MPEG-4 Compatible Faces from Orthogonal Photos

Won-Sook Lee, Marc Escher, Gael Sannier, Nadia Magnenat-Thalmann
MIRALab, CUI, University of Geneva
http://miralabwww.unige.ch
E-mail: {wslee, escher, sannier, thalmann}@cui.unige.ch

Abstract

MPEG-4 is scheduled to become an International Standard in March 1999. This paper demonstrates an experiment for a virtual cloning method and animation system, which is compatible with the MPEG-4 standard facial object specification. Our method uses orthogonal photos (front and side view) as input and reconstructs the 3D facial model. The method is based on extracting MPEG-4 face definition parameters (FDP) from photos, which initializes a custom face in a more capable interface, and deforming a generic model. Texture mapping is employed using an image composed of the two orthogonal images, which is done completely automatically. A reconstructed head can be animated immediately inside our animation system, which is adapted to the MPEG-4 standard specification of face animation parameters (FAP). The result is integrated into our virtual human director (VHD) system.

1. Introduction

Newly standardized MPEG-4 is developed in response to the growing need for a coding method that can facilitate access to visual objects in natural and synthetic video and sound for various applications such as digital storage media, internet, and various forms of wired or wireless communication. ISO/IEC JTC1/SC29/WG11 (Moving Pictures Expert Group - MPEG) is currently working on this standard [16][17] to make it International in March 1999. Targets of standardization include mesh-segmented video coding, compression of geometry, synchronization between Aural and Visual (A/V) objects, multiplexing of streamed A/V objects, and spatial-temporal integration of mixed media types[18].

In a world where audio-visual data is increasingly stored, transferred and manipulated digitally, MPEG-4 sets its objectives beyond “plain” compression. Instead of regarding video as a sequence of frames with fixed shape and size and with attached audio information, the video scene is regarded as a set of dynamic objects. Thus the background of the scene might be one object, a moving car another, the sound of the engine the third etc. The objects are spatially and temporally independent and therefore can be stored, transferred and manipulated independently. The composition of the final scene is done at the decoder, potentially allowing great manipulation freedom to the consumer of the data. MPEG-4 aims to enable integration of synthetic objects within the scene. It will provide support for 3D Graphics, synthetic sound, text to speech, as well as synthetic faces and bodies. This paper will describe the use of facial definitions and animation parameters in an interactive real-time animation system.

1.1. Face definition and animation in MPEG-4

The Face and Body animation Ad Hoc Group (FBA) has defined in detail the parameters for both the definition and animation of human faces and bodies. Definition parameters allow a detailed definition of body/face shape, size and texture. Animation parameters allow the definition of facial expressions and body postures. These parameters are designed to cover all natural possible expressions and postures, as well as exaggerated expressions and motions to some extent (e.g. for cartoon characters). The animation parameters are precisely defined in order to allow an accurate implementation on any facial/body model. Here we will mostly discuss facial definitions and animations based on a set of feature points located at morphological places on the face. The two following sections will shortly describe the *Face Animation Parameters* (FAP) and *the Face Definition Parameters* (FDP).

1.1.1. Facial Animation Parameter set. The FAP are encoded for low-bandwidth transmission in broadcast (one-to-many) or dedicated interactive (point-to-point) communications. FAPs manipulate key feature control points on a mesh model of the face to produce animated visemes (visual counterpart of phonemes) for the mouth (lips, tongue, teeth), as well as animation of the head and facial features like the eyes or eyebrows.

All the FAP parameters involving translational movement are expressed in terms of Facial Animation Parameter Units (FAPU). These units are defined in order to allow the interpretation of FAPs on any facial model in a consistent way, producing reasonable results in terms of expression and speech pronunciation. They correspond to fractions of distances between some essential facial features (e.g. eye distance) (Figure 1). The fractional units used are chosen to allow enough accuracy.

The parameter set contains two high level parameters. The viseme parameter allows rendering of the visual aspect of the lower part of the face without the need to express them in terms of other parameters. Similarly, the expression parameter allows the definition of six high level facial expressions.

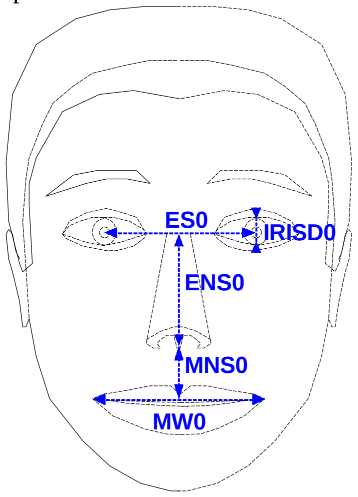


Figure 1: The Facial Animation Parameter Units

1.1.2. Facial Definition Parameter set. An MPEG-4 decoder supporting the Facial Animation must have a generic facial model capable of interpreting FAPs. This insures that it can reproduce facial expressions and speech pronunciation. When it is desired to modify the shape and appearance of the face and make it look like a particular person/character, FDPs are necessary. The FDPs are used to personalize the generic face model to a particular face. The FDPs are normally transmitted once per session, followed by a stream of compressed FAPs.

The FDP fields are as followings:

- **FeaturePointsCoord** – it specifies feature points for the calibration of the proprietary face.
- **TextureCoords** – it specifies texture coordinates for the feature points.
- **TextureType –** it contains a hint to the decoder on the type of texture image, in order to allow better interpolation of texture coordinates for the vertices that are not feature points.
- **FaceDefTables -** this field describes the behavior of FAPs for the face in the **FaceSceneGraph**.
- **FaceSceneGraph -** this node can be used as a container of the texture image as explained above or the grouping node for the face model rendered in the compositor and therefore it has to contain the face model.

We do not deal with **FaceDefTables** in this paper.

1.2. Outline

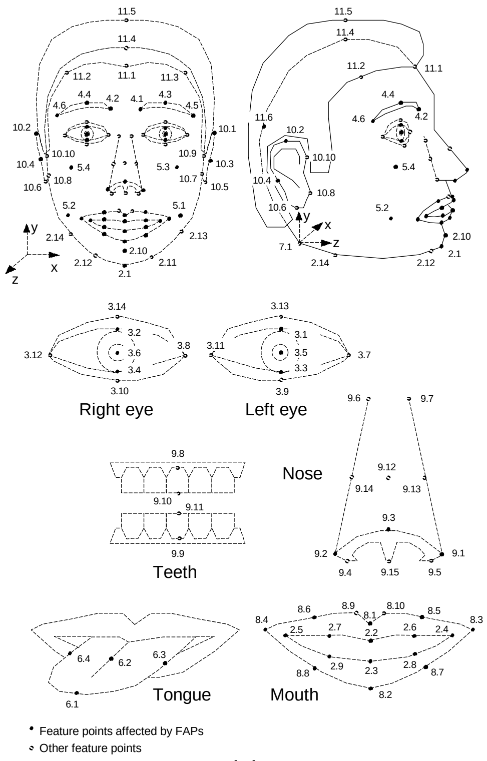
The organization of this paper is as follows. Section 2 is dedicated to the reconstruction processes of feature points detection (FeaturePointsCoord), modification of a generic model and seamless texture mapping (TextureCoords and texture image). In Section 3, we propose animation methods using FAPs. We introduce a real-time virtual director system, which uses MPEG-4 definition for faces in Section 4.

2. FDP cloning from two orthogonal pictures

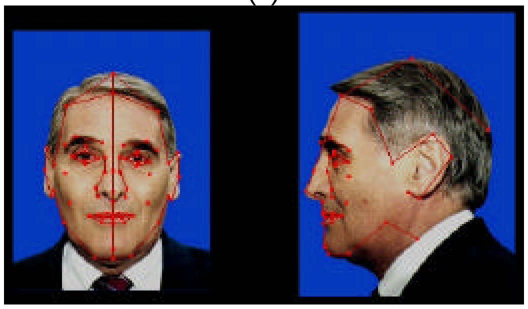
Approaches to the realistic reconstruction of individuals include the use of a laser scanner [4], a stereoscopic camera [1], or an active light stripper [5]. Some of these approaches also include a view to animation. There is also increasing interest in utilizing a video stream [7] to reconstruct heads. These methods are not developed enough, however, to be commercialized in the near future (such as in a camera-like device) in terms of direct input of data for the reconstruction and final animation as output. Modeling has also been done from picture data [3][6], detecting features, modifying a given generic model and then mapping texture on it. There was an approach to make an individualized head by modifying a generic model using MPEG-4 parameters, but it used range data as input, such as laser scanner[12], instead of easy-and-cheap-device like a camera.

Our methods use orthogonal photos (front and side views) as input to reconstruct 3D facial models. The method is based on extracting FDPs on photos, which initializes a custom face in a user-friendly interface, and deforming a generic model. Texture mapping is employed using a composed image from the two images in a fully automatic way.

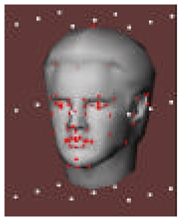
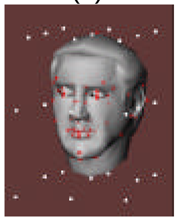
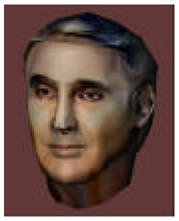
FDPs contains features of eyes, eyelids, eyebrow, lips, nose, chin, cheek, tongue, head rotation point, teeth, ear and some of face and hair outlines as shown in Figure 2(a). The FDP points on a pair of orthogonal photos are shown in Figure 2(b), which has some line segment to help the user to detect FDPs. The model in Figure 2(c) shows a previously constructed, animation-ready generic model, which is transformed to an individualized head, based on the FDPs shown in Figure 2(d). The two heads in Figure 2(c) and Figure 2(d) have red points on them, which are FDPs in 3D composed of two FDP sets, red points in Figure 2(b). The final cloned model shown in Figure 2(e) is ready to be animated immediately.



(a)



(b)

(c) (d) (e)

Figure 2 (a) FDP feature point set and feature points affected by FAPs. (b) FDPs on input pictures. (c) A generic model (d) An individualized shape (e) A final virtual human for animation.

2.1. FDP extraction and a shape modification

2.1.1. Some problems with FDP extraction on photos. There are total 84 parameters in FDPs. Some of them are not visible on photos (for example, 4 points on teeth, 4 points on tongue). In our animated model, since points 3.6 and 3.5 can be extracted approximately from 3.12, 3.8, 3.11 and 3.7, we do not use them. There are also some difficulties to identify features on photos such as 11.4 and 11.6. Although they are not visible on photos, we need some points to define hair shape. So we use 11.4 and 11.6 as boundary of head instead of 11.5. Finally we make use of only 84 – 4 – 4 – 2 – 1 = 73 points.

Another difficulty also comes from locating some points when they are too near each other, such as (3.4, 3.10), (3.3, 3.9), (2.5, 8.4), and (2.4, 8.3). Points (2.7, 2.9), (2.2, 2.3), and (2.6, 2.8) are sets, whose positions are the same when a person has a closed mouth. In addition, some important points to characterize a person are not defined in FDPs, which causes some problem later. There are too few points on nose, especially on side profiles. There is no parameter defined on central line on nose between two eyes and between two eyebrows, which limits the generation of various shapes on side profiles. There are not enough points on eyes and nosewings and it results some texture fitting error later. If there are some more points on outlines of a face and hair, the characterizing could have given better results.

2.1.2. Head modification. We deform a generic model using FDP points extracted from photos as control points. The deformation of a surface can be seen as an interpolation of the displacements of the control points.

Free-Form Deformations (FFD) [8] belong to a wider class of geometric deformation tools. Among several extensions of a basic FFD, we use Dirichlet FFD or DFFD, which is initiated by Farin [10] who extends the natural neighbors' interpolant based on the natural neighbors' coordinates, the Sibson coordinate system [9].

Here, DFFD [27] process gets new geometrical coordinates for a modification of the generic head on which are situated the newly detected feature points. All points on this head are located utilizing the feature points as constraint control points for DFFD. The convex hull of control points contains the most points on a 3D head, but there are some missing points outside the convex hull. So we add 27 extra points surrounding the 3D head, which are white points in Figure 2(c) to adjust missing points by FDP control points. These 27 points are not feature points, but control points for deformation. The correct shapes of the eyeballs and teeth are assured through translation and scaling appropriate to the new head. As shown in Figure 2(d), this is a rough matching method; it does not attempt to locate all points on the head exactly, in contrast to range data from laser scanners or stereoscopic cameras, which create an enormous numbers of points. Considering the input data (pictures from only two views), the result is quite respectable. Most important, it greatly limits the size of the data set associated with an individual head, as is necessary to accelerate animation speed. The problem of how to reduce the size of the data from range data equipment for animation purposes has not been satisfactorily solved.

2.2. Automatic Texture Mapping

Texture mapping serves not only to disguise the roughness of shape matching determined by feature point matching, but also to imbue the face with more realistic complexion and tint. We use information from FDP points detected to generate texture automatically, based on the two views. We then obtain appropriate texture coordinates for every point on the head using the same image transformation.

2.2.1. **Texture Generation.** The main criterion is to obtain the highest resolution possible for the most detailed portions [6]. We first connect two pictures along predefined feature lines using geometrical deformations and, to avoid visible boundary effects, a multiresolution technique is used. The feature lines are defined as a piecewise lines of 8 points such as for example (11.4, 11.3, 4.5, 10.9, 10.7, 2.13, 2.11, 2.1) on the left side, which follows from a point on top skull, a point between hair and forehead, an outer point of left eyebrow, an upper point between left ear and face, a lower point between left ear and face, a left point on jaw, a left point on chin, a middle point on bottom of chin. The counter feature piecewise line of the right side is also defined in similar way. We keep the front face and deform the side face to match the feature line on the side face to the feature line on the front face. The left view is a flip image of right view. Therefore, we use a side photo for both the right and the left views. More detail is found in reference [6]. Figure 3(a) shows the result of geometric deformation.

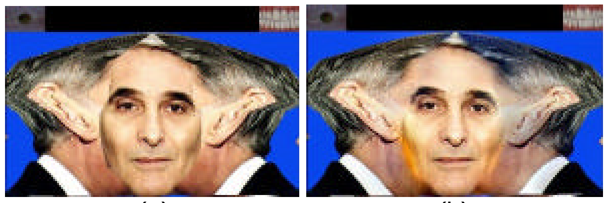

(a) (b)

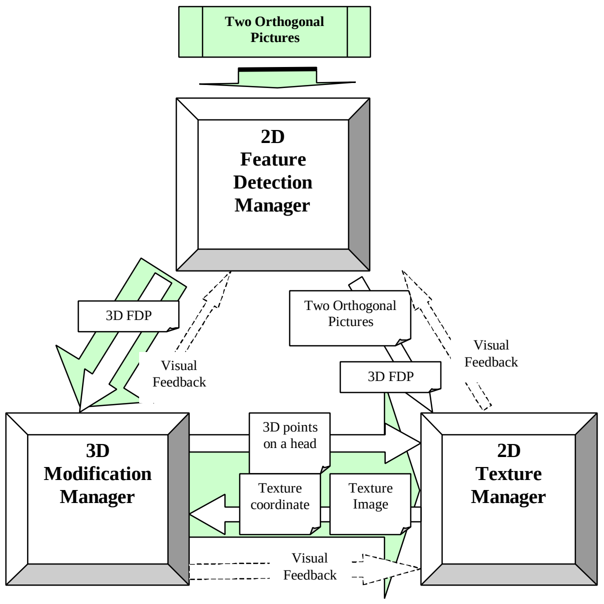
Figure 3 (a) A texture integrated after geometric deformation. (b) A texture with multiresolution technique.

Although there is no hold or visible mismatch among features, it has a strong boundary between front and right/left parts. No matter how carefully the photographic environment is controlled, boundaries are always visible in practice. To correct this problem, we apply multiresolution techniques. The three images resulting from the deformation are merged using a pyramid decomposition [2] based on the Gaussian operator. This multiresolution technique is effective in removing the boundaries between the three images. As in Figure 3(a), skin colors are not continuous when the multiresolution technique is not applied. The image in Figure 3(b) shows the results with the technique, which has smoothed the connection between the images without visible boundaries.

2.2.2. Texture Fitting. To find suitable coordinates on a combined image, for every point of a head, we first project an individualized 3D head onto three planes: the front, right and left. Guided by the feature lines used for image merging in the above section, we decide to which plane a point on a 3D head is to be projected. The points projected on one of three planes are then transferred to either the front or the side 2D-feature point space. Finally, one more transformation on the image space is done to obtain the texture coordinates, and then the final mapping of points on a texture image is generated.

The final textured head is shown in Figure 2(e). The eye and teeth fitting process is done with predefined coordinates and transformations related to the texture image size. After doing this once for a generic model, it is fully automated.

2.3. User-Friendly Graphical Interface



(a)

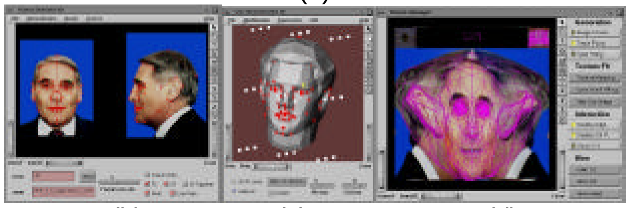

(b) (c) (d)

Figure 4 (a) Communication in Graphical interfaces for (b) a feature detection manager in 2D, (c) a modification manager in 3D and (d) a texture manager.

As mentioned in Section 2.1.1, there are some difficulties in obtaining satisfactory results using MPEG-4 FDPs completely automatically, especially for texture fitting between the hair and the face region and nosewings due to the lack of feature points. We provide a user-friendly interface to accelerate the results, whose diagram is shown in Figure 4. Normally the 2D-feature detection manager is processed in an interactive way and then the 3D-modification manager and 2D-texture manager follow in a fully automatic way. However, we introduce a visual feedback method to correct some minor errors.

2.3.1. Shape-texture separation. Good pairs of orthogonal photos are hard to come by, unless the photography is done by us in a controlled situation. Sometimes sets of images are available, some of which have good resolution, while in others the shape is clearer. For better results, we can combine shape data from the latter with texture image from the former. The process is outlined in Figure 5.

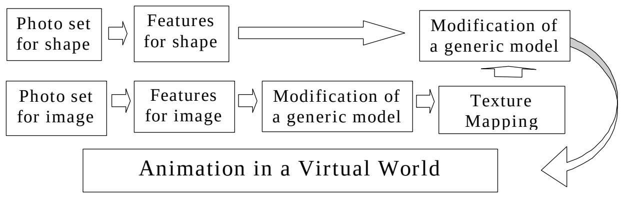


Photo set for shape
Features for shape
Modification of a generic model
Photo set for image
Features for image
Modification of a generic model
Texture Mapping
Animation in a Virtual World

Figure 5 A procedure for reconstruction from shape-image-separated input photos.

3. Real-time animation system

This section describes a real-time interactive animation system based on FAPs. To allow an inexperienced user to generate complex animations without getting into the strict specification of MPEG-4, we have implemented a multi-layered system that uses high-level actions to interface with the MPEG-4 FAP-FDP specifications. The definition of high-level actions and how they are defined and implemented will be detailed in the following sections.

3.1. From High-Level actions to FAPs

In order to allow an animator to work on a task level where she/he can define animation in terms of its inherent sources, we use a multi-layered approach [19], where at each level of the hierarchy, the degree of abstraction increases. It offers the advantage of providing the user a relatively low level of complexity with concepts more global than the previous layers. For instance, complex expressions such as emotions or even speech can be described more easily in this multi-layered system. In our whole animation system, we can distinguish four levels of abstraction as seen in Figure 6.

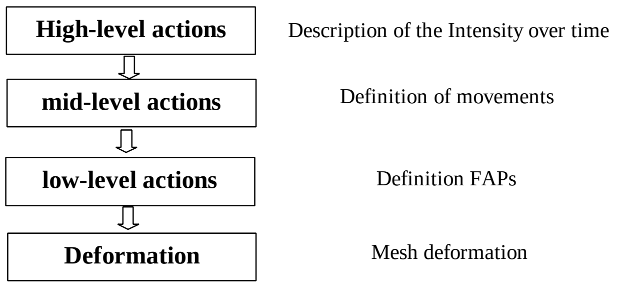


Figure 6 Levels hierarchy and effects

On the high-level actions layer, the animator can choose between three types of envelopes describing the evolution of the face deformation through time as shown in Figure 7.

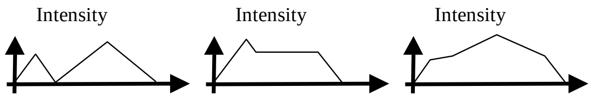


Figure 7 The three available envelope shapes of high-level actions

A mid-level action is a snapshot of the face: a static position/expression that has been defined by the user as shown in Figure 8. The user has interactively set a number of values to the desired FAPs to get the desired expression. They are generally composed/defined manually using 3D modeling software although they can also be extracted more automatically with the help of an external device such as a camera [21][22][23].

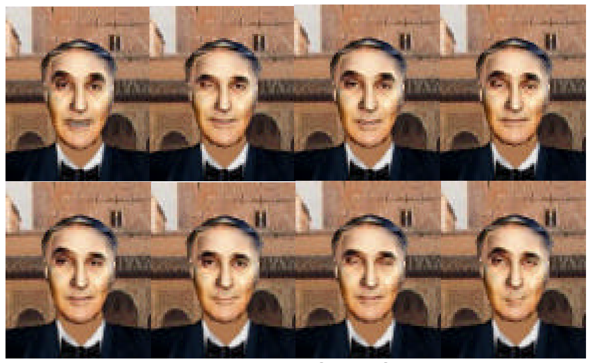


Figure 8 Mid-level user-defined facial expressions.

The low-level layer is the description of the FAP location on the virtual face in a neutral position. Among the high-level actions (the only layer visible to the animator), one can distinguish three different types of actions: basic emotions, visemes and user-predefined expressions. The basic emotions correspond to the 6 emotions defined in the first field of the MPEG-4 FAP definition: joy, sadness, anger, fear, disgust and surprise.

The visemes (visual phonemes) used directly correspond to the 14 visemes defined in the second field of the MPEG-4 FAP standard. A sequence of temporized visemes (with duration information) can be saved as a high-level speech action with its corresponding audio speech stream. Finally, an open set of expressions built by the user can be used to generate animations that are more original. Interactive real-time animation imposes that the animator can activate any action of any type at any time. This allows the possibility of having several high-level actions active at the same time. As long as these actions do not modify the same region of the face i.e. the same FAP, there is no problem, but when they do, it is necessary to have a mechanism to perform composition or blending. Good blending of several high-level actions is important to guarantee a continuous animation.

The deformation of the 3D-face mesh driven by FAP displacements is based on Rational Free-Form Deformations (RFFD) [11]. A surrounding control box controls each set of vertices of the face that is within the influence of a FAP. Setting a specific weight to a control point of the control box generates the mesh deformation.

3.2. Action blending

The problem of blending actions surges when several high-level actions are active at the same time and are each setting a value to the same FAP at the same moment. If it is not handled with care, it can produce discontinuities in the animation. This is particularly important if we want to get a natural facial animation. A discontinuity can happen at the activation or at the termination of each high-level action when combined with another one. For instance if we take the two FAP frame sequences (the first graph in Figure 9) and compute a simple average of each frame, one can see that at the beginning of action 2 and at the end of action 1 we have a big discontinuity as seen in Figure 9 (c). This problem was well treated by Cohen and Massaro[24].

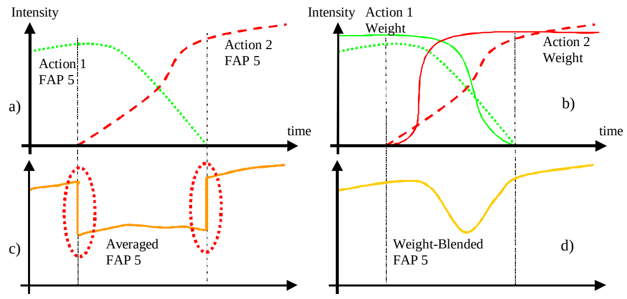


Figure 9 FAP composition problem and solution

The solution we have chosen is to add a weighting curve to each high-level action. It minimizes the contribution of each high-level action at its activation and at its termination as seen in Figure 9. The weighting curve values are generally set between *0* and *1*. The composition formula used to compute the value of an FAP at a specific time *t* modified by *n* high-level actions is given by the below equation.

$$FAP_{Final} = \frac{\sum_{i=1}^{n} Weight_{i,t} \cdot Intensity_i \cdot FAPValue_{i,t}}{\sum_{i=1}^{n} Weight_{i,t}}$$

where *Intensity* is the global intensity of the high-level action *i*, and *FAPValue* is the FAP movement in the high-level action *i* at time *t*.

This blending function is acting only at the FAP level and only when several high-level actions are setting values to the same action-unit. We have introduced the *Intensity* factor to enable the modulation of a similar high-level action in different contexts. For instance, we can use the same high-level action to generate a big smile or a small one depending on the value of the intensity associated to it. It is also used in the complex process of lip movement for speech articulation.

3.3. Speech co-articulation

Several works have been done in the area of speech co-articulation and visemes definition [24][25][26]. Concerning the visemes co-articulation, we have defined four types of visemes: vowels, consonants, plosives and fricatives. Each group has their own co-articulation parameters. In the process of articulating a word or a sentence our brain and mouth do some on the fly ‘pre-processing’ in order to generate a fluent and continuous speech. One of these complex processes is the mixing of lip/jaw movements to compose basic sounds or phonemes of the sentence. For instance when you pronounce ‘hello’, even before saying the ‘h’ your mouth is taking the shape of the ‘e’ that is coming afterwards. During the pronunciation of the ‘ll’, your mouth is also making the transition between the ‘e’ and the ‘o’. Temporal behavior of the different visemes groups has been specified. Especially the vowels tend to overlap the other visemes of the sentence. The region of the mouth that is deformed is also specific to each viseme group. Typically the plosives act on the closure of the mouth and lip protrusion, and have little action with cornerlips. Applying the blending function that was described previously we have associated 3 parameters to each viseme group: overlapping, intensity and weight. The overlapping describes the % of time that a viseme overlaps its neighbors. The intensity allows stressing some articulations or simulating shouting or whispering. Finally, the weight fixes priorities to some visemes, typically plosives for which the mouth closure is mandatory. For instance if we have a high-level action of surprise (with mouth opening) and we want to articulate ‘b’ in the same time frame, we will give a low weight to the surprise action and a high one to the ‘b’ viseme to guarantee the closure of the mouth. The next figure gives an example of co-articulation for the word ‘happy’.

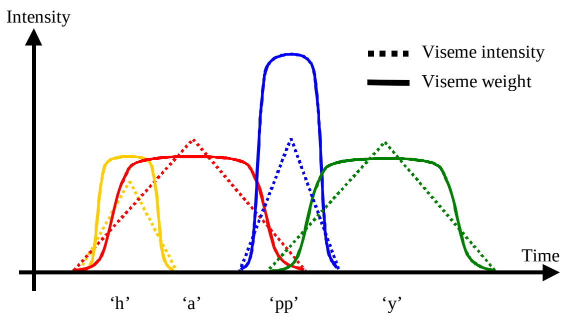


Figure 10 Viseme co-articulation for the word 'happy'

3.4. From speech or text to mouth movements

To generate speech, various methods can be used. What is needed as input, is an audio file with its corresponding temporized phonemes. Two natural types of data can be used to generate phonemes: speech and text. We are using the *AbbotDemo* Freeware to extract the phoneme information from a speech file. For text to phoneme, we are using *Festival* Speech Synthesis System from University of Edinburgh, UK, and for phoneme to speech (synthetic voice), we use *MBROLA* from Faculté Polytechnique de Mons, Belgium. Both are public domain. To go from phonemes to visemes we have integrated a correspondence table. This table has been built by designers using a proprietary face modeling software with snapshots of real talking faces to generate the visemes actions.

4. Virtual Human Director

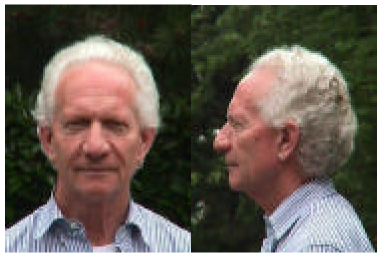
In the previous sections, we described how we generate a virtual face and its animation based on MPEG-4 parameters. In this section, we present a system managing real-time animation of virtual humans in a virtual environment based on MPEG-4 parameters.

4.1. Real-time director

Virtual Human Director (VHD) is a real-time system developed for the creation of scenario involving virtual humans [15]. A user-friendly GUI (Figure 12(a)) gives the control of the virtual faces, bodies and cameras to a single user. The issue of easy control of a virtual face was raised in [13]. The interface of VHD provides an extension of this concept through simple and easy controls of multiple actors and camera in real-time. An important aspect is that the user can trigger the main actions in real-time, similar to a Producer directing actors before a shooting. As this task is tricky to handle, we limit the interaction of the user to the high-level actions. The speech is activated by typing/selecting simple text-based sentences and predefined facial and body animation is available in the interface. In our case, these animations have been generated using proprietary facial animation software. The duration of the facial animation is controlled interactively from the interface. Though the actions are easy to trigger in real-time for one single virtual human, the task becomes more and more complicated with several virtual humans. To simplify this task, we develop a timing control device for recording activation of actions in advance and for repeating events such as eye blinking. A simple scripting language was also developed to be able to program activation of a list of actions through time. It allows the setup of "background" virtual humans with completely predefined "behavior". The user can also use this scripting/recording tool to associate actions together. For example, one can associate a smile, a head motion and the sentence "I am happy" into one single new action useable from the interface.

4.2. MPEG-4 in VHD

4.2.1. Virtual Human modeling. To create new virtual humans available in VHD, we are using heads created as described in Section 2.


(a)

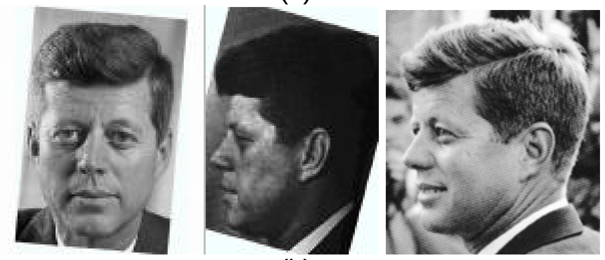

(b)

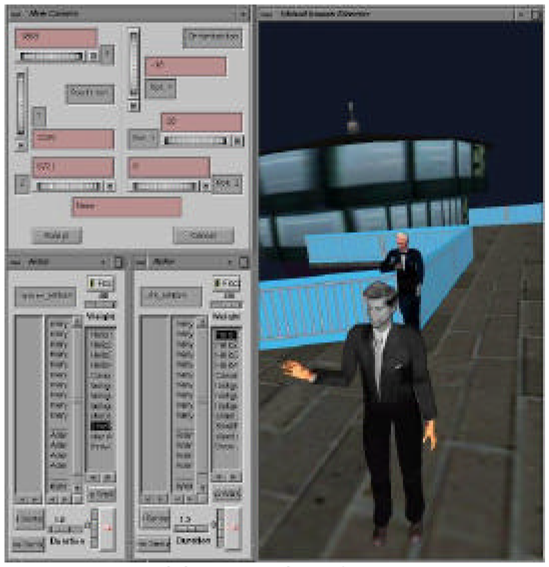
Figure 11 Input pictures.

Figure 11 (a) and (b) are orthogonal pairs, which are used for individualized virtual human modeling using FDP. Figure 11 (b) is an example of shape and texture separation described in Section 2.3.1. The front view is satisfactory both for shape and for texture resolution. The outline of the second image is clear, but the third image has much better resolution.

Template bodies are created for the male and female on which we replace the generic head. The methodology used for the creation of the virtual bodies is described in [14]. To animate a virtual head in VHD, the only sets of data needed are the topology and the facial-animation feature points. The heads created with our methodology, using

FDP parameters provide this information, so the heads are directly usable in VHD. Figure 12(b) shows the various snapshots of these three virtual clones acting in 3D virtual world.

4.2.2. **Animation.** In order to animate the virtual humans using the MPEG-4 parameters, we provide a set of predefined animations to each virtual human.


(a) A user interface

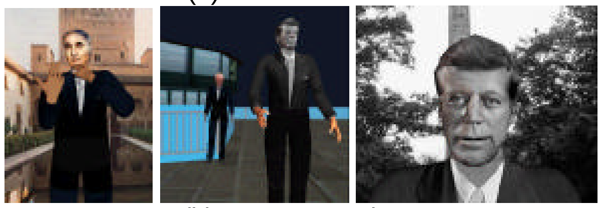

(b) Some snapshots

Figure 12 Real-time animated virtual humans in VHD.

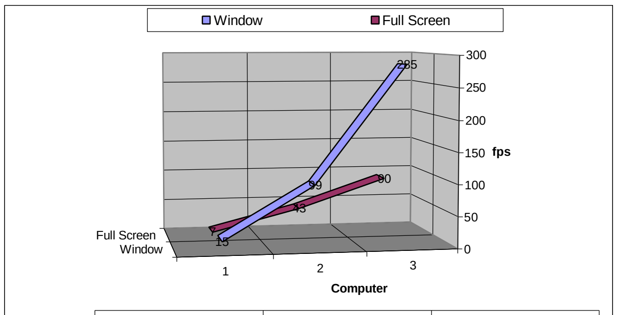
The body is animated using key-frames based on the joint angles of a virtual skeleton, which are close to the MPEG-4 BDPs. A most important aspect is that we can mix facial animations in real-time. For example we can direct a smile while the virtual human is speaking and we can also mix the emotion 'smile' with the emotion 'surprise' to obtain a new combined expression. The idea is then to provide only a simple set of facial animations as the basic emotional states of a virtual human. By combining these emotions, we can obtain more complex animation of the virtual face. The basic set of facial emotions includes local animations of the eyes, the eyebrows, the mouth and global animation of the face. By decomposing the animation of the face into one set for the upper part, one for the lower part, and one for the global facial animation, we can obtain a great range of possible combination. The result of such a combination can also be saved into a new FAP file. One can use this way to generate new facial animations, or to record a complete track of facial animations in MPEG-4 format.

4.3. VHD real-time performance

We tested our software extensively within the European project VISTA together with broadcasting partners, to produce television programs. After several tests with designers and directors, our concept of one single integrated platform for virtual humans has obtained good feedback. Our interface is very straightforward and has the highest level of control allowing designers to control virtual humans in a natural fashion. We included several traditional camera operations into our interface to allow directors to use VHD without a steep learning curve.

We tested VHD with a close-up virtual head made of 2313 triangles and fully textured. Table 1 presents the results of our trials on the following computers using full screen (1280x1024) and a 400x500-pixel window:

1) O2 R5000 at 200MHz
2) Octane R10000 at 195 MHz
3) Onyx2 2x R10000 at 195MHz



	1	2	3
🟦 Window	15	99	285
🟪 Full Screen	7	43	90

Table 1 Number of frames per second in VHD.

5. Conclusion

We have introduced a set of methods for the generation of realistic faces from just a pair of orthogonal pictures and for a real-time animation system in a virtual world, which are compatible with the newly standardized MPEG-4 specification of the facial object in object based coding syntax.

One key to our technique is the efficient reconstruction, based on extracting MPEG-4 *face definition parameters* (FDP), of animation-ready individualized faces fitted with seamless textures. This involves shape acquisition through the modification of a generic model and texture fitting through geometric deformation of an orthogonal pair of texture images, followed by multiresolution procedures.

Some problems for FDPs are also discussed. A reconstructed head can be animated immediately inside our animation system, which is adapted to MPEG-4 standard specification of *face animation parameters* (FAP). The virtual human director (VHD) system allows a user to act as a director to control virtual humans in real-time. It includes facial and body animations and speech for a given text input.

Ongoing research is a full integration of MPEG-4 compatible individualized face-to-face communication system through network.

6. Acknowledgment

The authors would like to thank every member of MIRALab, especially Laurent Moccozet, Pierre Beylot, and Hyewon Seo. We are also grateful to Chris Joslin for proof reading this document. This project is funded by European projects VPARK, VIDAS, VISTA and eRENA.

7. References

[1] http://www.turing.gla.ac.uk/turing/copyrigh.htm

[2] Peter J. Burt and Edward H. Andelson, "A Multiresolution Spline with Application to Image Mosaics", *ACM Transactions on Graphics*, 2(4):217-236, Oct., 1983.

[3] Tsuneya Kurihara and Kiyoshi Arai, "A Transformation Method for Modeling and Animation of the Human Face from Photographs", *Proc. Computer Animation'91*, Springer-Verlag Tokyo, pp. 45-58, 1991.

[4] Yuencheng Lee, Demetri Terzopoulos, and Keith Waters, "Realistic Modeling for Facial Animation", *Computer Graphics (Proc. SIGGRAPH'96)*, pp. 55-62, 1996.

[5] Marc Proesmans, Luc Van Gool. "Reading between the lines - a method for extracting dynamic 3D with texture", *Proc. of VRST'97*, pp. 95-102, 1997.

[6] Won-Sook Lee, Nadia Magnenat Thalmann, "Head Modeling from Pictures and Morphing in 3D with Image Metamorphosis based on triangulation", *Proc. CAPTECH'98 (Modelling and Motion Capture Techniques for Virtual Environments)*, (Springer LNAI LNCS Press), Geneva, pp.254-267, 1998

[7] P. Fua, "Face Models from Uncalibrated Video Sequences", In *Proc. CAPTECH'98*, pp. 215-228, 1998.

[8] Sederberg T. W., Parry S. R., "Free-Form Deformation of Solid Geometric Models", *Computer Graphics (Proc. SIGGRAPH'86)*, pp. 151-160, 1986.

[9] Sibson R., "A Vector Identity for the Dirichlet Tessellation", *Math. Proc. Cambridge Philos. Soc.*, 87, pp. 151-155, 1980.

[10] Farin G., "Surface Over Dirichlet Tessellations", *Computer Aided Geometric Design*, 7, pp. 281-292, North-Holland, 1990.

[11] Kalra P, Mangili A, Magnenat-Thalmann N, Thalmann D, "Simulation of Facial Muscle Actions Based on Rational Free Form Deformations", *Proc. Eurographics'92*, pp. 59-69, NCC Blackwell,1992.

[12] Marc Escher, Igor Panzic, N. Magnenat-Thalmann, "Facial Deformation from MPEG-4", *Proc. Computer Animation'98*, 1998.

[13] N. Magnenat Thalmann, P. Kalra, M. Escher, "Face to Virtual Face", *Proc. of the IEEE*. Vol.86, No. 5, May, 1998.

[14] J. Shen, D. Thalmann, "Interactive Shape Design Using Metaballs and Splines", *Proc. Implicit Surfaces*, Eurographics, Grenoble, France, 1995.

[15] G. Sannier, S. Balcisoy, N. Magnenat Thalmann, D. Thalmann, "An Interactive Interface for Directing Virtual Humans", *Proc. ISCIS'98*, IOS Press, 1998.

[16] SNHC, "INFORMATION TECHNOLOGY–GENERIC CODING OF AUDIO-VISUAL OBJECTS Part 2: Visual", *ISO/IEC 14496-2, Final Draft of International Standard, Version of: 13, November, 1998, ISO/IEC JTC1/SC29/WG11 N2502a*, Atlantic City, October 1998.

[17] SNHC, "Text for ISO/IEC FDIS 14496-1 Systems (2nd draft)", *ISO/IEC JTC1/SC29/WG11 2501, 1997. ISO/IEC JTC1/SC29/WG11 N2501*, Nov., 1998.

[18] Doenges P., Lavagetto F., Ostermann J., Pandzic I.S., Petajan E., "MPEG-4: Audio/Video and Synthetic Graphics/Audio for Mixed Media" , *Image Communications Journal*, Vol. 5, No. 4, May, 1997.

[19] Kalra P. "An Interactive Multimodal Facial Animation System", *PhD Thesis nr. 1183*, EPFL, 1993.

[20] Koenen R., Pereira F., Chiariglione L., "MPEG-4: Context and Objectives", *Image Communication Journal, Special Issue on MPEG-4*, Vol. 9, No. 4, May 1997.

[21] Terzopoulos D, Waters K, "Analysis and synthesis of facial image sequences using physical and anatomical models", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):569-579, 1993.

[22] Blake A, Isard M (1994), "3D position, attitude and shape input using video tracking of hands and lips", *Computer Graphics (Proc. SIGGRAPH'94)*, 28:185-192, July 1994.

[23] Pandzic I., Kalra P., Magnenat Thalmann N., Thalmann D., "Real time facial interaction", *Displays* 15 (3):157-163, 1994.

[24] M.M. Cohen and D.W. Massaro, "Modeling Coarticulation in Synthetic Visual Speech", *Models and Techniques in Computer Animation*, N. M. Thalmann & D. Thalmann (Eds.), Tokyo: Springer-Verlag, pp. 139-156, 1993.

[25] Benoît, C., Lallouache, T., Mohamadi, T., & Abry, C., "A set of French visemes for visual speech synthesis", G. Bailly & C. Benoît, (Eds.), *Talking machines: Theories, Models, and Designs*, Amsterdam: North Holland, 485-504, 1992.

[26] Boyce, S. E., "Coarticulatory Organization for Lip Rounding in Turkish and English", *Journal of the Acoustical Society of America*, 88(6), 2584-2595, 1990.

[27] Moccozet L., Magnenat Thalmann N., "Dirichlet Free-Form Deformations and their Application to Hand Simulation", *Proc. Computer Animation'97*, IEEE Computer Society, pp.93-102, 1997.