MPEG-4 COMPLIANT INFORMATION EXTRACTION FROM AN ANIMATED FACE IN MAYA

Charalampos Laftsidis, Constantine Kotropoulos, Ioannis Pitas

Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki 54006, Greece,
e-mail:pitas@zeus.csd.auth.gr

ABSTRACT

This paper presents a method for extracting facial information encoded in MPEG-4 format from an animated face model in Maya. The method generates the appropriate data, as specified in the MPEG-4 standard, such as the Face Definition Parameters, the Face Animation Parameters and the Facial Animation Table. The described procedure was implemented as a plug-in for a face model defined in Maya, which operates given only the animated model and the definition of its vertices that correspond to Face Definition Parameters. The extracted data were tested on a publicly available FAP-player. This application was further enhanced so as to allow for loading any model and its corresponding FAT table.

Categories: Facial Animation, MPEG-4, Data compression.

Keywords: Face Definition Parameters (FDPs), Face Animation Parameters (FAPs), Facial Animation Table (FAT).

1. INTRODUCTION

Among others, the MPEG-4 standard offers new capabilities in the domain of computer graphics. Specifically, in the domain of synthetic/natural hybrid coding a standard has been defined for the coding and representation of the human body and face, both for still and animation models. Its specification can be found in detail in [1].

Concerning the face specification, the standard introduces the Face Definition Parameters (FDPs) and the Face Animation Parameters (FAPs). The former parameters specify the three dimensional location of 84 points on a neutral face. These are points that correspond to facial features and, therefore, can roughly define the shape of the face. The latter parameters specify the displacements of several of the FDPs, which result in actual facial movements a realistic human face would make. For a detailed description on the FAPs' operation, see [1] and [5].

There are several modes of operation that the MPEG-4 standard specifies. According to one of them, an encoder-decoder couple can work by having the encoder only transmitting the FAP data to the decoder, which is the minimal information required to animate a model. The decoder then decodes the animation information onto an arbitrary model.

In a higher operation mode, the decoder, apart from the animation data, requires the facial model from the encoder too. In this case, additional information has to be provided in order to animate the model that the encoder provides. This is achieved by submitting the Facial Animation Table (FAT), which corresponds to the transmitted model. The FAT defines correspondences between the FAPs and the displacements of the vertices of the model. In other words, a FAT correlates the FAPs, which animate a model, with the movement of each specific vertex of the facial model mesh. Furthermore, it provides the appropriate information on how to displace each vertex according to the values the corresponding FAP can admit. A detailed description on the structure of a FAT is specified in [2]. A specific description of FAT functionality is given in [3] and [4].

The first part of this paper presents the Maya plug-in, which has been developed for extracting the forementioned information from an animated face model. It only requires the FDP points to be defined. It then extracts the FAPs and the FAT automatically from the animation, i.e. without prior knowledge of the animation modeling techniques. In the second part, an ap-

plication is presented, which can load the model and its FAT and can also perform an animation based on a FAP file.

2. MAYA PLUG-IN IMPLEMENTATION

The implemented plug-in operates on an animated face model in a Maya scene. The knowledge of the model vertices, which correspond to the standard FDPs, is of crucial importance. These vertices are the plug-in input and, for the purposes of this application, they are previously defined manually.

An important step to begin the operation with is to rotate the model to a specific position, which is to make the model gaze forward (with an orientation to the positive *z* axis) and in an upright position (having the eyes gazing forward, towards the direction that is parallel to the *z* axis). This operation is performed on the model as it appears in the first frame of the scene. Since the model FDPs have been located, this operation is feasible. First, the model is rotated in an appropriate manner to make the chin tip, the top of the head and the lower back point of the head (i.e. FDPs 2.1, 11.4 and 7.1) lie on the *y*-*z* plane. Then, the model is rotated round the *x* axis, in order to make the eyes gaze foreward. This is accomplished by rotating the model by that angle, which layes the eyelid tops and bottoms vertically. The result is subsequently refered to as the neutral model. The FDP coordinates of the neutral model are then taken to the output.

Knowing the FDP parameters of the model renders the calculations of the Facial Animation Parameter Units (FAPUs) possible. These are distances of specific facial features, namely the iris diameter, the eye separation, the mouth-nose separation and the mouth width. An additional unit exists, namely the angular unit, but its value does not depend on the model, which is under consideration. These units are used in order to determine the FAP values that occur along the animation and allow for a model-independent representation of the displacement values, which occur in the model vertex positions along its animation, as demonstrated subsequently.

2.1. Animation Extracion

While the plug-in operates in a specified period of the animation time, it translates and rotates the model back to its neutral position in every frame of the animated sequence, i.e., to the pose the neutral model has. This transform is applied globally on the surface and, therefore does not affect the facial expression of the model. In order to accomplish this procedure three FAP-independent FDPs are chosen, namely FDPs 7.1 (lower back of the head), 11.4 (head top) and 9.15 (point on the nose). These FDPs are not animated by any FAP and are considered to be independent from the model facial expressions and dependent only on the global rotation the model performs. The following steps are taken and resemble the transform that was applied to obtain the neutral model:
1. The model is translated to the position that brings FDP 7.1 to the location it has in the neutral model.
2. Then, the model is rotated around the *y* and *z* axes, so as to bring the remaining two FDPs to the *y*-*z* plane
3. A rotation around the *x* axis brings FDP 11.4 to the position it has in the neutral model.
The rotations in steps 2 and 3 are made using FDP 7.1 as a center. This operation ensures that the FAPs estimation that follows will be made along the correct correct direction of the FDP displacements.

The displacements of the FDPs are then translated to their corresponding FAP values. The rotation parameters, used for the rotation of the model to the pose of the neutral model, are as well encoded among the FAPs. The remaining FAPs correspond to the displacement of a facial feature along one direction for each parameter, unless they correspond to a feature rotation, which is the case, for instance, for the eyeball movement. Therefore, the FAP values can be estimated by applying some calculations on the corresponding FDP displacement. The FAPUs are taken into account in these calculations. For this reason, the FAP values, in the case of vertex displacements, are estimated by an equation of the following form:

$$FAP(i,t) = \pm \frac{\big(FDP(i,t) - FDP(i,t_0)\big) \times 1024}{FAPU(i)} \tag{1}$$

where $FAP(i,t)$ denotes the estimated i-th FAP value at the t-th frame of the animation, $FDP(i,t)$ is the

displacement of the FDP that corresponds to the i-th FAP along the direction, which the i-th FAP specifies, at the t-th frame of the animation and t_0 is the initial frame when the model is in its neutral position. Finally, $FAPU(i)$ is the FAP unit corresponding to the i-th FAP. As it is derived from (1), an FAP value shows the displacement of the corresponding FDP along the corresponding direction the FAP specifies as compared to its FAPU. The dependency of the FAP value is proportional to the displacement value. Moreover, an FAP value of ± 1024 shows that the FDP vertex displacement is equal to the FAPU value in the FAP's direction.

The result of this procedure for the entire animation is stored in a file, which can be given as an input to any FAP-player. Such a procedure generates the same animation on the model of the player as the one of the model in the Maya platform.

2.2. FAT content generation

Apart from the animation of the model, the model itself is extracted from the scene. Its vertices and their connections are extracted in a simple text file. Additionally, an FAT is created describing the animation of the model, as described subsequently.

The Facial Animation Table gives the appropriate correlations between the FAP values and the displacements of all vertices of the facial model, so that an identical animation to the one of the original model can be extracted from a sequence designed in a 3-d graphics modeler, for example Maya. Since the FAPs are derived from FDP displacements along one dimension, it is assumed in this case that the model vertices move under the influence of the FDP displacements. In other words, the displacement of an arbitrary vertex denotes that a FAP value -i.e. an FDP displacement-, which occurs in the same frame, has caused its movement. It is, therefore, considered that the model vertices move under the influence of the FDPs. The fact that MPEG-4 describes the face animation in terms of FAPs, allows such an assumption to hold.

If only one FAP is activated in the given frame, it is obvious that the vertex displacement must have been generated by this FAP. However, this usually is not the case, since any number of FAPs can be active at any frame of the animation. Therefore, the correct correspondence of every FAP and the list of vertices, which it affects, has to be identified.

During the model animation, potential correlations between FAPs and vertices are established, if a vertex is displaced from its neutral position in the same frame when a FAP is active. However, if a FAP is active in some other frame of the animation, whereas the vertex that was previously considered to depend upon this FAP is not displaced from its neutral position, the correlation between them is resolved. In other words, dependencies are established between a FAP and any vertex of the model, if the considered vertex is displaced in every frame that the FAP is active.

This results into a table of possible dependencies between each FAP and each vertex. However, this table may contain some redundant information, which occurs when a certain FAP occurs simultaneously with the movement of a vertex it is in reality not correlated to. This is due to the fact that these correlations are established only based on the displacements of the model vertices, which occur in the defined animation period. Therefore, if the animation duration is too short or the model is not animated in a manner that enables the aforementioned procedure yield correct decisions, the FAT content may be wrong.

An effective way to filter out the redundant vertices from the previous stage of the procedure, which has been utilized in the plug-in, is based on locality. In most cases, the erroneously accepted vertex-FAP dependencies occur in distant vertices, with respect to the mesh topology. The procedure that is followed in this second stage is based on the vertices connections of the model mesh and the FAP-vertex dependencies derived form the first stage. If a vertex that has been considered to be dependent on a FAP in the first stage is connected to the FDP that corresponds to this FAP, then this dependency is confirmed. Furthermore, if a vertex that has been considered to be dependent on a FAP in the first stage is connected to any vertex that has been confirmed to be dependent on this FAP in the second stage, then this dependency is confirmed as well. This procedure continues until it defines a neighborhood of confirmed vertices around the considered FDP, the limits of which are vertices which were not considered to be dependent on this FAP in the first stage. All other vertex dependencies that were derived in the first stage and lie outside this neighborhood are resolved.

An example of the resulting vertex selection for a Maya model can be seen in Figure 1. The figure shows the vertices, which have been chosen, by applying the previously described stages, for FAP number 19, which represents the vertical displacement of the top left eyelid.

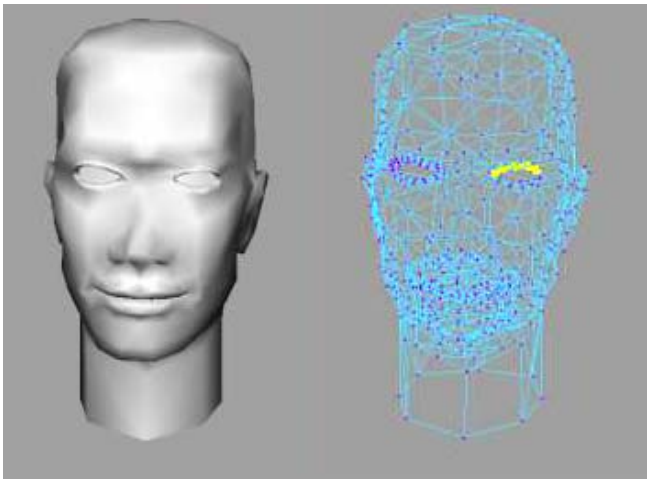

Figure 1: A face model on the Maya platform and the selection of the vertices of the top-left eyelid, which are dependent on FAP 19.

Another issue, which should be treated so as to extract all the necessary information for the creation of the FAT, is to determine how the previously derived dependent vertices are relocated under the influence of the value of their corresponding FAP. According to the standard, the FAT contains the dependencies of any vertex on the FAPs and on their values. There are two sorts of nodes that specify this dependency, the *Transform* node or the *IndexedFaceSet* node. The former node specifies that the vertex reacts to the considered FAP value according to a rotation or scale transform. The implemented plug-in uses only the *IndexedFaceSet* node form. In this case, a number of intervals of the FAP values is specified by the standard for every FAP that affects the movement of a set of vertices. For every such vertex a vector is provided that determines its displacement along each of the FAP value intervals (see [2] for more details on the standard).

Therefore, the plug-in tracks the displacement of all the vertices of the mesh along the animation. As long as the dependency of each vertex on certain FAPs has been established, the dependency of the vertex displacement vector on the FAP values has to be found. For this reason, the value of each vertex displacement is associated with values among the range of values of the considered FAP along the animation period, in which maximum and minimum vertex displacement values occur along any direction. These associated pairs of values, i.e. FAP value and vertex displacement, are stored for those FAP values that were derived from the aforementioned criterion. Since the vertex position is known for these FAP values, the estimation of the displacement vectors is straightforward.

3. FAT LOADER - FAP PLAYER APPLICATION

Regarding the domain of face animation compliant to the MPEG-4 standard, several applications have been implemented. Most of them are able to execute an FAP sequence to animate a predefined model. Few of them, however, allow another face model and its animation table to be loaded. The application which was developed in this case was based on an FAP animator, which was implemented for the European Project MoMuSys [6].

The basic idea was to enable the application to load an arbitrary facial model. This means the model vertex displacements had to be redetermined dynamically by the FAT input, so as to allow the retrieved data to control the model animation. The application operates by receiving a sequence of FAP values in the input and translating them in vertex movements, according to the specifications of the model FAT. A frame from the animation on the Maya platform can be seen in Figure 2 together with its corresponding frame, as seen after having loaded the model and its FAT on the MPEG-4 player and having executed the same FAP sequence.

4. CONCLUSION

The Maya plug-in, the development of which has been presented, is a fast method for extracting MPEG-4 content from an animated face model in Maya. The extracted data define the model with its animation table and include its animation. The MPEG-4 player application running on these data demonstrates that the resulting animation is successfully reproduced without resorting to the Maya platform, only by using the MPEG-4 datastreams. Furthermore, a good quality

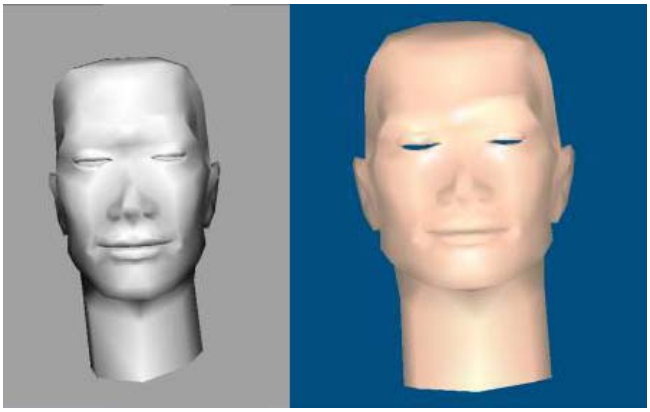

Figure 2: A snapshot from the Maya model's animation and its reproduction by the developed MPEG-4 face animator.

of data compression is achieved, since it encodes the model with its animation tables and permits the animation to be executed by applying a limited list of parameters for each frame to the model.

5. REFERENCES

[1] Moving Pictures Experts Group *Information technology - Generic coding of audio-visual objects - Part 2: Visual*, ISO/IEC JTC 1/SC 29/WG 11 N 2502, Atlantic City, October 1998.

[2] Moving Pictures Experts Group *FBA Core Experiments*, ISO/IEC JTC1/SC29/WG11/N1672, Bristol meeting, April 1997.

[3] S.Garchery and N.Magnenat Thalmann, *Designing MPEG-4 Facial Animation Tables for Web Applications*, Multimedia Modeling 2001, Amsterdam, pp 39-59, May, 2001.

[4] Ostermann Joern and Erich Haratsch, *An Animation Definition Interface: Rapid Design of MPEG-4 Compliant Animated Faces and Bodies*, International Workshop on synthetic - natural hybrid coding and three dimensionalimaging, Rhodes, Greece, Spetember, 1997.

[5] Lavagetto Fabio and Pockaj Roberto, *The Facial Animation Engine: Towards a High-Level Interface for the Design of MPEG-4 Compliant Animated Faces*, IEEE Trans. on Circuits and Systems for Video Technology, 9(2):277-289, March 1999.

[6] *MPEG-4 Facial Animation System, Version 3.3.1 - Gabriel Abrantes*, Developed in the context of the European Project ACTS MoMuSys (c) 97-98, Instituto Superior Tecnico