Cloning of MPEG-4 Face Models

Marc Vila Mani
marc.vila3@upcnet.es

Joern Ostermann
ostermann@research.att.com

AT&T Labs – Image Processing Software and Technology
Middletown,NJ - USA

Abstract-
In this paper, we present a method to clone MPEG-4 facial animation tables (FAT) from a source face model to an arbitrary target face models. FATs describe the deformation of a face model in order to create different facial expressions. Using correspondences between source and target model like MPEG-4 facial feature points, we compute an interpolation function based on Radial Basis Functions or B-splines with weights. Using the interpolation function, we clone the motion from one model to the other. Cloned FATs preserve the relative motions and character of the original facial animations. B-splines with weights and a surface distance measure produce smoother animations than RBF.

Keywords: Facial animation, Face Animation Tables, RBF, B-splines, surface distance, MPEG-4, Floyd.

I. Introduction

In recent years, talking face models became of interest to many researchers, because face models and face animation can be useful in applications like Electronic Commerce or entertainment. Face animation is also defined in MPEG-4. This object-based multimedia compression standard allows for encoding of different audio-visual object. The visual objects may have natural or synthetic content. Special synthetic objects such as human faces and bodies, as well as generic 2-D/3-D objects are composed of primitives like rectangles, spheres, or indexed face sets, which define an object surface by means of vertices and surface patches. The representation of synthetic visual objects in MPEG-4 is based on the VRML standard [4][3] using nodes such as *Transform*, which defines rotation, scale or translation of an object, and *IndexedFaceSet* describing 3-D shape of an object by an indexed face set.

MPEG-4 specifies a face model in its neutral state, a number of feature points on this neutral face as reference points, and a set of FAPs (facial animation parameters), each corresponding to a particular facial action deforming a face model from its neutral state. Deforming a neutral face model according to some specified FAP values at each time instant generates a facial animation sequence.

A Face Animation Table (FAT) defines how a model is spatially deformed as a function of the amplitude of an FAP. Using FATs with a face model, a generic render can create facial expressions like joy, sadness or surprise in response to FAPs.

To achieve the goal that talking heads become commonly used, a simple design procedure is required. The purpose of this paper is to present a fast method for generating FATs for a given new face model using a cloning process. We assume to have one face model and its FATs.
We will call the model from which we clone the FATs *source* model. The *target* model is the new face model for which we want to create new FATs.

One of the first steps to clone FAT tables requires establishing a correspondence between the vertices of the source and target models models. This requires the computation of an interpolation function. We compared the RBF interpolation methods of [1][2][8]. In this paper, we propose to use *B-splines with weights* as interpolation functions. These functions give us superior control over the smoothness of the interpolation. Furthermore, we propose to compute distances using surface distances (Floyd's algorithm) instead of Euclidian distances. This will avoid special treatment of eye and mouth contact lines. [2].
Another important differences from other proposals is that we neither need to define regions of influence for each FAP [1] nor do a direct mapping of facial expression or deformations of the source model onto the target model [2]. We are cloning FAT tables, which means that we don't need to animate the source model before cloning FATs for the target model.
In Section II, we review the interpolation methods. In Section III, we provide a brief description of FATs. Section IV explains the FAT cloning process. We present experimental results in Section V.

II. Interpolation Methods

In order to define the correspondences between the target and the source model, we consider two

networks of vertices in R^3 space, the source network $\{S_i = (p_i, q_i, r_i), i = 1..N\}$ (from which we will clone the FATs), and the target network $\{T_i = (x_i, y_i, z_i), i = 1..M\}$.

Only identifying some correspondences between the target and the source we want to find all the correspondences between the two networks. Then the interpolation problem is defining a function $F : R^3 \rightarrow R^3$ to satisfy:

$$F(T_i) = S_i, \qquad i = 1..M \qquad (1)$$

We investigated two interpolation methods to find this function *F*: *Radial Basis Functions* and *B-splines with weights*.

2.1 Radial Basis Functions

RBF interpolation is one method of scattered data interpolation, which is widely used in surface reconstruction, image morphing, data modeling and so on. The simplicity of this interpolation method and the good results that have been provided by other researchers were the reasons to choose this method as a first approach [1][2][5][8].

The definition RBF interpolation follows [6]: suppose the set $Q \in R^3$ includes the all target vertices. T represents some vertices of the target network and S represents their correspondences in the source network

$$T = \{t_1, t_2, ...t_N\} \subset Q, S = \{s_1, s_2, ...s_N\} \in R^3$$

RBF interpolation is defined as:

$$F(t) := \sum_{i=1}^{N} w_i \phi(t - t_i) \qquad (2)$$

where the base function $\phi(r) : R^3 \rightarrow R$ is positive definite, r in $\phi(r)$ represents Euclidean norm ($r = \|t - t_i\|^2$) and $w_i \in R^3$ is the weight of the *i*th base function. For the polynomial precision, we add to (2) the term:

$$F(t) := \sum_{i=1}^{N} w_i \phi(t - t_i) + \sum_{l=1}^{M} c_l p_l(t) \qquad (3)$$

where M = dim P_m^3, $c_1, ...c_M \in R$ are control coefficients, $p_1, ..., p_M$ is a basis of P_m^3 and P_m^3 is the space of three-variate polynomials of order at most m.

In this paper, we use *compactly supported positive definite radial functions* (RBFCS) [6], which are developed from

$$f_l(r) = (1 - r^2)^l_+, \qquad r \in R, \ l \geq 0, \quad (4)$$

For our purposes we used the following RBFCS :

$$\phi_{2,1} = (1 - r)^4 + (4 + 16r + 12r^2 + 3r^3) \quad (5)$$

2.2 B-splines with weights

Another way to solve the interpolation problem is using *B-splines with weights* (C^2 Continuous splines with weights) [7]. B-splines are defined in intervals $(t_0,s_0),\ldots,(t_N,s_N)$ and they are functions of cubic polynomials. Their basic property is that they minimize the seminorm of (1):

$$\int_{t_o}^{t_N} (F''(t))^2 dt \qquad (6)$$

A small $F''(t)$ is desirable in flat regions whereas a big $F''(t)$ is desirable in regions with big slopes. If we add the weight function w(t) to (6):

$$\int_{t_o}^{t_N} w(t)(F''(t))^2 dt \qquad (7)$$

then, the bigger weight within an interval, the better approximation of the segment joining the two vertices. This is what is known as B-slines with weights.

For our purposes we fixed w(t) to a constant in each interval, getting then a C^1 continuous spline with weights. $F(t)$ is defined as:

$$F(t) = \sum_{i=0}^{N} s_i p_i(t) + \sum_{i=0}^{N} a_i q_i(t) \qquad (8)$$

where $p_i(t)$ and $q_i(t)$ are cubic polynomials defined for each interval.

For a given set of correspondences (t_i,s_i), weights $\{w_0,\ldots,w_{N-1}\}$ and correspondences $(t_0,s_0),\ldots,(t_N,s_N)$ we can find the coefficients a_i and then the interpolating function $F(t)$.

III. MPEG-4 FAT

A FAT defines the spatial change of a model as a function of the amplitude of a FAP.

3.1 Face Animation Tables

FATs may define the animation of *transform nodes* which define rotation, scale or translation of an object. FATs may also describe the motion trajectory of one or more vertices of indexed face sets using displacement vectors. The trajectory is either linear or piecewise linear.

Assuming a linear motion trajectory, a vertex $\boldsymbol{P}_m$ of a face model in neutral state (FAP = 0), a 3D displacement vector $\boldsymbol{D_{m1}}$ and a given FAP amplitude, $\boldsymbol{P}_m$ moves to its new position $\boldsymbol{P'}_m$:

$$\boldsymbol{P'}_m = \boldsymbol{P}_m + \text{FAP} * \boldsymbol{D_{m1}}, \qquad (9)$$

3.2 Creation of FATs without Cloning

Using animation software, the procedure for creating FATs requires 2 steps for each FAP: First, the deformation of the face from its neutral state to its final state; second the computation of the

displacement vectors for each vertex that describes the transition from the neutrals state to the final state. Using this process, we created our source model [9].
Using the cloning process described in the next section, we can create a FAT for each FAP of a new model without the need to go through the process of deforming the face model in an animation software.

IV. Cloning of FAT

The FATs of a source model define the motion of vertices for a FAP and the related feature point. In the target model, we need to identify the feature point as well as the vertices corresponding to the vertices in the FAT of the source model. After the corresponding vertices are identified using a distance function, we need to compute the displacement vectors for these vertices.

4.1 Cloning Process

The cloning process can be explained in 7 steps Figure 1 shows these steps applied to two generic networks of vertices affected by one FAT:

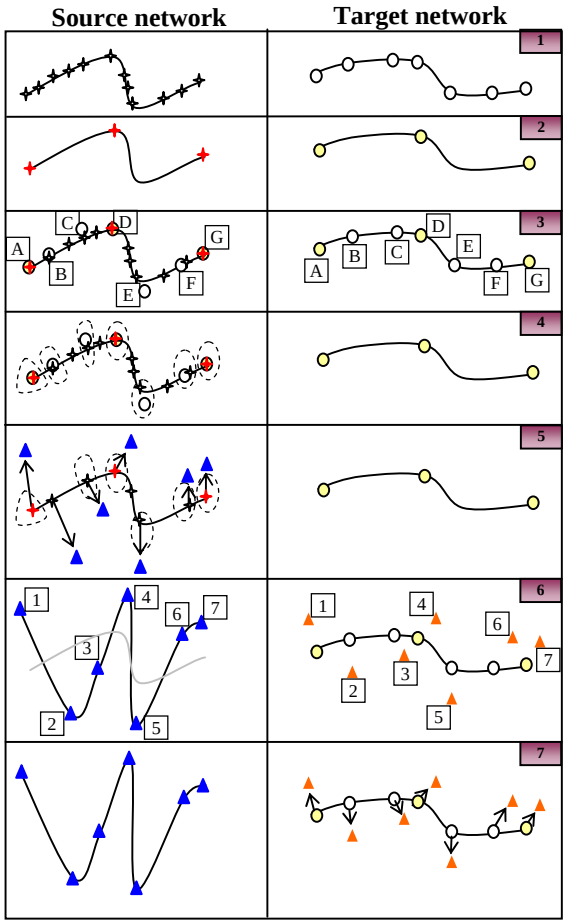

Figure 1: Representation of the cloning process with two networks of points.
Left column=source network , Right column=target network

1-The balls show the distribution of points in the target network and the stars show the distribution in the source network.
2-Select correspondences (three in this example) between the two networks. These corresponding points are used to compute the interpolation function that allows us to interpolate vertex positions from the target network to the source network (Section II).
3-Compute the position of each vertex of the target network (A,B,C,D,E,F and G in the example) in the source network using the interpolation function of step 2.
4-Using a distance function, find the closest vertex in the source network for each interpolated vertex of the target network and establish a correspondence between them.
5-Apply FATs to the vertices of the source network that have a correspondence assigned. This gives new vertex positions of the source network.
6-Using the inverse of the interpolation function defined in step 2, find the new displaced source points in the target network (1,2,3,4,5,6 and 7 in the example).
7-Finally, compute the new displacement vectors as the difference between a vertex of the target network and the position of the mapped vertex of the source network. The vectors define the FAT for the target network.

4.2 Distance Computation

In [1][2] the Euclidian distance metric is used to determine neighboring vertices. However, this metric assumes that vertices across the contact line between upper and lower lip are close. These vertices often move in opposite directions. A manual process is used to identify these points.
A clear distinction between points across contact lines can be made if we use a surface distance for defining neighboring vertices. Due to the mouth opening, the surface distance will not define points across the contact line as close (Fig. 2). Similarly, FATs for eye areas of the target model can be computed automatically if using a surface distance.
For this reasons, we changed the Euclidean distance for the distance between points computed along the surface. We computed this distance by using Floyd's algorithm.
In this way the points belonging to the upper lip and to the lower lip are always far apart also in the lip contact line, thus avoiding possible mistakes.
Figure 2-4 show some examples helping to

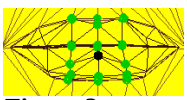

Figure 2: Euclidean distances. Maximum distance from the black point=50.

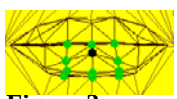

Figure 3: Surface distances. Maximum distance from the black point=50.

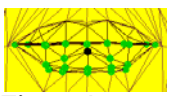

Figure 4: Surface distances. Maximum distance from the black point=100.

understand the difference between the two methods. Figure 2 shows one point (black) and the surrounding points within a maximum distance using the Euclidean distance metric to compute the distance between points. Figure 3 and 4 do the same but using the surface distance.

V. Experimental Results

We experimented cloning FATs related to facial expressions. Figures 5 shows some results of the cloning process for the smile, surprise and disgust expressions. The first model is the source model from which we clone FATs.

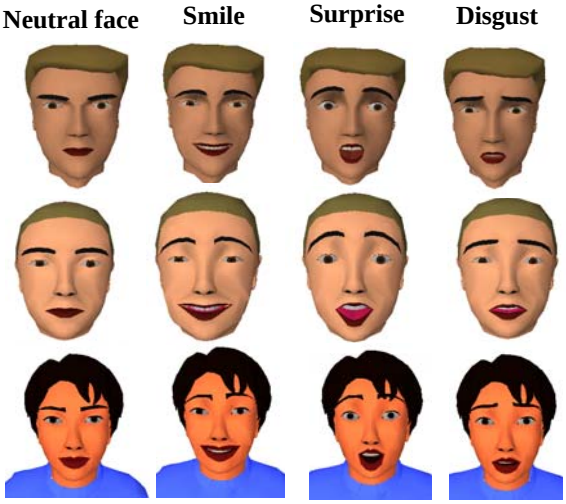

Figure 5: Top row: Source model, lower rows: target models with cloned facial expressions

We also experimented with the two interpolating methods purposed in this paper. Figure 6 shows an interpolation example using RBF and B-splines with weights.

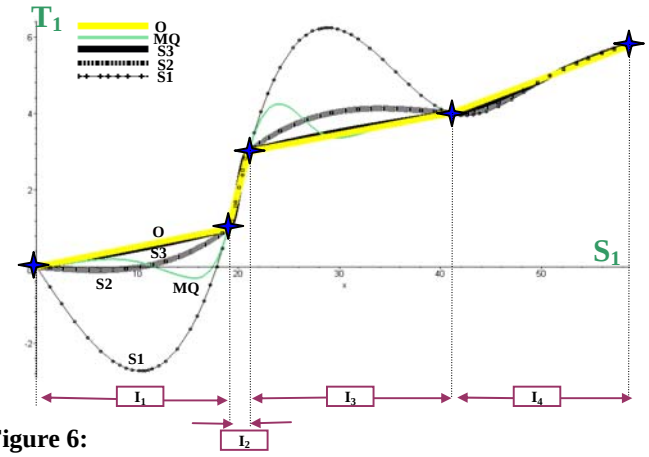

Figure 6:
O – Exact equivalences between the two models
MQ – RBF multi-quadratics interpolation
S1 – B-splines interpolation with weights 1 in interval I_1 and I_3
S2 – B-splines interpolation with weights 100 in interval I_1 and I_3
S3 – B-splines interpolation with weights 1000 in interval $I_1$and I_3

S_1 axis represents one coordinate space in the source model, and T_1 axis the same coordinate space in the target model. Crosses represent the correspondences between the two models.
The yellow line ("O" line) shows the "*ideal*" correspondences between the two models. RBFCS and B-splines with all weights fixed to 1 give us the worst approximation ("S1" line). RBF multi-quadratics, used in [1][2] give us "MQ" line.
B-splines allow us to vary the weights. Choosing appropriate weights enables us to match the curvature of the target network with high precision (lines S1, S2, S3). We propose to use the curvature of the target model for defining the weights. This technique enables us to create more predictable animations than using the other interpolation methods.

V. Conclusions

Cloning FATs is a powerful and efficient tool to transfer motion and face animation tables from one model to another. Using B-splines with weigths as the interpolation function allows us to adapt the motion to the specific surface of the target model. Furthermore, the use of a surface distance function enables us to automatically clone face models, provided the MPEG-4 feature points are known.

VI. References

[1] F.Lavagetto and R. Pockaj, "The facial animation engine: Toward a high-level interface for the design of MPEG-4 compliant animated faces", IEEE CSVT vol. 9, no. 2, pp. 277-289, 1999.
[2] Jun-yong Noh and Ulrich Neumann, "Expression Cloning", ACM SIGGRAPH 2001.
[3] ISO/IEC 14772-1: 1997, Information Technology – Computer graphics and image processing – The Virtual Reality Modeling Language – Part 1: Functional specification and UTF-8 encoding.
[4] ISO/IEC IS 14496-1 Systems, 1999.
[5] Li Mengdong and Ruan Qiuqi, "Facial Wire-Frame Model Adaptation Using RBF Interpolation", ICSP 2000.
[6] Schaback, R., Creating surfaces from scattered data using Radial Basis Functions, *Mathematical Methods for curves and surfaces*, Vanderbilt University Press, 1995, pp. 477-496.
[7] Springer-Verlag, 1978, *A practical Guide to splines*.
[8] Ana C.Valle and Joern Ostermann, "3D Talking Head Customization by Adapting a Generic Model to One Uncalibrated Picture", ISCAS-2001.
[9] Ostermann, J. and Haratsch, E., "An animation definition interface: Rapid design of MPEG-4 compliant animated faces and bodies," Int'l. Workshop on Synthetic-Natural Hybrid Coding and Three Dimensional Imaging, Rhodes, Greece, 5-9 September 1997, pp. 216-219.

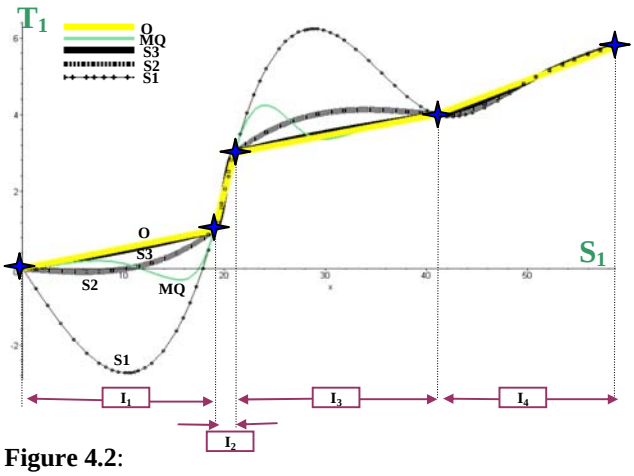

Figure 4.2:
O – Exact equivalences between the two models
MQ – RBF multi-quadratics interpolation
S1 – B-splines interpolation with weights 1 in interval I_1 and I_3
S2 – B-splines interpolation with weights 100 in interval I_1 and I_3
S3 – B-splines interpolation with weights 1000 in interval $I_1$and I_3