Face Animation Parameter Extraction in Real Video Environment

M. Ashourian[1], Sh. Mousavi[2], S.A. Monadjemi[3]
1- Assistant Professor, Faculty of Engineering, Islamic Azad University, Majlesi Branch, Iran,
Email: mohsena@ieee.org
2- MSc Student, Department of Computer Engineering, Faculty of Engineering, University of Isfahan, Isfahan, Iran
3- Assistant Professor, Department of Computer Engineering, Faculty of Engineering, University of Isfahan, Isfahan, Iran,
Email: monadjemi@eng.ui.ac.ir

Received: August 2009 Revised: October 2009 Accepted: December 2009

ABSTRACT:
One of the challenging problems in machine vision and computer graphic is realistic face modeling and animation. Using markers on the face key points and tracking the movements of the markers is a low cost method to determine these parameters. Since MPEG4 standard has defined parameters for face modeling and animation, we place markers on the key points defined in MPEG-4 standard, and propose a system for accurate tracking of them. We examine the system in real movie capturing conditions. Various distortions such as change of lightening, Gaussian noise, blurring, compression, and temporal filtering are examined, and the system degradation in each case is reported.

KEYWORDS: Face Animation, Face Modeling, Real Video, MPEG-4.

1. INTRODUCTION

Human face modeling and animation has many applications in machine vision and computer animation. There have been various approaches to locate and track face features. Eisert and Girod [1] develop the spline curve surface and get the face feature by optical flow, but their method cannot perform in real-time environments due to its computation complexity. Kass and Terzopoulos [2] proposed an Active Contour Model (ACM) and minimizing the energy function to locate the face feature points. Cootes [3] developed an Active Appearance Model (AAM) and set face shape and texture model by training examples to search the optimum face features. Blanz and Vetter [4] introduced the concept of linear combination model and Face Flexible Model to face reconstruction.

For facial landmarks tracking, adding markers on the human face was considered in many early work, such as heavy makeup [5] or using a set of colored dots [6], [7]. Once the position of the marker is determined, the facial motion features can be easily derived from image or video sequence. Williams [7] used it to get 2-D facial features from image. Guenter et al. [6] employed it to track 3-D facial features from video.

In our proposed scheme, we select the model of high-level face description defined in MPEG-4, and we adopted facial definition parameters (FDPs) and facial animation parameters (FAPs) to describe the facial behavior. Even though the simple combination of the above technology could be achieved with good, reliable performance, generally it is not really applicable in our case.

This paper deals with the tracking system of markers on the basis of MPEG-4 standard for real time conditions and where the captured video undergone various operations like addition of Gaussian noise, changing contrast and light, blurring, compression, and temporal filtering.

The organization of this paper is as follows. We begin with the face definition parameters in MPEG-4 standard. Then in Section II we present our algorithm for marker extraction and tracking. Section III provides the result of tracking system when the captured movie undergone various distortion. Finally, Section IV summarizes the result.

2. FACE DEFINITION AND ANIMATION IN MPEG-4

There are 84 feature points used to describe a face for both FAP and FDP in the MPEG-4 standard [8]. FDP gives the definition of face shape, size, and texture, while FAP gives the parameter description of facial deformation and expressions. Using these parameters, one can generate any possible facial expressions according to the specified FAP values. The

animation parameters are well designed in order to allow a satisfied implementation on any facial model ranging from video-realistic image warps to 3-D cartoon characters. Sections II-A and -B will show how FAP and FDP work.

A. FDP Set

One must have a generic facial model capable of interpreting FAPs. This insures that it can reproduce facial expressions and speech pronunciation for lip motion. MPEG-4 allows the encoder to completely specify the face model that the decoder has to animate. The FDPs can be used to personalize the generic face model by modifying the shape and appearance of the face to make it look like a particular person/character. The FDP fields include Feature Points Coord that specifies feature points for the calibration of the proprietary face, Texture Coords that specifies texture coordinates for the feature points, Texture Type that contains a hint to the decoder on the type of texture image, Face Def Tables that describe the behavior of FAPs for the face in the Face Scene Graph, and Face Scene Graph that can be used as a container of the texture image or the grouping node for the face model rendered in the compositor (therefore it has to contain the face model). Fig. 1 shows the feature points defined in FDP and FAP.

B. FAP Set

There are 68 FAPs categorized into 10 groups related to the parts of the face. FAPs manipulate key feature control points on a mesh model of the face to represent a complete set of basic facial actions, including head motion, tongue, eye, mouth control, and visemes (visual counterparts to phonemes). Since the FAPs are required to animate the face in different sizes and proportions, all parameters of FAP involving transnational movement are expressed in terms of FAP units (FAPUs). These correspond to fractions of distances among some essential facial features, e.g., eye distance (see Figure 1 and 2). The fractional units are chosen to maintain sufficient accuracy.

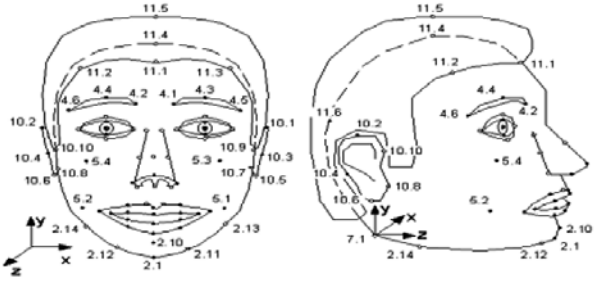

Fig. 1. The feature points definition for FDP and FAP.

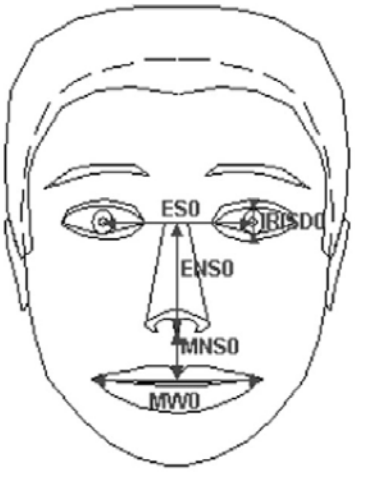

Fig. 2. Fractions of distances between some essential facial features

3. MARKERS TRACKING ALGORITHM

To track face markers, first the markers position should be determined in the first frame. We determine the markers using color and shape features [7]. The marker's color is usually selected blue or green to be much different from the skin color. In our experiments green markers have been used. Figure 3 shows a sample frame and its markers.

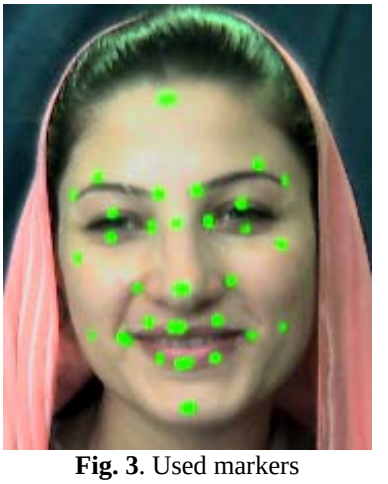

Fig. 3. Used markers

3.1. Markers Recognition in the First Frame

To find the markers, we use the color information in the RGB and HSV spaces. To recognize green color in RGB space the ratio of green color to sum of the colors is used. If this ratio is more than a threshold, the pixel is considered as a green pixel. If R, G, and B are the quantities of the color of one pixel in RGB space, the green marker pixel is specified through the following equation:

$$\frac{G}{R+G+B} > t_1 \Rightarrow GreenPixel \qquad (1)$$

Similarly, Euclidean distance in RGB space is used for the color vectors. If R, G, and B are the quantities of color of one pixel in the RGB space and index *p* shows the used pixel and index *t* shows the average pixel of markers, then we will have the following relation (2):

$$\sqrt{R_p - R_t)^2 + (G_p - G_t)^2 + (B_p - B_t)^2} > t_2 \quad (2)$$

It is clear that parameters t_1 and t_2 are the boundaries [8]. In HSV space, the third dimension specifies the amount of light, but does not contain the color data; therefore, we do not employ it. We select a pixel to be as marker if it satisfies the condition in both spaces. The limit boundary quantities are calculated through the trimming of the color data by the user or automatically.

3.2. Markers Tracking in Successive Frames

To track markers in successive frames the position of markers in previous frames is used as an initial point to recognize its position in new frame [9], [10]. We consider a searching window within the current frame for each marker around its position in the previous frame. We find the new position of markers in the new window. It is possible that we do not find any new marker position within the window which means that the marker has moved very fast and left the window. To solve this problem, we select larger searching windows. Also it is possible that more than one marker is found in the searching window. We need to find the true marker position. We assume the marker that has closer distance to the original position is the true candidate.

The size of the searching window is calculated according to the markers average size and distance.

Table 1. Number of markers in different states

More than one marker in window	One marker in window	None marker in window	Number of all markers
1104	1626	3	2730
3434	5596	2	9030
1386	1824	3	3210
938	1402	2	2340
1088	1642	2	2730
1367	1813	4	3180
3572	5098	3	8670
4170	5760	5	9930
4360	6020	4	10380
3346	4814	3	8160
3607	5423	4	9030

As you see, on the average 40% of the markers must be tracked within the window with more than one marker. Error tracking will happen in two states: It is possible that marker situation cannot be calculated since the loss of markers in the searching window, or since the calculated situation is not conforming to the real situation and has been calculated by mistake. The second state is more common. In this state the wrong calculated situation is relevant to one of the markers near the marker under discussion.

This kind of error is called *conflict error* and will be used as an error criterion in tracking. Conflict error is calculated comparing to the total markers. In Table (2) marker loss error and conflict error have been illustrated for each change in the movies. Present quantities in the following table are the average error for all used movies.

In markers tracking, it is possible that no marker is found in the searching window or one marker is found, or more than one marker is found. The statistics related to these cases are presented in Table (1).

Table 2. Conflict error

Conflict error	Lost error	Movie conditions
0.07 %	0 %	Normal
0.7 %	0 %	Gaussian noise
4.5 %	5.7 %	Filtering
0.4 %	0.3 %	Picture blurring
0.07 %	0 %	Changing contrast
1.4 %	1.2 %	Compression

If the problem of conflict is not solved accurately in a frame, this problem will be repeated for the next frames and will completely disrupt the operation of markers tracking.

4. SYSTEM ROBUSTNESS

The tested database consists of 12 movies which have been taken from the opposite sides of a person's face with 30 markers. The markers have been placed on the face according to the defined parameter points in MPEG-4 standard. Each movie has 150 frames on average which are recorded at 30 frames per second.

We applied various distortive operations on the captured video and compared the result of original movie with the distorted one. The operations were:

- Addition of Gaussian Noise
- Changing Contrast
- Blurring
- MPEG-2 Compression
- Temporal Filtering, employing the FIR filter defined as:

$$y(n) = 0.95x(n) + 0.05x(n-1) \quad (3)$$

The goal was to deal with the effect of these processing on markers tracking.

In Figures 4 to 8 the route of different markers in real state and under the influence of each set of operations are depicted.

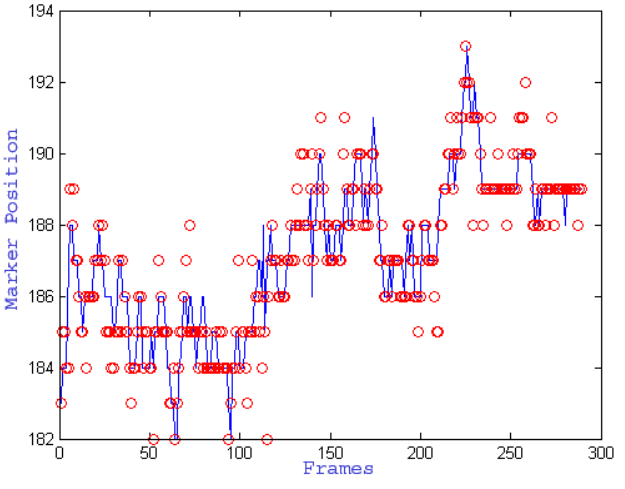

Fig. 4. Effect of Gaussian noise addition; Solid lines: the marker real route, Circles: after addition of Gaussian noise

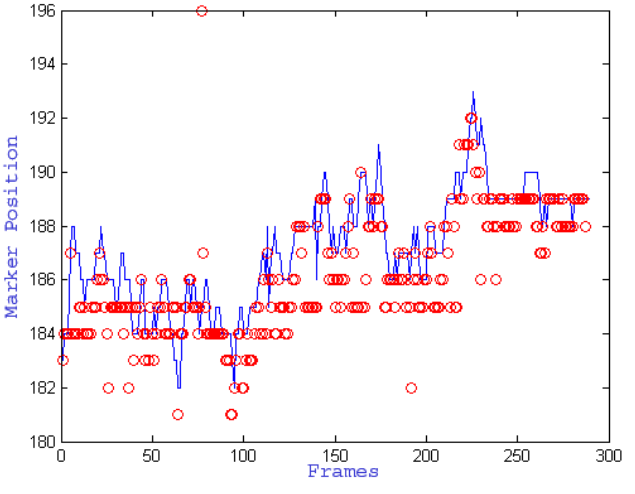

Fig. 5. Effect of temporal filtering; Solid lines: the marker real route, Circles: after temporal filtering

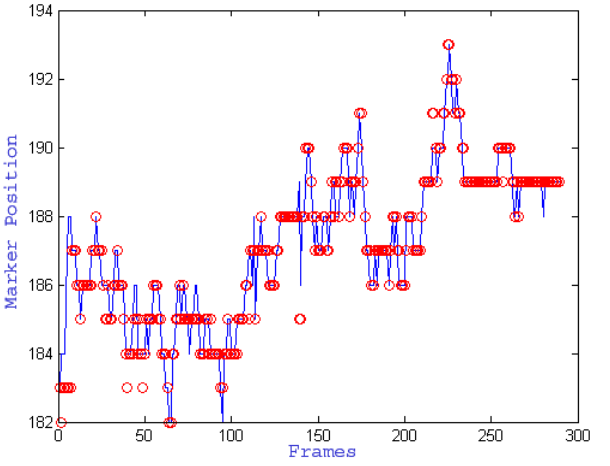

Fig. 6. Effect of blurring; Solid lines: the marker real route, Circles: after blurring

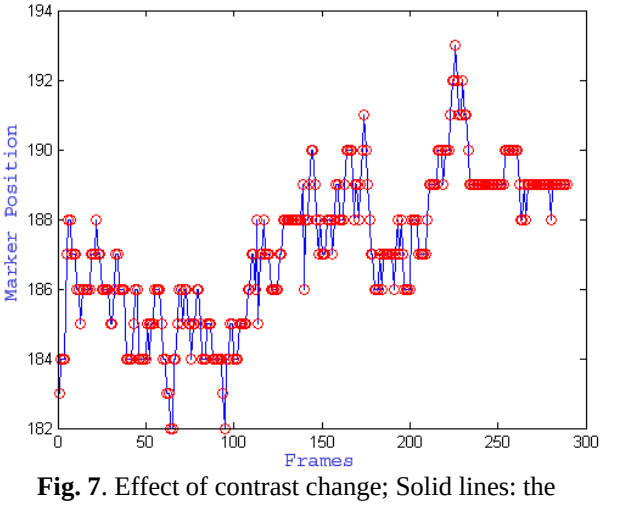

Fig. 7. Effect of contrast change; Solid lines: the marker real route, Circles: after contrast change

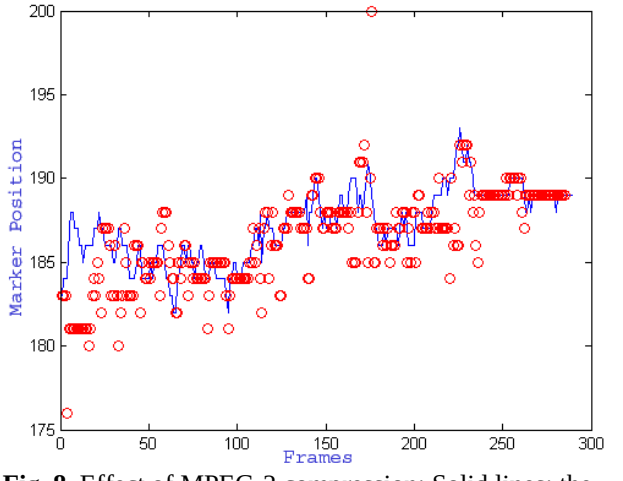

Fig. 8. Effect of MPEG-2 compression; Solid lines: the marker real route, Circles: after MPEG-2 compression

5. CONCLUSION

We developed a system for accurate face modeling using face markers based on the MPEG-4 defined parameters and standard. The system performance improved using various techniques for adaptive tracking of the markers. We also evaluate the system when the captured movie undergone different distortions such as compression, temporal filtering, contrast changing, noise addition, and blurring. The temporal filtering resulted in high error rate, but noise addition and contrast changing did not impose any error on the system, which suggests the robustness of the proposed scheme.

REFERENCES

[1] Eisert P. and Girod B.; **"Analyzing Facial Expressions for Virtual Conferencing"**, *IEEE Computer Graph*. pp. 70 – 78, (April 1998)

[2] Kass M., Witkin A. and Terzopoulos D.; "**Snakes: Active Contour Models**", *Int. J. Computer Vision*, pp. 321 – 331, (1988)

[3] Cootes T.F. and Kittipanya-ngam P.; **Comparing Variations on The Active Appearance Model Algorithm**, in: Proc. of BMVC, Vol. 2, pp. 837 – 846, (2002)

[4] Blanz V., Romdhani B.S. and Vetter T.; **Face Identification Across Different Poses and Illuminations with a 3d Morphable Model**, *Proceedings of the IEEE International Conference on Automatic*.

[5] Terzopoulos D. and Waters K.; **"Analysis and Synthesis of Facial Image Sequences Using Physical and Anatomical Models,"** *IEEE Trans. Pattern Anal. Machine Intell*., pp. 569 – 579, (1993)

[6] Guenter B., Grimm C., Wood D., Malvar H. and Pighin F.; **"Making Faces"** *in Proc. SIGGRAPH*, pp. 55–66, (July 1998)

[7] Williams L.; **"Performance-Driven Facial Animation"** in Proc. SIGGRAPH, Vol. 24, (Auguest 1990), pp. 235 – 242.

[8] Ostermann J.; **"Animation of Synthetic Faces in Mpeg-4"** *Computer Animation*, pp. 49 – 51, (June 1998)

[9] Ostermann J. and Weissenfeld A.; **"Talking Faces - Technologies and Applications"**, *IEEE International Conference on Pattern Recognition*, Vol. 3, pp. 826 - 833, (August 2004)

[10] Sarvan A., Arsla L. and Akarun L.; **"Speech Driven MPEG-4 Facial Animation for Turkish"**, *Proceedings SPECOM, St.Petersburg*, (September 2004)