neural network.
3. Apply AAM model to track the original face (F) in the input image and normalized it to another face (F'), defined on the base shape s_0. This will help eliminate effects cause by similar transform and face rotation
4. Perform PCA analysis on F' using the base texture $A_0, A_1, ..., A_M$ obtained previously in the training phase. We get

$$F' = A_0 + v_1 A_1 + ... + v_M A_M \quad (1.11)$$

5. Using the featuring vector - $v = [v_1, v_2, ..., v_M]^T$ calculate and choosing the maximum output of the neural network. Then, infer the corresponding emotion.

3.2.3.Result

The experiment is implemented on a Compaq nx6120 laptop, running at 1.73 GHz, Ram 256 MB, using Visual C++ and open CV.

Table 2 Result of detection using AAM and MLP

Emotions	% of correction/True images (out of 75)
Normal	82.66% (62)
Happy	96.00%(72)
Surprise	85.33%(64)
Angry	84.00%(43)

The average time for each image is about **750** milliseconds. The testing images came from a Genius' Slim 1322AF webcam. Totally, 75 images were tested for each emotion.

3.3. Fast Facial Expression using point Correlation

3.3.1.Background knowledge

The Facial Expression Recognition based on MLP proved to be effective but it's fairly slow due to the reason of performing PCA analysis. In order to improve the speed of recognition, we suggest the idea of using point correlation in combine with AAM model. Let us consider 66 points on the face after having fit the AAM model to the face.

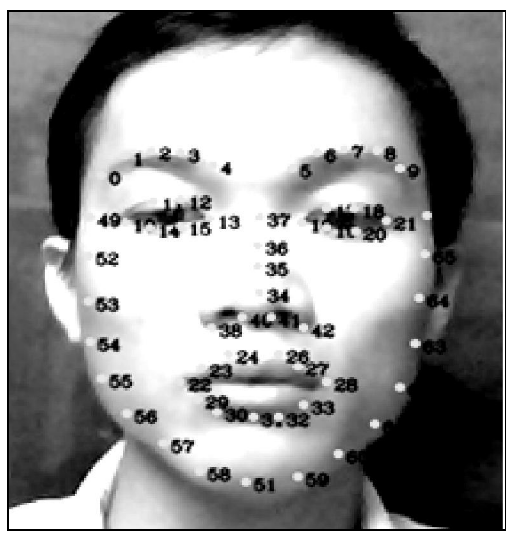

Fig. 5 66 face featuring points.

Let $d(m,n)$ be the Euclidean distance (in pixels) between the point m and n. Then, calculate the following ratios:

$$R_{mouth} = \frac{d(22,28)}{0.5d(17,19)+0.5d(12,15)} \quad (1.12)$$

$$R_{eye} = \frac{0.5d(17,19)+0.5d(12,15)}{d(37,34)} \quad (1.13)$$

$$R_{eyebrown} = \frac{d(4,5)}{d(13,16)} \quad (1.14)$$

On one hand, experimentation proves that the ratio R_{mouth} is relatively large when a person is happy or angry. On the other hand, it is small when he or she is normal or surprise. Besides, R_{eye} can be used to differentiate between normal and surprise emotions. Because, when a person is surprise, his eyes tend to open larger. Furthermore, $R_{\mathbf{eyebrown}}$ can be used to differentiate between angry and happy. This results from the fact that, when a person is angry, the distance between the eye-browns is enlarged.

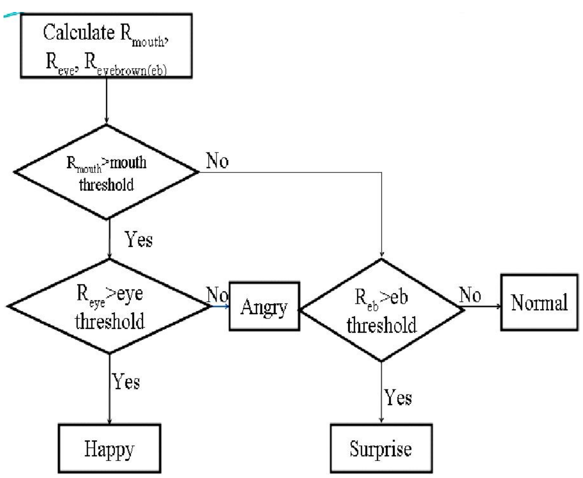

Fig. 6 Flowchart of Point Correlation method

The flowchart presented in figures 6 summaries our algorithm. In this flowchart, mouth threshold, eye threshold and eb threshold are three tunable parameters to be suitable for a specific person.

3.3.2.Result

Using test images in previous recognition method, we have the flowing result.

Table 3 Result of detection using AAM and Point Correlation

Emotion	% of correctness/ True images (out of 75 images)(*)
Normal	85.33%(64)
Happy	90.66%(68)
Surprise	81.33%(61)
Angry	82.66%(62)

The average processing time for a single image is about **250** ms, which is far more faster than the previous method. This is because most of the time is used for fitting AAM model. Besides, the distance calculations and comparing do not takes so much time.

(*)Conducted with mouth threshold = 7.2, eye threshold = 0.32, eb threshold = 0.90.

4. Conclusion

On the whole, facial recognition using AAM combined with neural network gives higher accuracy than that using point correlation method ,but point correlation give us faster recognition time. Thus, recognition using point correlation gives us a great chance to integrate the task into mobile platform, such as robots. However, that is over the scope of this article.

In order to improve the accurateness of recognition task, a better minimization techniques should be used, such as second-order minimization as in [11], but it will takes longer time for the algorithm to converge.

The experiment was conducted with a person-specific AAM model. In order to expand the system to recognize with various people, a lot of training textures and shapes should be used. In this case, the algorithm is exactly the same. Further research will be taken to get better accuracy and faster recognition time.

Many thanks for sponsor from the science foundation fund of VNUHCM

REFERENCES

1. W. Y. Zhao et al. , “Face Recognition: A Literature Survey”, UMD CfAR Technical Report CAR-TR-948, 2000.

2. B. Fasel and J. Luettin, "Automatic Facial Expression Analysis: A Survey", Pattern Recognition, Vol. 36(1), 2003, pp. 259-275.
3. Philip Michel & Rana El Kaliouby, Real time Facial expression recognition in Video using Support Vector Machines, University of Cambridge
4. Ashish Kapoor Yuan Qi Rosalind W. Picard, Fully Automatic Upper Facial Action Recognition, IEEE InternationalWorkshop on Analysis and Modeling of Faces and Gestures , Oct 2003
5. Marian Stewart Bartlett**,** Real Time Face Detection and Facial Expression Recognition: Development and Applications to Human Computer Interaction
6. CVPR Workshop on Computer Vision and Pattern Recognition for Human-Computer Interaction.
7. Marian Stewart Bartlett ,Towards social robots: Automatic evaluation of human-robot interaction by face detection and expression classification, Advances in Neural Information Processing Systems 2003.
8. G. J. Edwards, C. J. Taylor, and T. F. Cootes. Interpreting face images using active appearance models pp300-305, In Proc. International Conference on Automatic Face and Gesture Recognition, June 1998.
9. T. Cootes and P. Kittipanya-ngam. Comparing variations on the active appearance model algorithm. In 10th British Machine Vision Conference (BMVC 2002), pages 837{846, Cardi® University, September 2002.
10. I. Matthews and S. Baker. Active appearance models revisited. International Journal of Computer Vision, 60(2):135{164, November 2004.
11. Ezio Malis, "Improving vision-based control using efficient second-order minimization techniques," Proceedings of IEEE International Conference of Robotics and Automation, 2004.
12. Selim Benhimane and Ezio Malis, "Real-time image-based tracking of planes using Efficient Second-order Minimization," , pp. 943-948, Proceedings of IEEEIRSJ International Conference on Intelligent Robotics and Systems, 2004.