A NEW METHOD FOR GENERATING 3-D FACE MODELS FOR PERSONALIZED USER INTERACTION*

A. Tanju Erdem

Research and Development, Momentum A.S.
TUBITAK TEKSEB A-205, Gebze, Kocaeli, TURKEY
email: terdem@momentum-dmt.com
web: www.momentum-dmt.com

ABSTRACT

A new method for generating and animating a 3-D model of a person's face is proposed. The method involves novel algorithms for 2-D to 3-D construction under perspective projection model, real-time mesh deformation using a lower-resolution control mesh, and texture image creation that involves texture blending in 3-D. The resulting face models can be readily used in 3-D games, mobile messaging, e-learning applications such as lip-reading and personalized training, 3-D movies and 3-D TV programs, and for providing human-computer interaction (HCI). User interfaces with personalized face models could be used to guide people in accessing knowledge and information on the Internet or to facilitate the usage of computers and software. Thus, it is believed that the utilization of personalized 3-D face models will help bring down the barrier between computers and people.

1. INTRODUCTION

The methods disclosed in the literature for generating a 3-D face model can be generally classified as those that involve (i) a fully manual process, (ii) a semi-automatic process and (iii) a fully-automatic process. Fully manual processes are labor intensive and time consuming because every triangular patch of the 3-D face mesh has to be manually mapped onto the image of the face according to the facial features of the face.

Semi-automatic processes [1, 2] rely on automatically detecting or manually marking certain features on the face, such as eyes, nose, and mouth, and initialize the 3-D mesh by a global affine warping of a standard 3-D mesh based on the location of the detected facial features. However, a global affine transformation generally does not match all local facial features. Thus, the locations of the nodes are fine-tuned in a manual process for each person. Fully automatic processes drastically reduce the time required to map an image onto a 3-D mesh. However, while hardware based fully automatic processes [3, 4] are very costly, software based fully automatic processes [5, 6] are very sensitive to the image data and thus may not consistently produce accurate 3-D face models.

The facial animation methods on the other hand can be generally classified as (i) physics-based methods [7, 8] and (ii) rule-based methods [9, 10]. In physics-based methods, dynamic models of the facial muscles are employed to calculate the propagation of any facial force throughout the face and to obtain the resulting deformation of the surface of the face. Physics-based methods may produce realistic animations, however, because of their high computational cost, they may not be used in real-time applications.

In rule-based methods, a set of feature points is used to control the movement of the rest of the nodes of the face mesh. Each feature point is assigned an area of influence on the face mesh. When a feature point is moved, the nodes of the face mesh that belong to the area of influence of the feature point move according to some predefined deformation rules. Although the rule-based methods provide real-time deformations of the face, they may lack realism as they are not based on any physical model.

This paper proposes a new semi-automatic method for generating the 3-D face mesh with minimal manual fine tuning and a new method for animating the 3-D face mesh in real time via a lower-resolution 3-D control mesh.

2. METHOD FOR GENERATING 3-D FACE MODEL

The steps of the proposed 3-D face generation method are as follows. First, 2-D neutral images of the person's face are captured with a camera from 5 different directions, namely, front, forehead, chin, right, and left, as shown in Fig. 1. The following fourteen locations on the person's face are specified as the feature points: the centers of the right and left eye pupils, the central end points of the right and left eyebrows, the right and left corners and the tip of the nose, the top and bottom points of the right and left ears, and the right and left corners and the center of the lips. The locations of feature points are manually marked on all five neutral images where they are visible.

2.1 Calculating the 3-D Positions of the Feature Points and the Orientation of the Face in the 2-D Images

Given the 2-D locations of the feature points in the neutral images where they are visible, the 3-D positions of the feature points of the person's face are calculated using a modified version of the method in [11] as follows. Without loss of generality, assume that the coordinate axes of the camera system are the unit vectors $\hat{i} = (1,0,0)$, $\hat{j} = (0,1,0)$, and $\hat{k} = (0,0,1)$. Hence, the image plane passes at $(0,0,-F)$ and is perpendicular to $\hat{k}$, where F denotes the focal length of camera in meters. Let N denote the number of feature points and P_n, $n = 1,...,N$, denote the coordinates of the

*This work was supported by EC within FP6 under Grant 511568 with the acronym 3DTV, and by TUBITAK TIDEB under Grant 3040012.

feature points with respect to the origin $(0,0,0)$ of the camera system. Clearly, as the person's face is moved, the coordinates, $P_n,\ n=1,...,N,$ of all the feature points are changed. It is therefore more appropriate to use a local coordinate system for the face to represent the coordinates of the feature points. Let the unit vectors $\tilde{i}$, $\tilde{j}$, and $\tilde{k}$ denote the coordinate axes for an arbitrary local coordinate system for the face. The origin C_0 of the local coordinate system is defined to be the centroid of the feature points and is given by

$$C_0 = \frac{1}{N}\sum_{n=1}^{N} P_n \cdot$$

Furthermore, let $A_n,\ n=1,...,N,$ denote the coordinates of the feature points with respect to the origin of the local coordinate system. Thus, as the person's face is moved, the origin of the local coordinate system is changed but the local coordinates $A_n,\ n=1,...,N,$ of the feature points always remain fixed.

In order to relate the global coordinates $P_n,\ n=1,...,N,$ to the local coordinates $A_n,\ n=1,...,N,$ define the unit vectors $\hat{I}=(\tilde{i}_x,\tilde{j}_x,\tilde{k}_x)$, $\hat{J}=(\tilde{i}_y,\tilde{j}_y,\tilde{k}_y)$, $\hat{K}=(\tilde{i}_z,\tilde{j}_z,\tilde{k}_z)$, where the subscripts x, y, and z, denote the coordinates of the respective vectors along the axes $\hat{i}$, $\hat{j}$, and $\hat{k}$ of the global coordinate system. Then, the relationship between the global coordinates and the local coordinates of the feature points is given by

$$P_{n,x} = C_{0,x} + A_n \bullet \hat{I}, \quad P_{n,y} = C_{0,y} + A_n \bullet \hat{J}, \text{ and}$$
$$P_{n,z} = C_{0,z} + A_n \bullet \hat{K},$$

where $\bullet$ denotes the inner product of two vectors. Finally, the 2-D coordinates of the feature points projected on the image plane are expressed as

$$p_{n,x} = \frac{1}{E}\frac{P_{n,x}}{P_{n,z}} = \frac{1}{E}\frac{C_{0,x} + A_n \bullet \hat{I}}{C_{0,z} + A_n \bullet \hat{K}}, \text{ and}$$
$$p_{n,y} = \frac{1}{E}\frac{P_{n,y}}{P_{n,z}} = \frac{1}{E}\frac{C_{0,y} + A_n \bullet \hat{J}}{C_{0,z} + A_n \bullet \hat{K}},$$

where the quantities on the left hand side are in units of pixels while the quantities of the right hand side, except the *E* parameter of the camera and the unit vectors, are in units of meters. The *E* parameter is defined as the inverse of *DF* where *F* is the focal length of camera and *D* is meters-to-pixels conversion factor for the camera. The above equations can be rewritten with all quantities in units of pixels as follows:

$$p_{n,x} = \frac{c_{0,x} + S_n \bullet \hat{I}}{\lambda + ES_n \bullet \hat{K}}, \text{ and}$$
$$p_{n,y} = \frac{c_{0,y} + S_n \bullet \hat{J}}{\lambda + ES_n \bullet \hat{K}}, \quad (1)$$

where

$$c_{0,x} = \frac{C_{0,x}}{EW},\ c_{0,y} = \frac{C_{0,y}}{EW},\ \lambda = \frac{C_{0,z}}{W}, \text{ and } S_n = \frac{A_n}{EW},$$

where W is an arbitrary scaling constant in meters.

The method of [11] assumes a special form of 2-D projection, namely, the orthographic projection. In orthographic projection, it is assumed that $C_{0,z}$ is the same for all images, $W = C_{0,z}$, and $W >> 1$. Thus, the above perspective projection equations reduce to the following form in the case of *orthographic* projections:

$$p^f{}_{n,x} = c^f{}_{0,x} + S_n \bullet \hat{I}^f, \text{ and}$$
$$p^f{}_{n,y} = c^f{}_{0,y} + S_n \bullet \hat{J}^f,$$

where f denotes the frame number. The quantities on the left hand side are measured quantities while the quantities on the right hand side are unknown quantities. The method of [11] solves the above equations for the 3-D local coordinates S_n of the feature points, and the orientation vectors $\hat{I}^f$ and $\hat{J}^f$ and the 2-D position $(c^f{}_{0,x}, c^f{}_{0,y})$ of the centroid of the feature points in all images given the 2-D projected positions $(p^f{}_{n,x}, p^f{}_{n,y})$ of the feature points in all images.

In the following, a modification to the method of [11] is presented in order to solve the perspective 2-D projection equations (1) given above for the 3-D local coordinates S_n of the feature points and the orientation vectors $\hat{I}^f$ and $\hat{J}^f$, the 2-D position $(c^f{}_{0,x}, c^f{}_{0,y})$ of the centroid of the feature points, and the distance ratio λ^f in all images given the 2-D projected positions $(p^f{}_{n,x}, p^f{}_{n,y})$ of the feature points in all images. Note that the third orientation vector $\hat{K}^f$ is uniquely defined by the first two orientation vectors $\hat{I}^f$ and $\hat{J}^f$ simply as $\hat{K}^f = \hat{I}^f \times \hat{J}^f$ where $\times$ denotes the vector outer product. The proposed modification method is an iterative procedure whose steps are as given below:

1. Use the motion-and-shape-estimation method of [11] that employs the orthographic projection equations to calculate S_n for $n=1,...,N,$ and $\hat{I}^f$, $\hat{J}^f$ and $(c^f{}_{0,x}, c^f{}_{0,y})$ for $f=1,...,F,$ given the 2-D measurements $(p^f{}_{n,x}, p^f{}_{n,y})$ and the visibility information of the feature points. Note that $N = 14$ and $F = 5$. Let $\hat{K}^f = \hat{I}^f \times \hat{J}^f$. Note that certain details of [11] are provided in [12].

2. Calculate λ^f for $f=1,...,F,$ using the perspective projection equations as

$$\lambda^f = \frac{1}{\sum_{n=1}^{N} \left\| \left(p^f{}_{n,x}, p^f{}_{n,y}\right) \right\|} \cdot \sum_{n=1}^{N} \left(\left\| \left(c^f{}_{0,x} + S_n \bullet \hat{I}^f, c^f{}_{0,y} + S_n \bullet \hat{J}^f\right) \right\| - \left\| \left(p^f{}_{n,x}, p^f{}_{n,y}\right) \right\| ES_n \bullet \hat{K}^f \right)$$

3. Modify the 2-D measurements $(p^f{}_{n,x}, p^f{}_{n,y})$ for $n=1,...,N,$ and for $f=1,...,F,$ using the calculated values in Steps 1 and 2 as

$$p^f{}_{n,x} \leftarrow p^f{}_{n,x}(\lambda^f + ES_n \bullet \hat{K}^f), \text{ and}$$
$$p^f{}_{n,y} \leftarrow p^f{}_{n,y}(\lambda^f + ES_n \bullet \hat{K}^f).$$

4. Repeat Steps 1, 2, and 3 until a predetermined number of iterations has been reached, or the following average measurement of matching error

$$\varepsilon = \left(\frac{1}{V} \sum_{f=1}^{F} \sum_{n=1}^{N} \left\| \left(p^{f}{}_{n,x} - \frac{c^{f}{}_{0,x} + S_n \bullet \hat{I}^f}{\lambda^f + ES_n \bullet \hat{K}^f}, \; p^{f}{}_{n,y} - \frac{c^{f}{}_{0,y} + S_n \bullet \hat{J}^f}{\lambda^f + ES_n \bullet \hat{K}^f} \right) \right\|^2 \right)^{\frac{1}{2}}$$

goes below a predetermined threshold, where the summation is only over the visible points in each image, the quantity V denotes the total number of visible points in all images, and $(p^{f}{}_{n,x}, p^{f}{}_{n,y})$ are the original 2-D measurements. In a typical implementation, the number of iterations is selected to be 50 and the threshold is selected to be 1 pixel.

2.2 Global Face Adaptation

The 3-D positions S_n, $n = 1,...,N,$ of the feature points of the person's face obtained in the previous section are used to globally deform the initial geometry mesh to match the relative positions of the feature points on the globally deformed geometry mesh and to match the global proportions of the person's face.

Let r_1 and r_2 denote the 3-D positions of the right-ear-top and left-ear-top, respectively; f_1 and f_2 denote the 3-D positions of the right-eyebrow-central and left-eyebrow-central, respectively; and b denote the 3-D position of the lip-center. Then, define the following three vectors

$$u = r_2 - r_1, \quad v = \frac{1}{2}(f_1 + f_2) - b, \quad \text{and} \quad w = \frac{1}{2}(f_1 + f_2) - \frac{1}{2}(r_1 + r_2)$$

.

Similarly, let R_1, R_2, F_1, F_2 and B denote the 3-D positions of corresponding 3-D geometry mesh points. Then, define the following three vectors

$$U = R_2 - R_1, \quad V = \frac{1}{2}(F_1 + F_2) - B, \quad \text{and}$$

$$W = \frac{1}{2}(F_1 + F_2) - \frac{1}{2}(R_1 + R_2).$$

Then, first, rotate the initial geometry mesh so that the vector V is aligned with the vector v. Next, scale the rotated geometry mesh in the x-, y-, and z- directions by the following scale factors, respectively

$$\frac{\|u\|}{\|U\|}, \quad \frac{\|v\|}{\|V\|}, \quad \frac{\left(\|w\|^2 - \|v\|^2\right)^{\frac{1}{2}}}{\|W\|}.$$

Finally, translate the scaled and rotated geometry mesh so that the point $V + B$ coincides with the point $v + b$.

2.3 Attachment to a Control Mesh

Following the global adjustments made to the face geometry mesh, each and every node of the geometry mesh is attached to, and hence controlled by, a triangle of a lower-resolution control mesh. The following procedure is used to attach the nodes of the geometry mesh to the triangles of the control mesh:

1. Calculate the normal vectors at the nodes of the control mesh: The normal vector n_A at the node A is obtained by averaging the surface normals of all the triangles of the control mesh that have the node A as one of their corners. The result of the averaging is normalized so that the vector n_A has unit length.

2. Define a normal vector for every point on the triangles of the control mesh: the normal vector n_P at the point P on the control triangle is obtained by a weighted average of the normal vectors at the corners of the control triangle as follows:

$$n_p = \alpha n_A + \beta n_B + \gamma n_C$$

where the weights α, β and γ satisfy $0 \le \alpha, \beta, \gamma \le 1$ and are uniquely determined by solving the equation

$$P = \alpha A + \beta B + \gamma C$$

under the constraint

$$\alpha + \beta + \gamma = 1.$$

3. For each node of the geometry mesh, find a control triangle to attach the node: A node of the geometry triangle is attached to a triangle of the control mesh only if there is a point on the triangle such that the line passing through the point and the node is parallel to the surface normal vector at the point. Tthe node of the geometry mesh located at Q is attached to the triangle of the control mesh because the line passing through Q and P is parallel to n_P. Then, it is said that the node Q is attached to the triangle ABC at the point P. This attachment is quantified by four numbers, namely the weights α, β and γ, and the distance d between the node Q and the attachment point P.

2.4 Local Face Adaptation and Face Animation

Once the geometry mesh is attached to the control mesh, local modifications to the geometry mesh are automatically made by moving the nodes of the control mesh as follows. When a node of the control triangle is moved from position *A* to position *A'*, the point of attachment is moved automatically to a new position *P'* calculated as

$$P' = \alpha A' + \beta B + \gamma C.$$

As the definition of the control triangle is changed from *ABC* to *A'BC*, the normal vectors at the corners *A', B* and *C* are re-calculated to be n'_A, n'_B and n'_C which are then used in the following equation to yield the surface normal vector n'_P at the point *P'* of attachment:

$$n'_p = \alpha n'_A + \beta n'_B + \gamma n'_C.$$

Finally, the moved position Q' for the node of the geometry mesh is calculated as

$$Q' = P' + n'_p d.$$

In order to represent the face model in mathematical terms, let *K* denote the number of nodes of the control mesh and D_n, $n = 1,..., K$, denote the positions of the nodes of the control mesh for the neutral face. Also, let α_k, β_k, γ_k, d_k, and m_k denote the attachment coefficients for the nodes of the geometry mesh for $k = 1,..., L$, where *L* denotes the number of nodes of the geometry mesh and m_k denotes the triangle of the control mesh controlling node *k* of the geometry mesh. The animation of the geometry mesh is also realized by moving the nodes of the control mesh, where the locations of the animated vertices of the geometry mesh are calculated using the above equations.

2.5 Creating the Texture Image

The texture image is obtained from the front, right, left, forehead, and chin images. The triangles of the geometry mesh are divided into 5 disjoint regions, so that the triangles that are in the same texture region acquire their texture data from the same image. The triangles that are on the boundary of a neighboring texture region get a blended texture from two images for a smooth color transition along the region border. In particular, the color inside the front-projected-front-boundary triangle is blended with the color inside the right-projected-front-boundary triangle, and the color inside the right-projected-right-boundary triangle is blended with the color inside the front-projected-right-boundary triangle. The proposed region-pair blending approach has similarities to the view-pair integration method proposed in [13].

3. RESULTS AND CONCLUSIONS

The modeling tool user interface and a sample personalized 3-D face model obtained from 5 photographs using the proposed 3-D face generation method are shown in Fig. 1. Sample animations of the same face are given in Fig. 2. The results are very realistic thanks to the following novelties introduced in the proposed modeling and animation methods: (1) an iterative algorithm to solve the 3-D reconstruction problem under perspective projection, (2) improved 3-D geometry mesh and texture image generation by using forehead and chin photographs in addition to front, right, and left photographs, (3) a 3-D color blending method that avoids the problem of creating a single 2-D sprite for texture image, and (4) attachment of geometry mesh to a lower resolution control mesh and animation of the geometry mesh via the control mesh and actions.

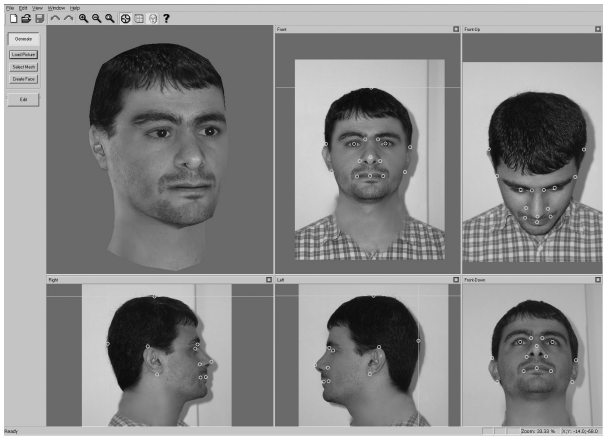

Figure 1. Five neutral photographs with feature points marked on them, and the generated 3-D face model (upper-left).

Figure 2. Sample face animations: Joy (left) and sadness (right).

REFERENCES

[1] Lavagetto, Fabio and Pockaj, Roberto 1999 “The Facial Animation Engine: Toward A High-Level Interface for the Design of MPEG-4 Compliant Animated Faces” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 2, pp. 277-289, March 1999.

[2] Escher, M. and Thalmann, N. M. 1997 "Automatic 3-D Cloning and Real-Time Animation of a Human Face" *Proceedings of Computer Animation*, Geneva, Switzerland, 1997.

[3] Sun, W., Hilton, A., Smith, R. and Illingworth, J. 1999 “Building Layered Animation Models from Captured Data” *Eurographics Workshop on Computer Animation*, pp. 145-154, September 1999.

[4] Hilton, A. and Inningworth J. 2000 “Geometric Fusion for a Hand-held 3-D Sensor” *Machine Vision Applications*, vol. 12, no. 1, pp. 44-51, 2000.

[5] Parke, F. I. And Waters, K. 1996 "Computer Facial Animation” *A.K. Peters, Wellesley*, 1996.

[6] Akimoto, T., Suenaga, Y. and Wallace, R.S. 1993 "Automatic Creation of 3-D Facial Models” *IEEE Computer Graphics & App.*, pp. 16-22, Sep. 1993.

[7] Waters, Keith 1987 “A Muscle Model for Animating Three-Dimensional Facial Expressions” *Computer Graphics (SIGGRAPH ‘87)*, vol. 21, no. 4, pp. 17-24, July 1987.

[8] Platt, Stephen M. and Badler, Norman I. 1981 "Animating Facial Expressions" *Computer Graphics (SIGGRAPH ‘81)*, no. 15, pp. 245-252, 1981.

[9] Lee et al. 1995 “Realistic Modeling for Facial Animation” *Computer Graphics (SIGGRAPH ‘95)*, pp. 55-63, Aug. 1995.

[10] Tekalp, A. Murat and Ostermann, Jorn 2000 “Face and 2-D Mesh Animation in MPEG-4” *Signal Processing: Image Comm.*, vol. 15, no. 4-5, pp. 387-421, 2000.

[11] Tomasi, Carlo and Kanade, Takeo 1992 “Shape and Motion from Image Streams under Orthography: A Factorization Method” *International Journal of Computer Vision*, vol. 9, no. 2, pp. 137-154, 1992.

[12] Morita, Toshihiko and Kanade, Takeo 1997 “A Sequential Factorization Method for Recovering Shape and Motion from Image Streams” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 8, pp. 858-867, Aug. 1997.

[13] Dorai, Chitra, Wang, Gang, Jain, Anil K. and Mercer, Carolyn 1998 “Registration and Integration of Mutliple Object Views for 3-D Model Construction” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, Jan. 1998.