S. Garchery, A. Egges, N. Magnenat-Thalmann. Fast Facial Animation Design for Emotional Virtual Humans. Measuring Behaviour, Wageningen, NL, CD-ROM Proceeding. September 2005.

# Fast Facial Animation Design for Emotional Virtual Humans

S. Garchery, A. Egges, N. Magnenat-Thalmann
*MIRALab, University of Geneva, Geneva, Switzerland*

## Abstract
Designing facial animation parameters according to a specific model can be time consuming. In this paper we present a fast approach to design facial animations based on minimal information (only feature points). All facial deformations are automatically computed from MPEG-4 feature points. We also present an extension of this approach that allows to personalize or to customize the deformations according to different characteristics. We will describe different prototypes of the facial animation system, available on different platforms. Then, we demonstrate how emotions and expression can be incorporated into the facial animation system.

## Keywords
Facial Animation, Parameterization, Personalization, Emotion, MPEG4

## 1 Introduction
A literature survey on different approaches on facial animation system reveals the following characteristics that should be found in an ideal facial animation system:

**Easy to use**: A facial animation system should be easy to use and simply to implement. This means:

- Be able to work with any kind of face model (male, female, children, cartoon like …)
- Require a minimum of time to set up a model for animation
- Allow for creative freedom of the animator to define specific deformations if necessary
- Get realistic results
- Be able to precisely control the animation

**Integration**: Using the system should be simple, fast and it should work in any kind of environment (PC, web, mobile…)
**Generality**: The possibility to reuse previous work (deformation data or animation) with a new model presents a big advantage and reduces the resources needed to develop new animations or applications.
**Visual quality**: finally, the result should look realistic with a cartoon like model or a cloned one. The quality should also be taken into account during the design process.

In order to achieve a maximum of these goals, it is crucial to properly define which parameters are used in the model. By proposing a parameterization system (FACS), Paul Ekman [5] started the definition of a standardization system to be used for facial synthesis. In 1999, MPEG-4 defined an interesting standard using facial animation parameters. This standard proposed to deform the face model directly by manipulating feature points of the face and presented a novel animation structure specifically optimized for e.g. networking applications. These parameters are completely model-independent, based on very few information and leave open the adaptation of animations for each face model according to the facial engine that is used. A lot of research has been done in order to develop facial animation engines based on this parameterization system. Commonly, a piece-wise linear interpolation function for each animation parameter [9,15] is used to produce the desired result (see section 2.1). This approach is easy to deploy but not easy to implement. It is useful for animating a face model, but it requires the preparation of deformation data. This data could be designed manually but in order to obtain good results a huge effort is required. Some research [13,16] has been done in order to simplify this work or to propose semi-automatic approaches.

In this paper we will briefly describe some aspects of the MPEG-4 standard for facial animation and present how it is applied in our facial deformation engine. Additionally, we will show a method to easily combine a quick preparation of a 3D model, together with a personalization of deformation. We will describe how different automatic and semi-automatic approaches can be applied to a talking head system, including a means to automatically personalize the facial animation in real-time. We will present different models of emotion that are used to control the expressivity of the face. Finally, we will present some results and conclusions.

## 2 Fast Facial Deformation Design

### 2.1 MPEG-4 overview and description
In order to understand facial animation based on MPEG-4 parameters system, we should describe some keywords of the standard and the pipeline in order to animate compliant face models.

- FAPU (Facial Animation Parameters Units): all animation parameters are described in FAPU units. This unit is based on face model proportions and computed based on a few key points of the face (like eye distance or mouth size).
- FDP (Facial Definition Parameters): this acronym describes a set of 88 feature points of the face model. FAPU and facial animation parameters are based on these feature points. These points could be also used in order to morph a face model according to specific characteristics.
- FAP (Facial Animation Parameters): it is a set of values decomposed in high level and low level parameters that represent the displacement of some features points (FDP) according to a specific direction. Two special values (FAP 1 and 2) are used to represent visemes and expressions. All 66 low level FAP values are used to represent the displacement of some FDPs according to a specific direction (see ***Figure 1***). The combination of all deformations resulting from these displacements forms the final expression. A facial animation then is a variation of these expressions over time.

Another aspect of MPEG-4 for facial animation, like Facial Interpolation Tables, could be applied to simplify the quantity of data needed to represent an expression or animation. With this approach, an animation can be represented by a small set of parameters, which is an efficient approach for network applications (less than 2 Kb for each frame).

The FAP stream does not provide any information on the displacement of neighboring vertices; therefore, for each FAP, we use a method that defines each displacement, i.e. which vertices are influenced and in which weight and direction according to FAP intensities. MPEG-4 provides a referencing method called Face Definition Tables, based on a piece-wise linear interpolation in order to animate the face model [8].

These tables (also referred to as Facial Animation Tables or FAT) provide information about which vertices should be translated or rotated for each FAP displacement. More information can be found in Magnenat-Thalmann & Thalmann [12].

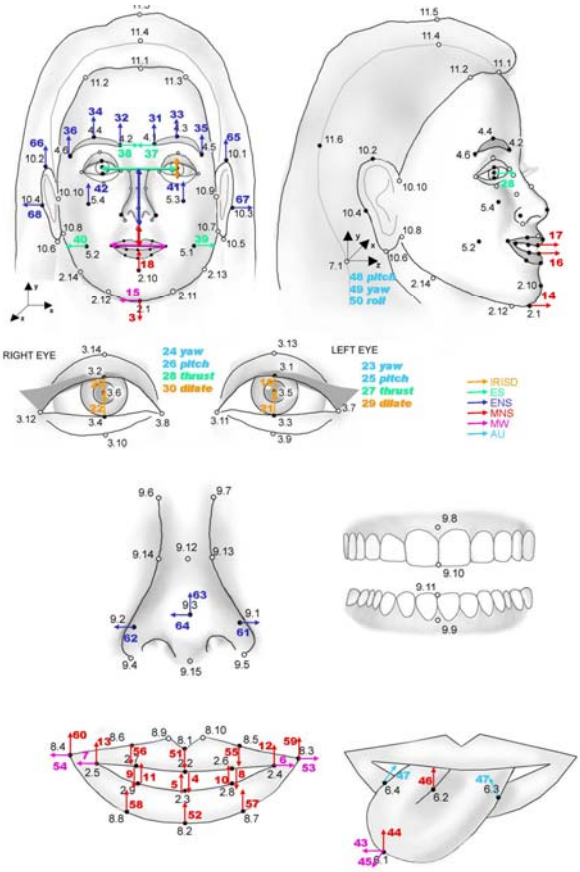


*Figure 1: Partial FDP position and FAP orientation*

The Face Definition Tables representation is optional: the information it provides can be created by an animator by defining each influence on each FAP manually. However areas such as the lips (shown in ***Figure 1***) make this work very tedious due to the close proximity of the 21 FAP values in a very small region.

### 2.2 Automatic and generic influence definition

In this paper we propose a generic approach to compute each region of influence. The main idea is to keep coherence and a maximum of simplicity in order to adapt this approach to different platforms or environments, but in the same time also to be able to produce realistic expressions. Exactly the same process is used for each FDP in order to define the influence area. In order to obtain a maximum of simplicity, we have developed an approach to compute deformations from simple FDP information.

## Computation description

The main problem is to find a correct definition of the influence area for each FAP according to its neighboring vertices. We propose an approach based on the following process:

- Compute the distribution of FDPs on the model and a relation of distance between them.
- For each vertex of the mesh:
    1. Define which control point is able to influence it
    2. Define the ratio of influence of each control point

A 3D face model is composed of different meshes for eyes, teeth, tongue and skin. The skin mesh is mainly defined by the holes for the eyes and the mouth. Also, often a face model has a vertex distribution that is not uniform over the mesh. In order to develop a model-independent approach we have to take into account these specificities, and then we should define an appropriate distance measurement. A measurement based on Euclidian distance is efficient to manage the variations in mesh density but it does not take into account problems like holes in the model. A measurement based on the topology like the number of edges between vertices takes the holes into account, but it is not efficient for the mesh density variations.
We propose to use a metric based on both aspects: Euclidian distance and mesh topology. The metric is computed following this rule: "the distance is equal to the sum of the edge distances along the shortest path between two vertices". Using this metric, we are able to manage in the same time, holes and mesh density variation on the face model.
Our approach is based on the definition of a list of influencing feature points for each vertex in the face mesh. Initially, all of the feature points are in the influence list. First, we find the closest feature point to the vertex (according to the previously defined metric). Then, we remove all the feature points from the list that are in the plane perpendicular to the vector between the vertex and the selected feature point (see ***Figure 2***). Then, we select the next closest feature point in the remaining list and we apply the same procedure until all feature points have been taken into account.

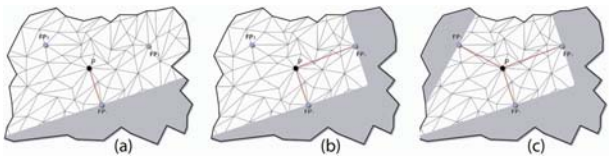


*Figure 2: 3 steps to define potential feature point*

When the list of influencing feature points is established, we compute the influence of each of them to the current vertex by using a rule based on the metric explained above. If $P$ is a vertex of the mesh, we compute a balanced sum $d$:

$$d = \frac{\sum_{i=1}^{n} d_i * \cos\theta_i}{\sum_{i=1}^{n} \cos\theta_i}$$

where $n$ equals the number of influencing feature points for $P$, $\theta_i$ the angle between $P$ and the feature point and $d_i$ the distance between the feature point and the vertex. The weight associated to a specific feature point for $P$ is computed as:

$$W_{i,P} = \sin\left(\frac{\pi}{2}\left(1 - \frac{d_i}{d}\right)\right)$$

By applying this weight computation for each vertex of the mesh, we obtain for each feature point a list of the vertices that are influenced by it, with an associated weight.
This approach takes into account the problem of overlapping regions and the diversity of the mesh (variation of density and holes).

## Application to the face by defining simple constraint

This generic approach is applied on the 3D face model 3 times, one time for each direction of deformation. When we look at the FAP repartition from directional point of view, we can see a big variation in the feature points (see ***Figure 3***).

In order to take into account this diversity and produce a realistic deformation, we compute a different influence area according to each displacement direction. We obtain then 3 different lists of vertices influenced for each feature point for

each region. We use this list during the animation in order to deform the mesh according to feature point's displacements.

This information can easily be represented in the FaceDefMesh format and be used in an MPEG-4 compliant face system.

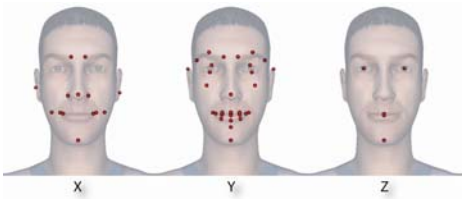


*Figure 3: Directional repartition of Facial Animation Parameters*

This initialization step of influence computation is done only once or is already saved in the MPEG-4 format. This approach is computationally very efficient (see ***Figure 4***). We only need a few seconds to compute all this information for a model composed with more than 10K polygons on a standard PC.

| Model Name | Nb Vertices | Nb Polygons | Influence process (sec.) | |
|---|---|---|---|---|
| Seb | 431 | 774 | PC1 | 0.50 |
| | | | PC2 | 0.37 |
| | | | PC3 | 0.23 |
| Linda | 2020 | 3849 | PC1 | 2.48 |
| | | | PC2 | 2.38 |
| | | | PC3 | 1.47 |
| Inam | 5781 | 11017 | PC1 | 6.71 |
| | | | PC2 | 5.48 |
| | | | PC3 | 3.34 |

*Figure 4: Computation of influence area on different face model. PC1 = AMD Athlon 1.7+, PC2 = PIV 2 Ghz, PC3 = PIV 3 Ghz.*

The advantages of our approach are multiple: we find a quick way to simplify the number of influent point, the same computation applies to all vertices independent of the part of face. Therefore we can add or remove feature points easily without changing the computation process. This approach works not only with MPEG-4 but with any feature point-based deformation approach.

### 2.3 Manual personalization

After computing the influences, a designer can still edit this information and modify it if necessary or if the designer would like to control a specific deformation. Starting for the previous process (automatic definition of influence) has big advantages. Defining all influence areas in a part of the face composed of many vertices and feature points is a lot of work. Starting from already calculated influence areas is easier for the animator to modify and it saves a lot of manual work.

## 3 Animation Design

Different approaches are possible in order to produce a facial animation stream more or less in real-time depending on the intended application. In this section, we will briefly present some of these approaches.

### 3.1 Text-to-visual approach

When starting from written text, we use a Text-to-Speech engine to produce the phoneme data and the audio. For defining the visemes and expressions, we use the Principal Components (PCs) as described by Kshirsagar et al. [10]. The PCs are derived from the statistical analysis of the facial motion data and reflect independent facial movements observed during fluent speech. The main steps incorporated in the visual front-end are the following:

1. Generation of FAPs from text
2. Expression blending: Each expression is associated with an intensity value and it is blended with the previously calculated co-articulated phoneme trajectories.
3. Periodic facial movements: Periodic eye-blinks and minor head movements are applied to the face for increased believability.

### 3.2 Optical capture

In order to capture a more realistic trajectory of feature points on the face, we are using a commercial optical tracking system to capture the facial motions, with 6 cameras and 27 markers corresponding to MPEG-4 FDPs. In the parameterization system, a total of 40 FDPs are animatable, but since markers are difficult to be set up on the tongue and lips, we use a subset of 27. We obtain 3D trajectories for each of the marker points as the output of the tracking system, suitable as well for 2D animation. During the data capture, head movements are not restricted and thus, a compensation process is required to obtain the local deformation of the marker [2].
Once we extract the head movements, the global movement component is removed from the motion trajectories of all the feature point markers, resulting in the absolute local displacements. The MPEG-4 FAP values are then easily calculated from these displacements. For example, FAP 3 open jaw is defined as the normalized displacement of FDP 2.1 from the neutral position, scaled by the FAPU MNS. As FAP values are defined as normalized displacements for the feature points from the neutral position, it is trivial to compute the FAP value given the neutral position and the displacement from this position [11]. The algorithm is based on a general purpose feature point based mesh deformation, which is extended for a facial mesh using MPEG-4 facial feature points.

### 3.3 Automatic personalization of animation

As presented above, an optical tracking system can be used in order to capture spatial motion of marker placed on the face. This information is converted in MPEG-4 FAPs that can be interpreted by any MPEG4 compliant facial animation engine. But during the conversion of motion capture data to FAP, we lose some information due of FAP restrictions (see section 2.1). In other words, with the standard format we cannot fully recover the same displacement as we had from the motion capture. The displacement in the directions which are not stored will be lost. We will present a solution to restore this information during the synthesis process.
As described above, our system is able to deform a face model according to the FDP position. This deformation is normally computed for FAP values in term of FAPU units. The system could move any FDP point and not only those points that are FAP values. In other words, we compute the deformation simply from the spatial position of control points. Thus we are able to deform a face model according to any the FDP position coming from FAP values or randomly. We propose then to apply a spring mass network in order to recalibrate the spatial position of control points. We do not want to connect the points linearly, because than it would be less dynamic.
In order to reach this goal, the first step is to define which FDP to connect and in which direction. For this we have studied different optical capture data and analyzed which displacements are lost during MPEG-4 conversion. We have obtained a relation between some feature points like eyebrows and head border for Z displacements or mouth displacement with cheek and the chin. With this mass spring correction of the FDP position before the deformation computation we are able to reproduce more degrees of freedom from the same data.

Another interesting idea with this approach is the possibility to personalize the value of the mass springs according to a specific person. The ultimate goal is to be able to

automatically set these spring mass parameters, and use these parameters in order to reproduce a more realistic animation with the same number of facial animation parameters. This research is ongoing.

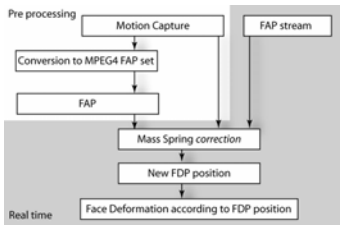


*Figure 5: Pipeline of facial animation with optional Mass spring correction of FDP position*

## 4 Emotional System/Model

The concept of emotion has been widely researched in the psychology field. Many approaches and theories exist, but according to Cornelius [3], they can be broadly organized in four different theoretical perspectives: the Darwinian, Jamesian, cognitive, and social constructivist perspectives.

The central organizing idea of the Darwinian perspective is the notion that emotions are phenomena that have evolved from important survival functions. This means that we should see more or less the same emotions (and expressions) in all humans. Theory and research from the Jamesian perspective views emotions from the perspective that it would be impossible to have emotions without bodily changes and bodily changes always come first. The cognitive approach to the study of emotions started with the work of Arnold [1]. In this approach, the central assumption is that thought and emotions are inseparable. More specifically, all emotions are seen within this perspective as being dependent on appraisal, the process by which events in the environment are judged as good or bad for us. Finally, Cornelius identifies the social constructivist approach to emotion study. Social constructivists believe that emotions are cultural products that owe their meaning and coherence to learned social rules.

In each of these perspectives, it is generally assumed that an emotional state can be viewed as a set of dimensions. The number of these dimensions varies among different researchers. Ekman [6] has identified six common expressions of emotion: fear, disgust, anger, sadness, surprise and joy, Plutchik [17] proposes eight, and in the OCC appraisal model, there are 22 emotions. The OCC model of appraisal has a computational counterpart that was developed by Elliott [7]. We propose a dynamic emotion simulation system that allows for any amount of different dimensions. The facial expressions are generated automatically from the emotional state [4].

## 5 Results and discussion

The techniques presented above have been tested and implemented on different platform and for different kind of applications. We present here briefly some result.

### 5.1 Multi-platform results

The same facial animation engine based on FDP position and manual adaptation in real time has been developed for different platforms such as PC, web or mobile.


*Figure 6 (a) (b) : Java web application (without plugin) for facial and body animation. (c) Mobile facial animation system. (d) PC application sample.*

### 5.2 Application of emotional system

Using the system for emotion simulation, we can create a talking head which integrated speech, facial animations and emotional expressions. *Figure 7* shows some different expressions on a virtual human.


*Figure 7: Some expressions displayed by different virtual humans.*

*Acknowledgements*

*This work is partially supported by the European Project HUMAINE (IST NoE 507422). A special thanks to Lionel Egger for his help in emotional design and to Edwin Tangenberg for work done on MPEG4 automatic recalibration data.*

## References

1. Arnold, M.B. (1960). Emotion and personality. *Columbia University Press*, New York.
2. Blostein, S.; Huang, T. (1988). Motion Understanding Robot and Human Vision. *Kluwer Academic Publishers*, pp. 329-352.
3. Cornelius, R.R. (1996). The science of emotion. *Research and tradition in the psychology of emotion*. Prentice-Hall, Upper Saddle River (NJ).
4. Egges, A.; Kshirsagar, S.; Magnenat-Thalmann, N. (2004). Generic personality and emotion simulation for conversational agents. *Computer Animation and Virtual Worlds*, **15(1):**1-13.
5. Ekman, P.; Friesen, W.V. (1978). Facial Action Coding System: A Technique for the Measurement of Facial Movement. *Consulting Psychologists Pres*s, Palo Alto, California.
6. Ekman, P. (1982). Emotion in the human face. *Cambridge University Press*, New York.
7. Elliott, C.D. (1992). The Affective Reasoner: A process model of emotions in a multiagent system. *PhD thesis*, Northwestern University.
8. Garchery, S.; Magnenat-Thalmann, N. (2001). Designing mpeg-4 facial animation tables for web applications. *In Multimedia Modeling 2001*, Amsterdam, pages 39-59.
9. Kim, J.W.; Song, M.; Kim, I.J; Kwon, Y.M; Kim, H.G.; Ahn, S.C. (2000). Automatic fdp/fap generation from an image sequence. *In ISCAS 2000 - IEEE International Symposium on Circuits and Systems*.
10. S. Kshirsagar, T. Molet, N. Magnenat-Thalmann (2001). Principal components of expressive speech animation. *In Proceedings Computer Graphics International*, pages 59-69.
11. Kshirsagar, S.; Garchery, S.; Magnenat-Thalmann N. (2001). Feature Point Based Mesh Deformation Applied to MPEG-4 Facial Animation. *Kluwer Academic Publishers*, pp. 33-43.
12. Magnenat-Thalmann, N.; Thalmann D. (2004), Handbook of Virtual Human, *Eds Wiley & Sons, Ltd.*, publisher, **ISBN: 0-470-02316-3**.
13. Noh, J.Y.; Fidaleo, D.; Neumann U. (2000). Animated deformations with radial basis functions. *In VRST*, pages 166-174.
14. Ortony, A.; Clore, G.L.; Collins A. (1988). The Cognitive Structure of Emotions. *Cambridge University Press*.
15. Ostermann, J. (1998). Animation of synthetic faces in mpeg-4. *In Computer Animation.* Held in Philadelphia, Pennsylvania, USA.
16. Pasquariello, S. ; Pelachaud C. (2001). Greta: A simple facial animation engine. *In 6th Online World Conference on Soft Computing in Industrial Appications*, Session on Soft Computing for Intelligent 3D Agents.
17. Plutchik, R. (1980). Emotion: A psychoevolutionary synthesis. *Harper & Row*, New York.