Fast Facial Motion Cloning in MPEG-4

Marco Fratarcangeli and Marco Schaerf
Department of Computer and Systems Science
University of Rome ”La Sapienza”
frat,schaerf@dis.uniroma1.it

Abstract

Facial Motion Cloning (FMC) is the technique employed to transfer the motion of a virtual face (namely, the source*) to a mesh representing another face (the* target*), generally having a different geometry and topology. We present a novel FMC method relying on the combination of the Radial Basis Functions (RBF) based scattered data interpolation with the encoding capabilities of the MPEG-4 Facial and Body Animation (FBA) international standard. Starting from a manually picked set of correspondences, we find a scattered data interpolation function that precisely fit the source face mesh into the target one. Then, all the MPEG-4 FBA basic movements (encoded as morph targets) of the source face are deformed using the same function and the deformations are mapped to the corresponding vertices of the target mesh. By doing this, we obtain, in a straightforward and simple way, the animatable MPEG-4 compliant version of the target face in a short amount of time.*

1. Introduction

In this paper, we propose a Facial Motion Cloning (FMC) method used to transfer pre-existing facial motions from one face to another. In our method, a facial motion is represented as a set of morph targets encoded by the MPEG-4 Facial and Body Animation (FBA) standard. A morph target (MT) is a variation of the face that has the same mesh topology, but different vertex positions. Essentially, it is the face mesh performing a particular key movement. Each MT corresponds to a basic facial action encoded by the proper MPEG-4 FBA parameter. Blending together the MTs, it is possible to represent a wide range of face movements. We address the problem of automatically cloning the whole set of MTs, defined by MPEG-4 FBA, from a generic source to a generic target face model.

The basic steps of our FMC method are schematically represented by Figure 1 and can be summarized as follows:

1. - 2. given a manually picked set of feature points on the input face meshes, we compute a scattered data interpolation function $f(p)$ employed to precisely fit the shape of the source into the shape of the target mesh through a volume morphing;

3. we map each vertex of the target face into the corresponding triangular facet of the deformed source mesh in its neutral state through a proper projection. The target vertex position is thus expressed in a function of the vertex positions of the source triangular facet through barycentric coordinates;

4. - 5. we deform all the source MTs by applying to each one of them the same morphing function $f(p)$ used to fit the neutral source into the neutral target mesh. Hence, we can compute the new position of each target vertex considering the location of the corresponding deformed source facet, obtaining the target MT.

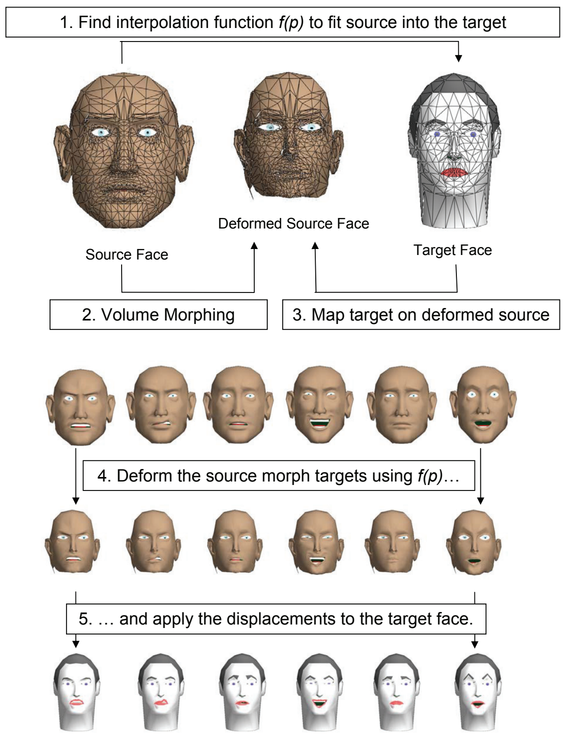
1. Find interpolation function *f(p)* to fit source into the target
Deformed Source Face
Source Face
Target Face
2. Volume Morphing
3. Map target on deformed source
4. Deform the source morph targets using *f(p)*…
5. … and apply the displacements to the target face.

Figure 1. Our FMC approach.

The remainder of this paper is organized as follows. In Section 2 we give some references to MPEG-4 FBA standard specification and review related work. In Section 3 we detail how to find the fitting function $f(p)$ and in Section

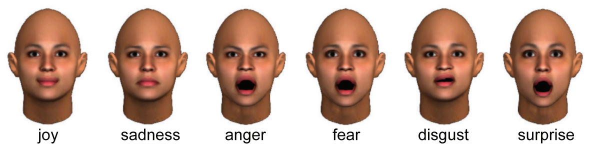

Figure 2. Expression morph targets.

4 we describe the cloning process. Section 5 presents our experimental results and in Section 6 we conclude by discussing results and future work.

2. Background

2.1. MPEG-4 Facial and Body Animation

As background knowledge, we assume the reader familiar with the basic notion of MPEG-4 FBA. Beside the standard itself [1], there are other excellent references [17, 14] covering this subject.

The MPEG-4 specification defines 68 low-level Facial Animation Parameters (LL-FAPs) and two high-level (HL) FAPs. LL-FAPs represent a complete set of basic facial actions, and therefore allow the representation of most natural facial expressions. All LL-FAPs are expressed in terms of the Facial Animation Parameter Units (FAPU). They correspond to distances between key facial features and are defined in terms of distances between the MPEG-4 facial Feature Points (FPs). The FP set is a sparse data set giving a spatial reference for the geometry of the face model. FPs are arranged in groups, each one defining a particular zone of the face (right eye, nose, upper teeth, etc.). For example, FAP no. 3, `open_jaw`, moves the Feature Point 2.1 (bottom of the chin) downwards and it is expressed in MNS (mouth-nose separation) units. This latter is defined as the distance between the nose and the mouth divided by 1024. The specification includes two HL-FAPs: `expression` (containing two out of six basic expressions, Figure 2) and `viseme` (containing two out of a predefined list of 14 visemes). The specification also defines the Facial Animation Tables (FATs). The FATs allow to specify, for each FAP, the exact motion of the vertices and/or transforms in the 3D model. This means that the expressions, visemes and LL-FAPs are described in a way that is essentially equivalent to the mentioned morph-targets. Animation systems then interpolate and blend between the values from the FAT.

There are two main advantages in using morph targets encoded by MPEG-4 FBA. The first one is in the capability of the MPEG-4 FBA to express and precisely control a wide range of facial expression [17]. Computing each morph target of the target face, we are sure to be able to perform generic face animation encoded in a MPEG-4 FBA stream in a realistic manner by a proper blending. The second one lies in the possibility to use other already-existing application relying on the same standard. For example, to animate the produced talking heads, we employed a lightweight and portable MPEG-4 FBA player [2] and the correspondent plug-in available for 3DS Studio MAX^{TM}.

2.2. Related Work

Facial Motion Cloning is a problem of high interest in the 3D content production phase, since it affects a very time consuming task. Escher *et al.* [6, 7], developed a cloning method based on Dirichlet Free Form Deformation (FFD) in which the deformation is controlled by a few external points. To achieve volume morphing between the source and the target face meshes, the control points are usually difficult to define and not very intuitive for manipulation. Mani *et al.* [12] use B-splines with weights to clone MPEG-4 FATs from a source face model to a target face model. Manual selection of corresponding points between source and target faces is required, additional manual adjustment of B-spline weights is also required to increase the correctness of the mapping even if this latter is not guaranteed.

In Expression Cloning developed by Noh *et al.* [13], the source mesh is morphed, through the use of Radial Basis Functions (RBF)-based scattered data interpolation to match the shape of the target mesh. Then, the source motion vectors are sheared according to the *local* shape of the deformed source facets through proper affine transformations and applied to the correspondent target vertices. In Pandzic's approach [16], movements are encoded in MPEG-4 FBA morph targets. Facial cloning is obtained by computing the normalized difference of 3D vertex positions between the source morph targets and the neutral source face. The key positions represented by morph targets are expressed by the MPEG-4 Facial Animation Parameters (FAPs). Each morph target corresponds to a particular value of one FAP. The facial motion is then normalized and added to the vertex positions of the target face, resulting into the animatable target face.

The core idea of our method is the following. Instead of re-targeting the facial motion locally like in Noh *et al.* [13], we *globally* deform the source MTs applying $f(p)$ and then relocate the position of the target vertices according to the new coordinates of the deformed source vertices. This allows for a good cloning quality in a rather short amount of time (see Section 5). The resulting target model is expressed by MPEG-4 FBA morph targets like in Pandzic [16].

3. Shape Fitting

The task of the shape fitting process is to adapt the generic source face mesh to fit the target face mesh. As input, we take the source and target face meshes. The source and the target faces are both available in neutral position as defined in the MPEG-4 FBA specification [17]. For each neutral face mesh, the corresponding MPEG-4 Feature Point (FP) set must be defined, that is, each FP is manually mapped onto a vertex of the input face meshes. Taking these FP sets as a reference, we are able to extract some face features, that is, the eye contour and the inner lip contour (Section 3.1). We put together the FPs with the eye and lip contours features obtaining a set of correspondence points between source and target. These points are used to find the scattered data interpolation function that roughly fits the source into the target mesh (Section 3.2). The precise fitting of the moving zone of the deformed source model is obtained by iteratively refining the position of the points already in the correspondence set and enriching this latter with further correspondence points, introduced to reduce the error (Section 3.3).

3.1. Eye and Lip Feature Extraction

Starting from the manually picked MPEG-4 FPs, we extract the eye and lip features from the input meshes through automatic algorithms.

Eye Inner Contour Extraction: Driven by some of the MPEG-4 FPs belonging to the eye group, we find a proper path in the mesh graph going from a start point to an end point, where the start and the end point are defined according to the round robin UP $\Rightarrow$ LEFT $\Rightarrow$ DOWN $\Rightarrow$ RIGHT $\Rightarrow$ UP. To find the path between the start and the end point, we use a greedy strategy:

```
Right Eye FPs:  3.2 ⇒ 3.8 ⇒ 3.4 ⇒ 3.12 ⇒ 3.2
Left Eye FPs:   3.1 ⇒ 3.7 ⇒ 3.3 ⇒ 3.11 ⇒ 3.1

  v = start_point;
  while (v ≠ end_point)
      for each neighbour n of v
          $n_n$ = (n - v) normalized;
          d = distance ((v + $n_n$), end_point);
      end for;
      v = n having smaller d;
      insert v in the eye contour set;
  end while
```

Lip Inner Contour Extraction: We find the inner lip contours by applying the following algorithm, once for the upper lip and once for the lower one:

```
  start_point = FP 2.5 (inner right corner lip);
  end_point = FP 2.4 (inner left corner lip);
  v_direction = end_point - start_point;
  v = start_point;
  insert v in the lip contour set;
  while (v ≠ end_point)
      for each neighbour n of v
          a = angle (v_direction, (n - v)) on $xy$-plane;
      end for;
      v = n having smaller [greatest] a, a∈ $(-\frac{\Pi}{2}, \frac{\Pi}{2})$;
      insert v in the upper [lower] lip contour set;
  end while
```

Note that a MPEG-4 FBA compliant synthetic face has the gaze and the nose tip towards the positive z-axis [17]. We do not provide a method to assess the correctness of such extraction algorithms, however they works in a satisfactory way with all our test face models.

3.2. Scattered Data Interpolation

Having computed the face feature points on both the source and the target face mesh in their neutral state, we construct a smooth interpolation function $f(p)$ that fits precisely the source model into the target face model.

Considering the domain of the function as the original 3D coordinates of the space where the input face models are, we find $f(p)$ fitted to the known data $u_i = f(p_i)$, from which we compute $u_j = f(p_j)$. For our purpose, Shepard-based methods and Radial Basis Function (RBF) methods [8], are easier to manipulate. In particular, RBFs are more effective when correspondence points are not distributed evenly and they are also computationally efficient. In our case, the known points p_i are located on the source mesh and the destination points u_i are the correspondent points on the target mesh. We use Hardy multiquadrics as the basis function. See [8] for a step-by-step process to obtain $f(p)$ from the set of correspondences $\{p_i, u_i\}$.

Basically, $f(p)$ can be computed once the correspondence set has been defined. The denser the correspondence set is the closer the resulting fit. We build a first guess of the interpolation point set $\{p_i, u_i\}$ by considering only a subset of FPs. To be precise, we choose to insert in the interpolation point set, all the FPs except group 10 (ears), 6 (tongue), FPs from 9.8 to 9.11 (teeth) and 11.5 (top of the head). We compute $f(p)$ and apply it to the source mesh obtaining a rough fitting. We then enrich the correspondence set by inserting the vertices belonging to the eye and lip contours of the source and the point lying on the nearest edge of the correspondent target contours. We recompute $f(p)$ with the enriched set of correspondences and apply it again on the source mesh in its neutral state obtaining again a rough fitting but this time with a correct alignment of the eye and lip contours.

3.3 Correspondence Set Refinement

After morphing the generic source face mesh to its new shape, we further improve the fitting by specifying additional correspondences. A vertex p is called a movable vertex, if its position in one of the morph targets is not equal to its position in the neutral face. Thus, such a vertex will potentially move during the animation. We compute the movable vertices for the deformed source mesh and then we project them on the target face surface by casting rays with a fixed length along the vertex normals and compute the intersection of the ray with the target mesh.We experimentally found a fixed length of $ENS0 * 0.3125$ for the casted rays, where ENS0 is the MPEG-4 FAPU defining the distance between the eye and nose.

By doing this, we are able to know for each movable vertex of the source face p_i, the corresponding intersection point u_i on the target surface mesh. Having chosen a fixed length, only a part of the movable vertices will have a correspondent point on the target surface. This is because, after the initial rough fit, only the nearest vertices will be close enough to the target mesh surface to permit a ray-facet intersection. We tried also to use rays with unlimited length. However, we achieved better results with the fixed ray length, probably because in this way the interpolated surface slowly stick to the target surface without inserting high-frequency elements.

We consider the first 125 moving vertices having greatest error $e_i = \|u_i - p_i\|$ and we insert the pair $< p_i, u_i >$ in the correspondence set. If a point p_i is already in the set, then only the position of the correspondent u_i is updated. We solve again the linear system to find $f(p)$ [8] for the enriched correspondence set and we re-run the scattered data interpolation algorithm to update the whole source face model from its neutral state. This process is iterated until no more movable vertices are inserted in the correspondence set. This may happen for two different reasons:

- all the movable vertices have already been inserted in the correspondence set;
- the actual set has enough correspondence pairs to fit carefully the source to the target mesh.

By applying the refined $f(p)$ to all the vertices of the neutral source face mesh, this latter fits precisely the target mesh in the area where there are vertices that will potentially move during the animation (Figure 3).

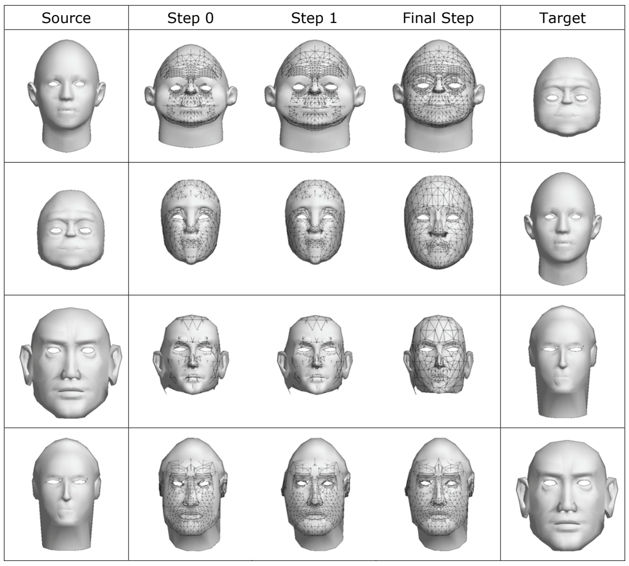

Figure 3. Source shape fitting iterative process. Step 0: rough fit. Step 1: the eye and the lip features are aligned. Final Step: after some iterative steps, the source fits to the target face in the moving zone.

4 Cloning Process

As input of the motion cloning process, we need the scattered data interpolation function $f(p)$ just refined and the morph targets, corresponding to the MPEG-4 FAPs, of the source face. To be precise, we need the 64 MTs corresponding to low-level FAPs and the 20 MTs corresponding to high-level FAPs (14 visemes and 6 emotions), for a total of 84 MTs. We map the target vertices on the deformed source mesh through the same projection used in Section 3.2, this time casting the fixed-length rays from the target vertices towards the deformed source mesh. At this stage of the process, the deformed source and the target meshes are very similar to each other and each target vertex can be considered as located where the casted ray intersects the proper triangular face of the source mesh. Thus, we can compute the barycentric coordinates of each target vertex considering the vertices of the correspondent source triangular facet. The position of the target vertices can be expressed as a linear combination of the positions of the three corresponding source vertices. Then, for each particular source MT S_i we compute the corresponding target MT T_i by applying the following algorithm:

```
$T_i$ = neutral target face (stored in $T_0$);
apply $f(p)$ to $S_i$ only in each vertex $p \in D_i$;
for each vertex $q \in T_i$
    if $(p_a \in D_i) \vee (p_b \in D_i) \vee (p_c \in D_i)$
        $q = p_a \cdot b + p_b \cdot c + p_c \cdot a$;
end for;
```

where D_i is the set of moving vertices of S_i, p_a, p_b and p_c are the vertices of the source triangular facet corresponding to the vertex q and a, b and c are the proper barycentric coordinates. Note that we apply the scattered data interpolation function $f(p)$ *only to the movable vertices* D_i of each particular source MT S_i in order to speed up the cloning process. That is, we apply the same global deformation function $f(p)$ used to fit the source into the target, to the local part of the source MTs S_i that will move when S_i is employed during the animation. Then, we compute the position of the corresponding target vertices by linear combination of the barycentric coordinates with the new positions of the source triangular facet obtaining, finally, the resulting target MT T_i.

We clone the global motion of the head as well as the movements of the tongue and teeth (if present) through affine transformations like in Pandzic [16].

5. Results

We performed our experiments on an AMD PC (AthlonXP 2,14 GHz processor and 512 MB RAM). The test models have different polygonal resolutions, shapes and topologies (both symmetric as well as asymmetric). A full set of high- and low-level FAP motions was available for each model. All motion was cloned from each model to each other model, producing the grids of cloned animated models for each motion (Figure 4). Therefore, looking at each row shows how an expression is cloned from one face to all the other faces; looking at each column shows how the expression is cloned onto the same face from different sources.

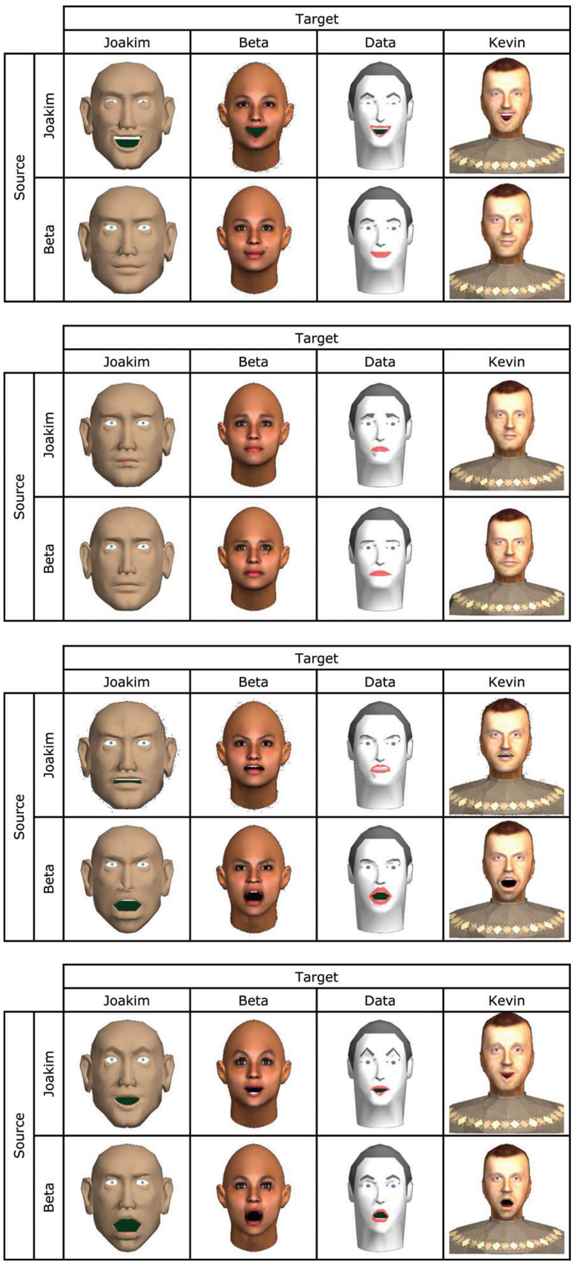

Figure 4. Cloning grids for Expression Joy, Sadness, Anger and Surprise.

To assess objectively the quality of our approach, we cloned the same source animatable faces to themselves and we computed the error between the source and the output animatable model as the average of the error of each of the 64 morph targets:

	source ⇒ source
`beta`	0.028 %
`data`	0.280 %
`joakim`	0.084 %
`kevin`	0.141 %

Table 1. Average error on cloned morph targets.

	MVs	**CPs**	**Is**	**RT**	**CT**	**TT**
`beta` ⇒ `data`	396	378	8	5.5 s	3.2 s	8.8 s
`data` ⇒ `beta`	224	224	3	0.7 s	1.0 s	1.7 s
`joakim` ⇒ `data`	479	461	8	8.6 s	1.4 s	10.0 s
`data` ⇒ `joakim`	224	224	4	1.1 s	1.0 s	2.1 s
`beta` ⇒ `kevin`	396	378	9	6.5 s	3.5 s	10.0 s
`kevin` ⇒ `beta`	294	290	5	10.8 s	3.2 s	14.0 s

Table 2. MVs: Moving Vertices. CPs: Correspondence points. Is: Iterations to refine. RT: $f(p)$ Refinement Time. CT: Cloning Time. TT: Total Time.

$$e = \left(\frac{1}{84} \sum_{j=1}^{84} \frac{\sum_{i=1}^{N_v} \|v_i^S - v_i^T\|}{\sum_{i=1}^{N_v} \|v_i^S\|} \right) \times 100 \qquad (1)$$

where N_v is the number of the vertices of the face mesh, v_i^S and v_i^T are, respectively, the i-th vertex of the source and the corresponding one on the cloned output model. Table 1 shows the obtained values.

Finally we present in Table 2 the computation time for some cloning processes performed during our tests.

6. Discussion and Conclusions

We proposed a method dealing with the problem to reuse existing facial motion to produce in a short amount of time a ready to be animated MPEG-4 FBA compliant talking head. Apart from an initial manual picking of some correspondence points all the techniques presented here are fully automatic. In terms of visual results shown, even though only a small subset of them could be presented here, we believe that most facial movements for expression and low-level FAPs are copied correctly to the target face models.

We did not address the topic of how to create the motion of the source model itself. Source models motion can be created manually by 3D artists (with the tools they are used to), or can be computed automatically, for example, by physically-based algorithms [9].

The main computational effort during the cloning process lies in the refinement process of the $f(p)$ interpolation function. This is because each pair of correspondence points corresponds to a linear equation in the system to resolve in order to obtain $f(p)$. The asymptotic behavior of the linear equation-solving algorithm (LU decomposition) is $O(n^3)$, where n is the number of moving vertices of the source face. Since the correspondences can be very close each other, we think that not all of them are necessary and, as a future work, a principal component analysis could be employed to remove the less significant ones.

After the cloning process is finished, the target face is ready to perform *generic* facial animation encoded into a MPEG-4 FBA stream. The computational cost of the animation depends from the player employed. In our case, we used the lightweight MPEG-4 FBA player provided by [2] in which the animation is achieved, in each frame, through linear interpolation between the proper morph targets and rendered in OpenGL. Thus, the computational cost is rather low. The FBA parameters, both FPs either FAPs, can be extracted automatically from visual, audio or motion capture systems [4, 3, 10, 11], therefore they can be properly compressed in a data stream. Combined with facial feature tracking, MPEG-4 FBA talking heads can potentially be used for a very low bitrate visual communication in a model-based coding scenario (teleconferencing, games, web interfaces) [5, 15].

References

[1] Moving Pictures Expert Group. MPEG-4 International standard. ISO/IEC 14496. http://www.cselt.it/mpeg.

[2] Visage Technologies AB. http://www.visagetechnologies.com, february 2005.

[3] J. Ahlberg. *Model-based Coding – Extraction, Coding and Evaluation of Face Model Parameters*. Phd thesis no. 761, Department of Electrical Engineering, Linköping University, Linköping, Sweden, September 2002.

[4] J. Ahlberg and F. Dornaika. ”Efficient Active Appearance Model for Real-Time Head and Facial Feature Tracking”. In *IEEE International Workshop on Analysis and Modelling of Faces and Gestures (AMFG)*, Nice, October 2003.

[5] T. Capin, E. Petajan, and J. Ostermann. ”Very Low Bitrate Coding of Virtual Human Animation in MPEG-4”. In *IEEE International Conference on Multimedia and Expo (II)*, pages 1107–1110, 2000.

[6] M. Escher and N. Magnenat-Thalmann. ”Automatic 3D Cloning and Real-Time Animation of a Human Face”. In *Computer Animation*, Geneva, Switzerland, June 1997.

[7] M. Escher, I. Pandzic, and N. Magnenat-Thalmann. ”Facial Deformations for MPEG-4”. In *Computer Animation 98*, pages 138–145, Philadelphia, USA, 1998. IEEE Computer Society Press.

[8] S. Fang, R. Raghavan, and J. Richtsmeier. ”Volume Morphing Methods for Landmark Based 3D Image Deformation”. In *SPIE International Symposium on Medical Imaging*, volume 2710, pages 404–415, Newport Beach, CA, February 1996.

[9] M. Fratarcangeli and M. Schaerf. ”Realistic Modeling of Animatable Faces in MPEG-4”. In *Computer Animation and Social Agents*, pages 285–297, Geneva, Switzerland, July 2004. Computer Graphics Society (CGS).

[10] T. Goto, M. Escher, C. Zanardi, and N. Magnenat-Thalmann. ”MPEG-4 Based Animation with Face Feature Tracking”. In *Computer Animation and Simulation ’99*, pages 89–98.

[11] W.-S. Lee, M. Escher, G. Sannier, and N. Magnenat-Thalmann. ”MPEG-4 Compatible Faces from Orthogonal Photos”. In *CA*, pages 186–194, 1999.

[12] M. Mani and J. Ostermann. ”Cloning of MPEG-4 face models”. In *International Workshop on Very Low Bit rate Video Coding (VLBV 01)*, Athens, October 2001.

[13] J. Noh and U. Neumann. ”Expression Cloning”. In *SIGGRAPH*, pages 277–288. ACM SIGGRAPH, 2001.

[14] J. Ostermann. ”Animation of Synthetic Faces in MPEG-4”. In *Computer Animation*, pages 49–51, Philadelphia, Pennsylvania, June 1998.

[15] I. Pandzic. ”Facial Animation Framework for the Web and Mobile Platforms”. In *Web3D Symposium*, Tempe, AZ, USA, 2002.

[16] I. Pandzic. ”Facial Motion Cloning”. *Graphical Models Journal, Elsevier*, 65(6):385–404, 2003.

[17] I. Pandzic and R. Forchheimer, editors. *”MPEG-4 Facial Animation – The Standard, Implementation and Applications”*. John Wiley & Sons, LTD, Linköping, Sweden, 1st edition, 2002.