EURASIP Journal on Applied Signal Processing 2002:10, 1005–1020

Three-Dimensional Facial Adaptation for MPEG-4 Talking Heads

Nikos Grammalidis
Informatics and Telematics Institute, Centre for Research and Technology Hellas, 1st Km Thermi-Panorama Road, Thessaloniki 57001, Greece
Email: ngramm@iti.gr

Nikos Sarris
Information Processing Laboratory, Electrical and Computer Engineering Department, Aristotle University of Thessaloniki, Thessaloniki 54006, Greece
Email: sarris@iti.gr

Fani Deligianni
Informatics and Telematics Institute, Centre for Research and Technology Hellas, 1st Km Thermi-Panorama Road, Thessaloniki 57001, Greece
Email: fd301@doc.ic.ac.uk

Michael G. Strintzis
Informatics and Telematics Institute, Centre for Research and Technology Hellas, 1st Km Thermi-Panorama Road, Thessaloniki 57001, Greece
Email: strintzi@iti.gr

Received 31 August 2001 and in revised form 14 May 2002

This paper studies a new method for three-dimensional (3D) facial model adaptation and its integration into a text-to-speech (TTS) system. The 3D facial adaptation requires a set of two orthogonal views of the user's face with a number of feature points located on both views. Based on the correspondences of the feature points' positions, a generic face model is deformed nonrigidly treating every facial part as a separate entity. A cylindrical texture map is then built from the two image views. The generated head models are compared to corresponding models obtained by the commonly used adaptation method that utilizes 3D radial bases functions. The generated 3D models are integrated into a talking head system, which consists of two distinct parts: a multilingual text-to-speech sub-system and an MPEG-4 compliant facial animation sub-system. Support for the Greek language has been added, while preserving lip and speech synchronization.

Keywords and phrases: MPEG-4, 3D model-based coding, text-to-speech, facial adaptation, talking face.

1. INTRODUCTION

Talking heads apply in a great variety of applications, such as human-machine interfaces (HMI) or virtual reality applications. For example, such systems can be used in information kiosks, virtual shopping agents, news and e-mail reading in the Internet, for pleasant navigation tours in 3D worlds or for interactive educational software. Although cartoon-like characters can be used in some of these applications, synthesizing photo-realistic models of a specific person, although challenging, opens new horizons in realism and personalization.

The system presented here can be used to generate personalized 3D talking heads based on a generic MPEG-4 compliant model by using a new 3D adaptation approach. Our system consists of three modules: the 3D facial adaptation algorithm, the text-to-speech converter and the facial animation engine. These will be outlined separately in the following paragraphs and in detail in the following sections.

Several methods have been proposed in the literature for the *deformation of a generic 3D face model*: in [1, 2, 3, 4] the required geometry of the face is captured by a 3D laser scanner. In [2], the generic geometric model is deformed by applying physical spring forces to predetermined nodes according to the positions of the corresponding features in the target face geometry. In [1], the generic model is built using triangular B-spline patches and the deformation involves the displacement of the spline control points which correspond

to facial features. In [3, 4], the deformation is performed by an interpolation scheme utilizing radial basis functions. In [5], two orthogonal photos of the target face are used and the generic face model is deformed by a 3D geometric transformation, specifically the Dirichlet free form deformation (FFD) [6, 7]. In [8], the required 3D positions of the facial features are estimated from a series of captured image frames of the target face and the generic model is transformed by applying an interpolation function based on radial basis functions. Information from one view of the target face is utilized in [9] to measure the face, eyes, and mouth dimensions which are used to adapt a simple 3D face model by rigidly transforming the whole face and locally correcting the position and orientation of the face, eyes, and mouth. Geometric assumptions have to be made however, as the 3D characteristics of the features cannot be totally deduced from only one view.

Our approach differs from those of all above methods in that it treats the facial model as a collection of facial parts, which are allowed to deform according to separate affine transformations. This is a simple method which does not require the use of any specialized equipment (as a 3D laser scanner) and is effective for face characterization because the physiological differences in characteristics between faces are based on precisely such local variations, that is, a person may have a longer or shorter nose, narrower eyes, and so forth. Thus, in the proposed method after rigidly transforming the whole model so that it is aligned and scaled according to the target face, each facial part is stretched, rotated, and translated separately and in the optimal way (by minimization of a distance cost function) to fit to the required corresponding target facial part. The model nodes between different parts are interpolated to provide a natural smooth transition from one facial part to the other.

The available front and profile views of the face are then combined using the knowledge of the human face symmetry to generate a texture map containing the texture information for the entire head. The texture value stored to the texture map is produced as an average of the corresponding color values obtained by projecting the corresponding 3D point to the available images.

The generated 3D models are then integrated into a talking head system which consists of two distinct parts: a multilingual *text-to-speech* system and a *facial animation* engine based on MPEG-4 facial animation parameters (FAPs).

A text-to-speech synthesizer is a computer-based system that should be able to read any text aloud [10]. A talking head combines the TTS engine with a facial animation engine able to naturally reproduce a person's facial expressions while he is reading a specific text. While various commercial [11, 12] and research [13, 14] talking head systems are already available, the MPEG-4 standard provides new kinds of standardization to the field. The synthetic-natural hybrid coding (SNHC) part of MPEG-4 describes all possible facial movements and expressions using 68 parameters, referred as facial animation parameters (FAPs). Although MPEG-4 supports TTS synthesis by providing an interface to a proprietary text-to-speech synthesizer (TTSI), the synchronization of the FAP synthesizer with the TTS system, which is usually an asynchronous source, is generally difficult [15]. As in [15], we have solved the problem by converting the phonemes (and facial expressions that input by the user as special characters—smileys along with their duration) to FAPs. Since the durations of the phonemes are available both the TTS and the facial animation engine, lip and speech synchronization is always achieved.

Many research or commercial TTS synthesizers are currently available. We have chosen to base our system to the multilingual speech synthesizer of the MBROLA project [16]. This synthesizer needs as input a list of phonemes, together with prosodic information (duration of phonemes and a piecewise linear description of pitch), and produces speech samples on 16 bits (linear), at the sampling frequency of the diphone database used. Phoneme transcription is achieved by a rule-based system for the Greek language or by a decision-tree-based system that is trained using a pronunciation dictionary for the English language. Durations and prosody information are synthetically generated using techniques similar to those developed for the MBRDICO project [17].

The paper is organized as follows: in Section 2, the face model adaptation method is described, while in Section 3 the two parts of the talking head system are presented. Generated 3D models are presented in Section 4 and compared to the same models obtained using the radial basis function (RBF) interpolation approach. Finally, conclusions are drawn in Section 5.

2. 3D FACIAL MODEL ADAPTATION

In this section, we propose a method for the adaptation of a generic 3D face model to a face captured from two orthogonal views: a front and a profile. A number of feature points have to be located on both (2D) views in order to deduce the 3D positions of the head we wish to model. Several methods exist for the automatic extraction of facial feature points as seen in [18, 19, 20], although no method has yet been reported to be 100% accurate and insensitive to lighting and background conditions. Thus, in the presented system, previous work, detailed in [21], has been exploited to locate automatically the positions of the facial features but a manual user interactive tool has also been developed to allow for corrections in the calculated positions. This was necessary because the feature points in the two views need to be exactly positioned, as minor errors will result in unnatural looking 3D models.

The rest of this section outlines the proposed facial adaptation algorithm, which follows the following steps.

(1) Calculation of the 3D positions of the located feature points in the two views.
(2) Rigid transformation of the generic 3D model so that it follows the orientation and scale of the required face.
(3) Nonrigid transformation of the generic 3D model so that the model nodes corresponding to the features are displaced to locations as close as possible to the

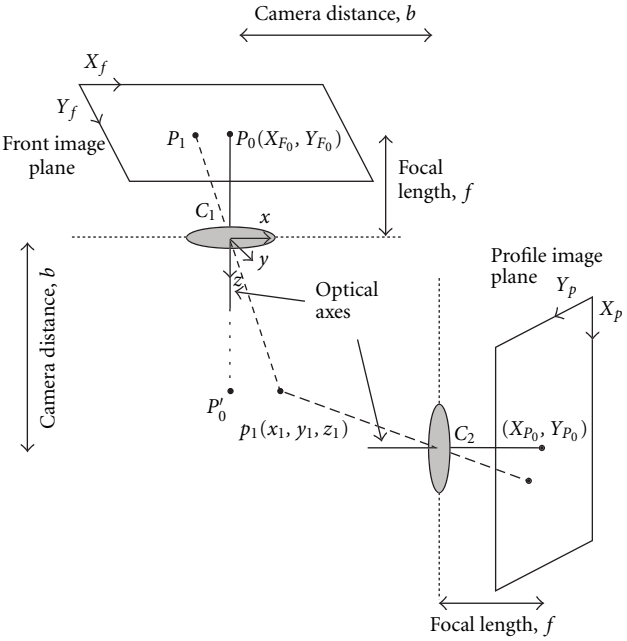


Figure 1: Image acquisition layout.

required 3D positions, while the rest of the model nodes are displaced so that the natural characteristics of the human face are preserved.

2.1. Calculation of the 3D positions of the features

Having located the set of characteristic feature points in both views, the calculation of their 3D coordinates is carried out in the following way.

We assume the use of a perspective projection camera system, which is shown in Figure 1. The 3D points with coordinates (x, y, z) are projected on two image planes, the front and the profile, with perspective rays passing through the two corresponding projection centers C_1 and C_2, which lie within the physical camera and are at a distance b from each other. The focal length f is the distance of the image planes from the corresponding centers of projection, while the point where the optical axis of the camera intersects the image plane is called the principal point. For the frontal image plane the principal point coordinates are (X_{F_0}, Y_{F_0}) and for the profile (X_{P_0}, Y_{P_0}). In our experiments, these were fixed to the centers of the corresponding projection images. Finally, X_F and Y_F are the projections on the frontal view, while X_P and Y_P are the projections on the profile view of the person's face.

The projection coordinates for the frontal image can be computed from the similar triangles $P_1C_1P_0$ and $p_1C_1P'_0$, shown in Figure 1. In the same way, the projection coordinates for the profile image may also be determined

$$\begin{aligned} X_F &= f\frac{-x}{z} + X_{F_0}, & X_P &= f\frac{b-z}{b-x} + X_{P_0}, \\ Y_F &= f\frac{-y}{z} + Y_{F_0}, & Y_P &= f\frac{-y}{b-x} + Y_{P_0}. \end{aligned} \tag{1}$$

This system gives for every feature point four equations with three unknown (the 3D position of the feature point (x, y, z)). This is solved by least squares methods, detailed in the appendix, to provide the 3D position (x, y, z) of every feature point, given the projections of the feature points on the frontal (X_F, Y_F) and profile view (X_P, Y_P).

2.2. Rigid adaptation

Having calculated the positions of the feature points in 3D, a generic 3D facial model needs to be deformed so that its corresponding feature nodes are displaced as close as possible to these required positions, while preserving the natural smoothness and geometry of the human face.

The first part of the deformation involves a rigid transformation of the model. Thus, the model is rotated and translated to match the pose of the real face. The rotation and translation transformations are calculated by a slight modification of the *spatial resection* problem in photogrammetry [22]. Specifically, the relation between the initial and the transformed model is given by

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \mathbf{R} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} + \mathbf{T}, \tag{2}$$

where $(x_0, y_{0,}, z_0)$ are the coordinates of a model node at its initial position, (x, y, z) are the coordinates of the transformed node, $\mathbf{R}$ is a 3×3 rotation matrix, and $\mathbf{T}$ is a 3D translation vector $[T_x, T_y, T_z]^T$. We will assume a left-hand coordinate system, and let ω be the angle of rotation around the x-axis of the camera reference frame, φ the angle of rotation around the y-axis, and κ the angle of rotation around the z-axis. The κ, φ, and ω angles are often called *Euler angles*.

Then the rotation matrix $\mathbf{R}$ will be given by [22]

$$\mathbf{R} = \mathbf{R}(\kappa)\mathbf{R}(\phi)\mathbf{R}(\omega), \tag{3}$$

where

$$\mathbf{R}(\omega) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\omega & \sin\omega \\ 0 & -\sin\omega & \cos\omega \end{bmatrix},$$
$$\mathbf{R}(\phi) = \begin{bmatrix} \cos\phi & 0 & -\sin\phi \\ 0 & 1 & 0 \\ \sin\phi & 0 & \cos\phi \end{bmatrix}, \tag{4}$$
$$\mathbf{R}(\kappa) = \begin{bmatrix} \cos\kappa & \sin\kappa & 0 \\ -\sin\kappa & \cos\kappa & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Our purpose is the minimization of the following sum of squares by appropriate selection of the 6 unknown parameters (3 translation coefficients: T_x, T_y, T_z and 3 rotation angles κ, φ, ω):

$$\sum_{i=1}^{N} \left(\begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} - \begin{bmatrix} x_i' \\ y_i' \\ z_i' \end{bmatrix} \right)^2, \tag{5}$$

where N is the number of feature points located in the front and profile image frames, (x_i', y_i', z_i') are their required positions in the 3D space calculated as described in Section 2.1 and (x_i, y_i, z_i) are the positions of the model 3D nodes after the transformation (2). This minimization is accomplished by use of the nonlinear Levenberg-Marquadt algorithm [23, 24, 25]. A number of iterations of the above method are used to determine the values of the Euler angles corresponding to the rotation matrix $\mathbf{R}$ and the three components of the translation vector $\mathbf{T}$.

2.3. Nonrigid adaptation

The rigid transformation may align the generic model with the required face and scale it to meet the total face dimensions. However, the local physiology of the face cannot be altered in this way. Thus, in the second step of the adaptation process the model is transformed in a nonrigid way aiming to further displace the feature nodes bringing them as close as possible to their exact calculated positions, while the facial parts retain their natural characteristics. To perform this adaptation, the generic model[1] (shown in Figure 2) is split into separate face parts (left eye, right eye, mouth, etc.) and a rigid adaptation is performed on every part separately. This means that every 3D face part is rotated, translated, and stretched so as to minimize the distances of the feature points belonging to that face part from their required positions. This is accomplished with the following transformations.

[1]The generic model used in our experiments was adapted from the model provided by Instituto Superior Tecnico (IST), Universidade Tecnica de Lisboa, Portugal [26].

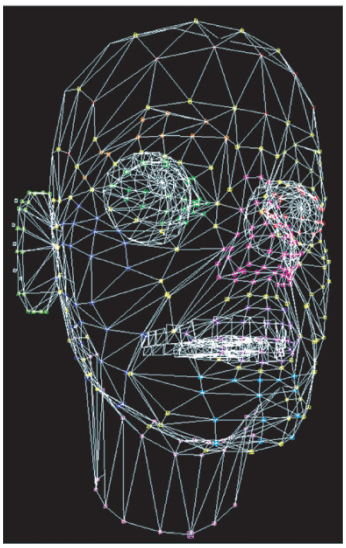


Figure 2: Generic 3D head model.

Centering at the origin

The center of the face part is found as the 3D center of the feature nodes contained and the whole part is translated towards the origin so that this center is translated on the origin. The same is done with the set of required feature positions (i.e., their center is found and they are translated towards the origin in the same manner).

Alignment

The face part is rotated around the origin so that three lines connecting three pairs of feature nodes are aligned with the corresponding lines connecting the required feature positions. This is accomplished by minimizing the differences in the gradients of these lines. For one pair of nodes at (x_1, y_1, z_1) and (x_2, y_2, z_2), with required positions at (x_1', y_1', z_1') and (x_2', y_2', z_2') this would involve finding the 3×3 rotation matrix $\mathbf{R}$ which would perform the following minimization:

$$\min \left\| \left(\text{grad}_y - \text{grad}_y' \right)^2 + \left(\text{grad}_z - \text{grad}_z' \right)^2 \right\|, \tag{6}$$

where

$$\text{grad}_y' = \frac{y_2' - y_1'}{x_2' - x_1'}, \qquad \text{grad}_z' = \frac{z_2' - z_1'}{x_2' - x_1'},$$
$$\text{grad}_y = \frac{y_{R_2} - y_{R_1}}{x_{R_2} - x_{R_1}}, \qquad \text{grad}_z = \frac{z_{R_2} - z_{R_1}}{x_{R_2} - x_{R_1}}, \tag{7}$$
$$\begin{bmatrix} x_{R_i} \\ y_{R_i} \\ z_{R_i} \end{bmatrix} = \mathbf{R} \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}, \quad i = 1, 2.$$

Stretching

The face part is scaled around the origin with different scale factors for every axis (s_x, s_y, s_z) so that the distances of the transformed nodes from their required positions are

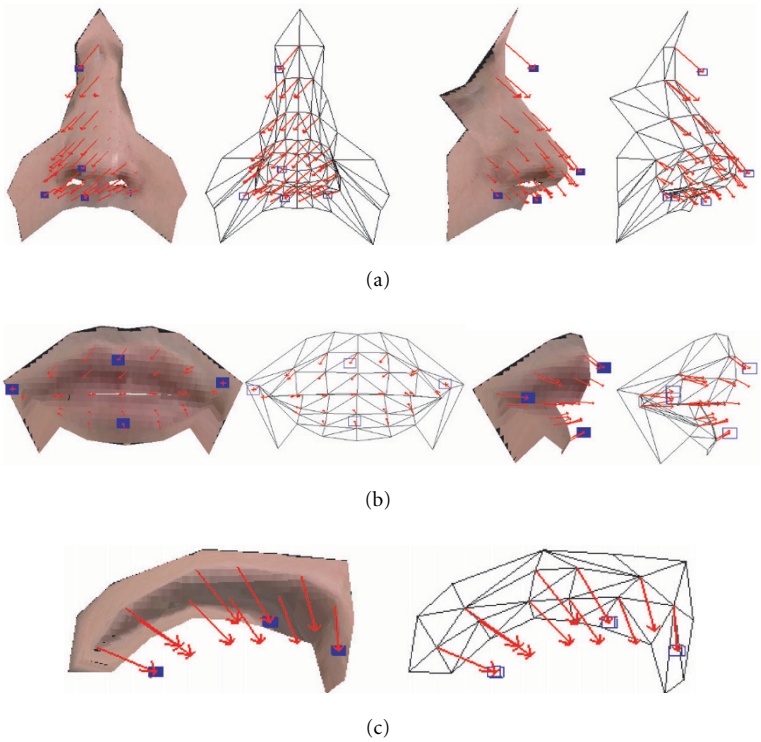


FIGURE 3: Local adaptation of the nose (a), mouth (b), and right eyebrow (c). The wireframe (both textured and transparent) is drawn at the rigidly deformed position, while the arrows represent the nonrigid deformation and the cubes show the desired position of the feature points.

minimized. Thus, if $(x_{R_i}, y_{R_i}, z_{R_i})$ are the coordinates of the feature nodes after the rotation and (x'_i, y'_i, z'_i) their required positions this would involve finding the 3×3 scaling matrix $\mathbf{S}$ which would minimize the following set of equations:

$$\sum_{i=1}^{N} \left(\begin{bmatrix} x'_i \\ y'_i \\ z'_i \end{bmatrix} - \begin{bmatrix} x_{S_i} \\ y_{S_i} \\ z_{S_i} \end{bmatrix} \right)^2, \tag{8}$$

where

$$\begin{bmatrix} x_{S_i} \\ y_{S_i} \\ z_{S_i} \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{bmatrix} \begin{bmatrix} x_{R_i} \\ y_{R_i} \\ z_{R_i} \end{bmatrix}, \quad i = 1, \ldots, N. \tag{9}$$

Translation

After the stretching transformation the face part is translated back towards its original position by adding the position vector of the face part center calculated and subtracted in step 1.

Results of these steps for three facial parts are shown in Figure 3, where the model is drawn at its rigidly deformed position (x_i, y_i, z_i), the arrows show the final nonrigid displacement vectors $(x_{S_i}, y_{S_i}, z_{S_i})$ and the cubes represent the desired positions for the feature nodes. It is evident that the desired positions of the feature nodes are achieved after the nonrigid deformation (arithmetic results are given in Section 5), while the neighboring (to the features) nodes exhibit similar motion vectors, thus accomplishing a smooth deformed surface. Visual results of the complete deformed head are given at the end of this section as well as in Section 5.

Between all facial parts one series of nodes is defined as border nodes. These nodes may belong to more than one facial part and thus, their deformed positions are found by linear interpolation of their candidate positions according to every facial part they may belong to. This is done to assure that a smooth transition is achieved from one facial part to the other without neglecting the original surface curvature at those points in the generic model.

Thus, the final deformed model adapts to the particular characteristics implied by the feature points (e.g., bigger nose or smaller eyes) keeping the generic characteristics of a human face (smoothness of the skin and symmetry of the face). Figure 4 shows the results of the rigid and nonrigid adaptation procedures, by projection of the rigidly and nonrigidly deformed model on the frontal and profile image views of the target face.

Finally, in order to be able to visualise a textured head model from any viewpoint, it is necessary to blend the frontal and profile view images into a single texture map. As in other approaches [27], we define a mapping between the 3D coordinates on the face model and the 2D positions on the texture map using a cylindrical projection. The cylindrical projection is used so that the texture map contains all available texture

Figure 4: Rigid (upper) and nonrigid (lower) adaptation of the face model on the front and profile views.

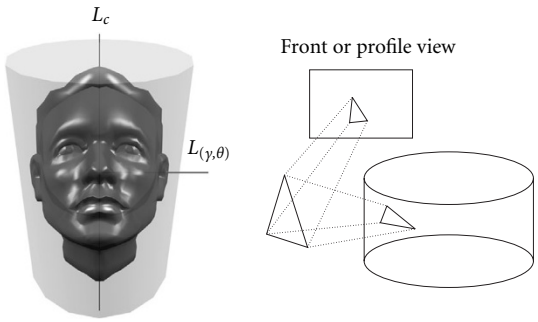


Figure 5: Projecting the 3D model triangles on the cylindrical surface in order to generate the texture map.

of the head model from any given viewing angle. To achieve this the central axis, L_c, of the cylinder must lie within the model. In our implementation this was accomplished by selecting the cylinder axis to be the same as the 3D ellipsoid major axis. The mapping of 3D coordinates (x, y, z) to cylindrical coordinates (y_c, θ_c) (Figure 5) is given by

$$\tan\theta_c = \frac{z - z_0}{x - x_0}, \qquad y_c = y, \tag{10}$$

where (x_0, y_0, z_0) is the 3D head model centroid. For each position on the cylindrical surface $C(y_c, \theta_c)$, the corresponding position on a wireframe triangle is found as the intersection of the perpendicular to the cylinder axis semi-line, $L_{(y,\theta)}$, starting from this axis and passing through C, and the wireframe, as shown in Figure 5.

We assume that this semi-line intersects with the wireframe only once, which is to be expected, given the physiology of the human face. The texture value T which is stored to the texture map at this position C, is produced as a weighted average of the corresponding colour values obtained by projecting the corresponding 3D point to the available images

$$T = kI_{\text{front}} + (1 - k)I_{\text{profile}}, \tag{11}$$

where $k = 1$ if the 3D point is projected only on the front view and $k = 0$ if the 3D point is projected only in the profile view. If the 3D point is projected on both the front and the profile views, k is selected to be proportional to the area of the 2D triangle projections on the front and profile views of the wireframe. This is reasonable because the greater this triangle area is, the better the resolution of the corresponding texture image in the vicinity of this 3D point. More specifically, since the front plane is perpendicular to the z-axis, the area of the projected triangle in the frontal view equals

$$E_{\text{front}} = |n_z|E, \tag{12}$$

where E_{front} is the area of the 2D triangle projection in the front view, $\mathbf{n} = (n_x, n_y, n_z)$ is the normal vector to the 3D triangle and E is the area of the 3D triangle. In a similar way

$$E_{\text{profile}} = |n_x|E. \tag{13}$$

Using (12) and (13), the following equation can be used to calculate k if the triangle is visible in both views:

$$k = E_{\text{front}}/(E_{\text{front}} + E_{\text{profile}}) = |n_z|/(|n_x| + |n_z|). \tag{14}$$

A problem in the generation of the texture map is that the illumination conditions in front and the profile images differ

(a) (b) (c)

Figure 6: (a) Front image, (b) profile image, (c) profile image after equalization.

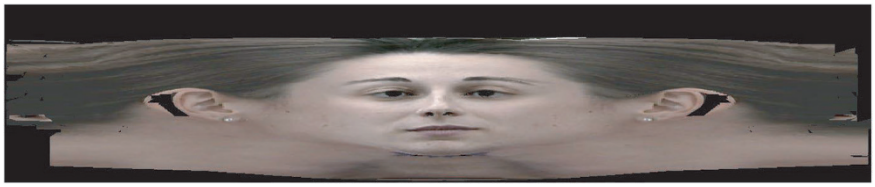


Figure 7: The texture map generated from the front and the profile view.

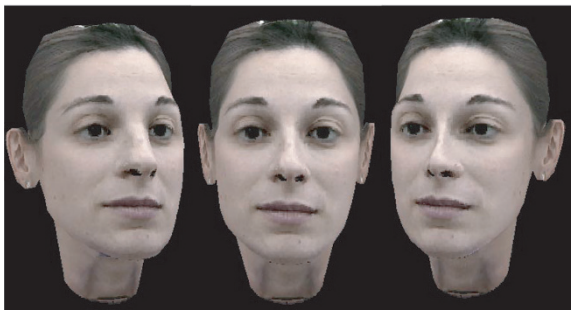


Figure 8: The generated head model, viewed from three different viewpoints.

significantly, thus the combination of both images into one homogeneous texture map becomes a very difficult task.

To solve this problem we use a linear calibration model to correct for systematic luminance differences between the two cameras. More specifically, in the texture generation procedure, we use the following transformation of the profile view:

$$I_{\text{pr}}^{\text{eq}} = AI_{\text{pr}} + B, \tag{15}$$

where the parameters A and B are computed so that the skin region in the calibrated view has similar statistics (mean, variance) as in the front view. The optimal parameters A and B are given by

$$A = \frac{\sigma_{\text{fr}}}{\sigma_{\text{pr}}}, \qquad B = \mu_{\text{fr}} - A\mu_{\text{pr}}. \tag{16}$$

Using the above calibration procedure for the profile view, the generated texture maps are much smoother and the quality of the visualized models is significantly improved, as shown in the example in Figure 6.

The texture map generated from the front and the profile view is shown in Figure 7, while three views of the texture-mapped head model are shown in Figure 8.

3. 3D FACIAL ADAPTATION USING RBF

For comparison, a 3D facial adaptation procedure based on radial basis function (RBF) theory [23] was implemented.

Assuming that n 3D (feature) points y_i, $i = 1, \ldots, n$ are available and that x_i, $i = 1, \ldots, n$ are the corresponding points before adaptation, the following interpolation function is used:

$$f(x) = \sum_{j=1}^{n} w_j \Phi(x - x_j) + \sum_{l=1}^{M} v_l p_l(x), \tag{17}$$

where $\Phi(x) = \phi(\|x\|_2)$ are radial basic functions (RBF) and

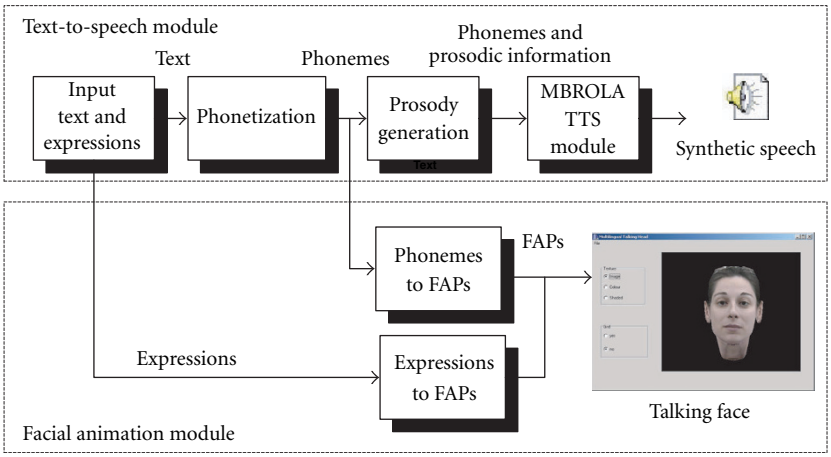


Figure 9: Overview of the system.

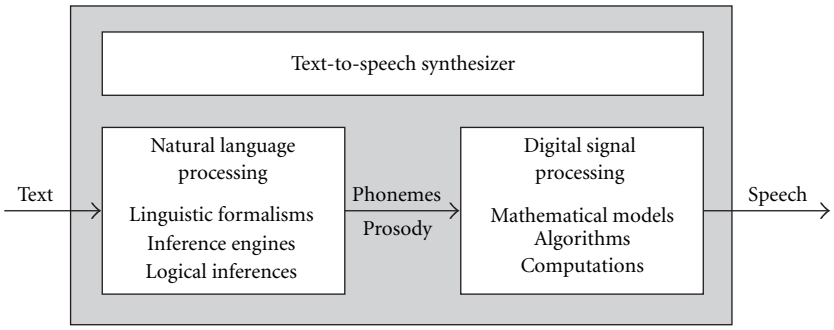


Figure 10: General overview of a TTS synthesizer [10].

the Eucledian norm is calculated in the 3D space, $p_l(x)$ are first-order polynomial terms in the 3D space, used to simulate rigid (rotation, translation, and scale) transformations. Weights $w_j \in \mathbb{R}^3$ and $v_l \in \mathbb{R}^3$ are estimated by solving the following linear system of $n + M$ equations with $n + M$ unknown:

$$\begin{aligned} f(x_i) &= y_i, \quad i = 1, \ldots, n, \\ \sum_{i=1}^{n} w_i p_l(x_i) &= 0, \quad l = 1, \ldots, M. \end{aligned} \tag{18}$$

In our tests we implemented the Wu functions which belong in the family of compactly supported RBF functions [23]

$$\Phi(r) = (1-r)_+^6 (6 + 36r + 82r^2 + 72r^3 + 30r^4 + 5r^5), \tag{19}$$

where $r = r_1/r_{\max}$, and $r_{\max}$ defines the function support.

4. THE TALKING HEAD SYSTEM

A text-to-speech synthesizer is a computer-based system that should be able to read any text aloud [10]. A talking head combines a TTS engine with a facial animation engine able to naturally reproduce a person's facial expressions while he is reading specific text. Our talking face can additionally reproduce human emotions, which are input as special symbols (smileys) within the text. The basic system structure of the talking head system that was developed is illustrated in Figure 9.

The basic modules of the system are the text-to-speech module for speech synthesis and the facial animation module for synthesis of facial expressions during speech. Synchronization between these two modules, resulting to lip-speech synchronization, is guaranteed since the same phonemic description of speech is used as input at both modules. More details about these two modules are provided in the following subsections.

4.1. Text-to-speech module

Figure 10 introduces the functional diagram of a very general TTS synthesizer. For human reading, it comprises a natural language processing module (NLP), capable of producing a phonetic transcription of the text read, together with the desired intonation and rhythm (often termed as prosody), and a digital signal processing module (DSP), which transforms the symbolic information it receives into speech [10]. A number of simplifications are usually used in some processing steps in order to be able to solve the problem in real time with limited memory requirements. The price paid for

such simplifications is often a reduction of the *naturalness* and *color* of the generated synthetic voice.

Many research or commercial TTS synthesizers are currently available. We have chosen to base our system to the multilingual speech synthesizer of the MBROLA project [15]. The multiband resynthesis pitch synchronous overlap add (MBROLA) algorithm is an improved version of the time domain-pitch synchronous overlap (TD-PSOLA) technique. Both algorithms produce speech by concatenating elementary speech units called diphones. Unlike TD-PSOLA, MBROLA makes use of a diphone database specially adapted to the requirements of the synthesizer, and obtained after a complex processing (actually, a hybrid harmonic/stochastic analysis-synthesis) of an original diphone database (i.e., a database composed of speech samples). The resulting synthesis technique takes advantage of the flexibility of parametric speech models while keeping the computational simplicity of time-domain synthesizers [10]. The MBROLA system actually provides only the DSP component of a speech synthesizer (see Figure 10) and is based on the concatenation of diphones. It uses a list of phonemes as input, together with prosodic information (duration of phonemes and a piecewise linear description of pitch), and produces speech samples on 16 bits (linear), at the sampling frequency of the diphone database used. The advantages of using this system are

(i) it is a noncommercial product of a very successful EU project;
(ii) it allows us to focus on the more important natural language processing module to improve the naturalness and to add color in the synthesized voices;
(iii) it allows designing truly multilingual TTS systems by supporting already more than 24 languages, while more can be supported in the future;
(iv) it is an easy to use Windows-based application.

For the results of this paper, we have used the English MBROLA diphone database and the Greek diphone MBROLA database that has been built by Bletsas and Sergiadis [28]. However, we have defined a set of rules to achieve phonemic analysis and used simple rules to add prosody information. The design of the natural language processing module differs for each language and consists of the following stages.

A text pre-processor is used to identify different words and to identify numbers, abbreviations, acronyms, and idiomatics and transform them into full text when needed. A lookup dictionary is used to directly provide the phonetic transcription of such words or symbols.

A phonetisation module is responsible for the automatic determination of the phonetic transcription of regular words. This module is used to identify the phonemes comprising each word and estimate their duration. While the mean duration of each phoneme is a user-defined parameter mathematical models are used to modify the duration of each phoneme in order to achieve more natural speech synthesis.

An (optional) synthetic prosody generation module, which adds additional properties to the final speech signal that are related to audible changes in pitch, loudness, syllable length. For example, adding certain prosody to a specific syllable of a sentence can simulate focus on this syllable. Another example is the accent in the word, which can be simulated by prosody generation as well as by increasing the duration of the syllable with the accent.

Phonetisation for the English language was achieved using the decision-tree-based phonetisation module provided by the MBRDICO project [17]. The same project provides utilities for automatic training of the decision trees based on a pronunciation dictionary for English language [29]. However, we have observed that phonetisation rules in the Greek language are much simpler than the English language and that a rule-based phonetisation module could be designed instead. More specifically, rules are made up of four parts:

(1) The left context.
(2) The text to match.
(3) The right context.
(4) The phonemes to substitute for the matched text.

For each unmatched letter in a word, each rule is examined, in turn, when the *text to match* starts with the letter in the word. If the text to match is found and the right and left context patterns also match, the phonemes for that rule are output and the procedure is repeated for the next unmatched letter. This algorithm is faster and also solves the problem of creating or finding an existing Greek phonetisation dictionary to train the decision tree. We have written rules to cover all possible phoneme combinations in the Greek language and the extensive tests show that results are very satisfactory.

A lookup table containing the mean duration of each phoneme is used to provide an initial value of its duration. Then, according to the position of the corresponding syllable inside the word and the possible presence of accent, its duration is accordingly modified, using rules similar to those implemented by the MBRDICO project [17]. More specifically, in the Greek language, an accent-mark shows the accent of the word, so the corresponding prosody can be easily added, that is, the duration of the phoneme under the accent-mark is increased. Furthermore, phoneme durations are inversely proportional to the number of syllables in a word and at the end of the word the tone must fall and thus the pitch reduced by 60 Hz.

All the generated synthetic speech parameters (i.e., list of phonemes along with their durations and prosodic information) are written in a text file and are used by MBROLA to actually synthesize the speech, which is then either directly output from the speakers or stored in the disk as a sound (.wav) file.

4.2. Facial animation module

The facial animation part system is analogous to the facial animation system described in [26] and has been extended for animating Greek phonemes. MPEG-4 uses a set of facial animation parameters (FAPs) to describe each possible facial

Table 1: Facial expressions and corresponding symbols in the TTS system.

Facial expressions	Symbol
Joy	\ :)
Sadness	\ : |
Surprise	\ : O
Anger	\ : (
Disgust	\ :;
Fear	\ : o

Table 2: FPD method deformation error, measured in the projected positions of the feature points.

Test image	MSE in feature points (pels)
Mary	0.279
Nikos	0.252
Dimitris	0.334
NGramm	0.330
Venny	0.331
Nikolas	0.417

expression. In addition, there are two special high-level FAP's that are used to describe *visemes* and *facial expressions*:

The viseme FAP identifies 14 *visemes*, each corresponding to the facial expression produced when a specific phoneme is articulated. Thus each phoneme is mapped to a viseme (although some phonemes may map to the same viseme). Visemes (as well as expressions) are characterized as *high-level* FAPs, since their values can be expressed in terms of standard (low level) FAPs. The visemes for the Greek phonemes are similar to the visemes for the English phonemes. Thus, we have designed a lookup table that maps Greek visemes to the corresponding English ones, or performs a blending of two visemes in cases that this proves to be necessary. This is supported by MPEG-4, as a Viseme FAP can specify two out of the predefined list of 14 visemes, and a factor to blend between them.

The expression FAP is used to describe emotional face expressions (e.g., joy, sadness, etc.). There are 6 such expressions currently supported by the MPEG-4 standard and our TTS system. We can introduce *facial expressions* by using special symbols in the text, as shown in Table 1. Expression symbols are input as text along with a number, which defines the desired duration of the expression in ms, e.g., \ :)1000. The TTS module produces no output (silences) within the expression duration. As the viseme FAP, the expression FAP can specify two out of the predefined list of six basic expressions and intensity values allowing the blending of the two expressions.

In order to achieve a more natural facial animation the viseme or expression FAP values are attributed to specific time instants, that is, the midtime of their corresponding time interval and used as keyframes. Then an interpolation procedure is used to calculate FAP values corresponding to frames between two consecutive keyframes. For this purpose, a smoothing function is used to smooth out the transitions between two visemes (or expressions). Specifically, the third-order Hermite function is used

$$f(x) = (2x^3 - 3x^2 + 1)a_1 + (-2x^3 + 3x^2)a_2 + (x^3 - 2x^2 + x)g, \tag{20}$$

where a_1 and a_2 are the FAP values corresponding to the two keyframes, $x \in [0, 1]$, and g is the desired gradient at the first keyframe ($x = 0$). This function, also used in [15], provides very satisfactory interpolation results. The same procedure is used for Greek and English TTS.

In addition, facial movements (small head rotations and eye blinking), are periodically performed by inserting specific FAP sequences, along the lip movements, thus adding a higher degree of realism to the talking head.

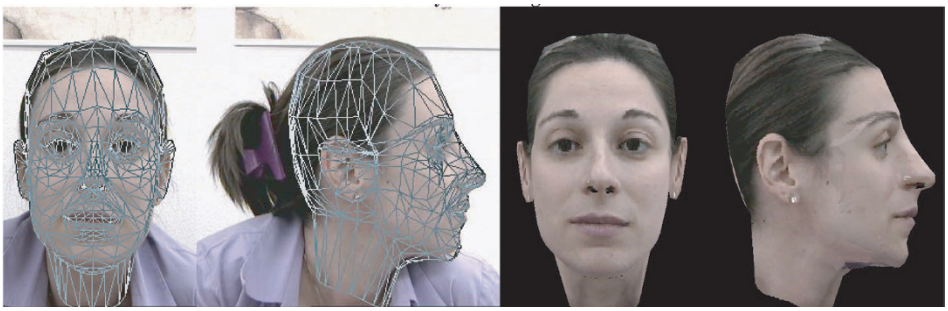
Synchronization between the sound file produced by MBROLA and the FAP file produced by converting the phonemes to FAPs is achieved, since the durations of the phonemes are available to both the TTS and the facial animation engine.

5. EXPERIMENTAL RESULTS

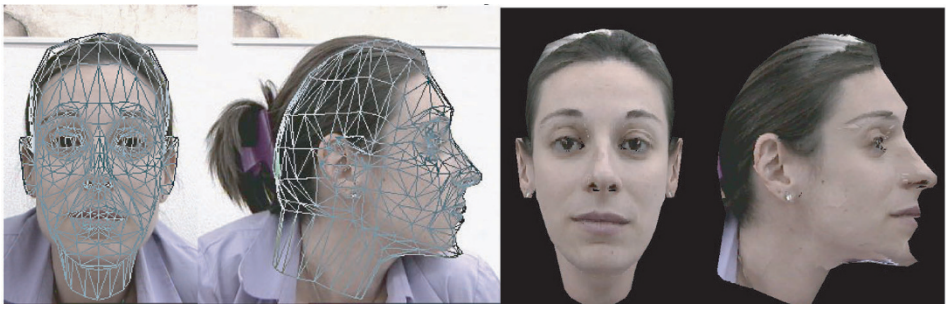
In Figures 11, 12, 13, 14, and 15 we provide visual results for our 3D adaptation method compared to the RBF adaptation method. It is evident that while the RBF adaptation method ensures that the given feature points are displaced at their exact required positions the natural smoothness of the human face is not maintained in all cases. This is achieved by our method (facial parts deformation—FPD) which by operating on every part separately ensures that the initial characteristics of most facial parts are maintained. The proposed method, although not guaranteeing displacement of the feature points at their exact required position, results to a negligible deviation from this requirement, as shown in Table 2. This is illustrated also visually in Figure 18, where an adapted with the FPD method model (this adaptation is illustrated in Figure 17) is animated presenting the seven different emotions provided by the MPEG-4 standard. In Figure 16 we illustrate the provided interface of our viewer together with some frames of the resulting talking head.

6. CONCLUSIONS

We have presented a method for the construction of 3D head models produced by using two orthogonal views (frontal and profile) of a person's face. The positions of feature points in 3D are calculated and are used to deform a generic face model so that the corresponding model nodes are displaced optimally close to the calculated 3D positions, while maintaining the inherent facial characteristics. The two views of the face are used to generate a cylindrical texture map, which is consulted when visualizing the 3D head model. The produced 3D head model is then integrated in a TTS system which combined with a face animation engine produces a 3D talking face capable of speaking the English or Greek language.

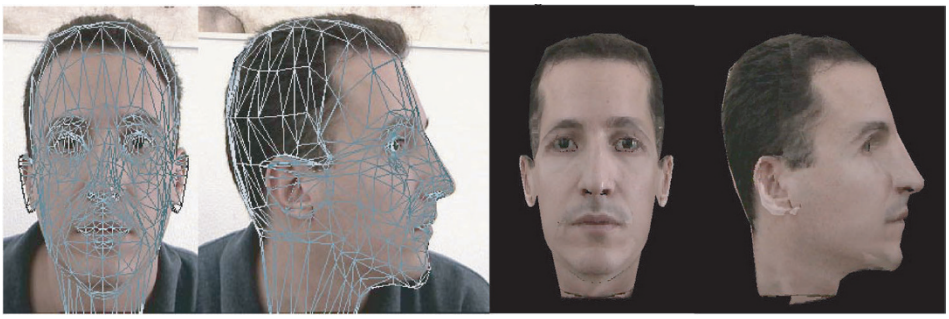


(a) FPD adaptation.

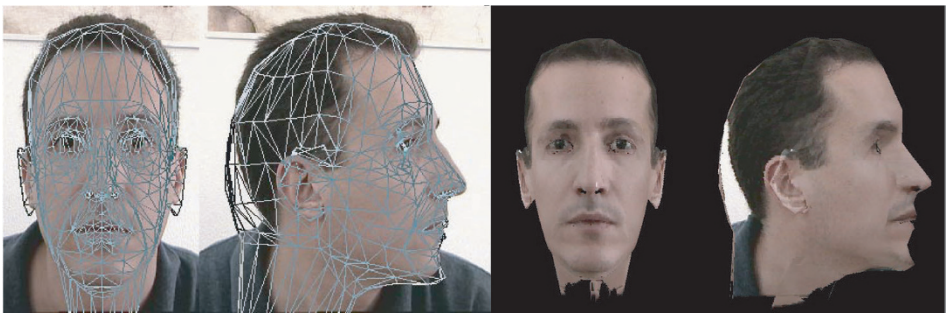


(b) RBF adaptation.

Figure 11: *Mary* test images.

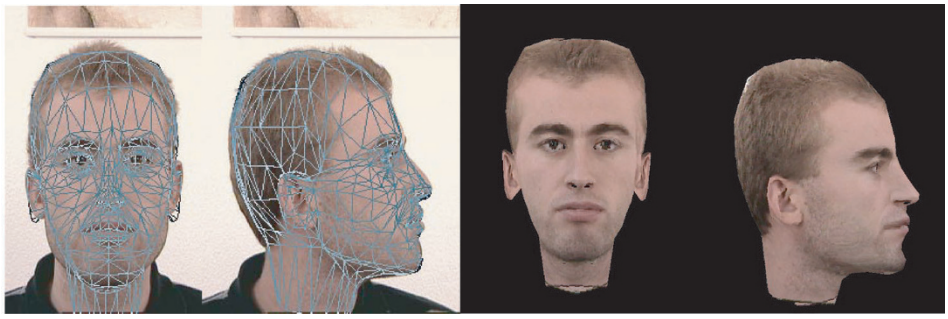


(a) FPD adaptation.

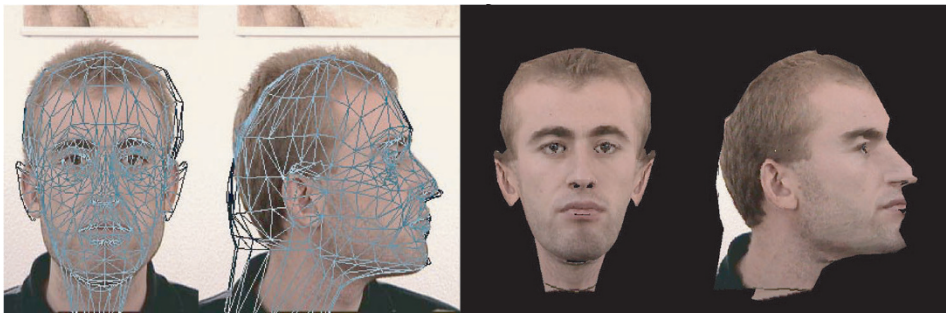


(b) RBF adaptation.

Figure 12: *Nikos* test images.

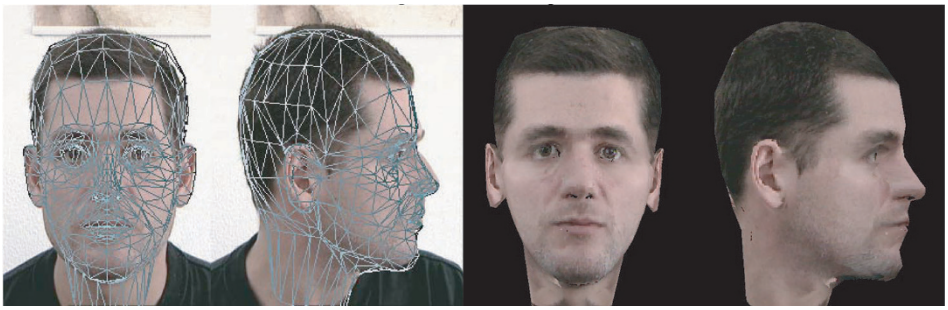


(a) FPD adaptation.

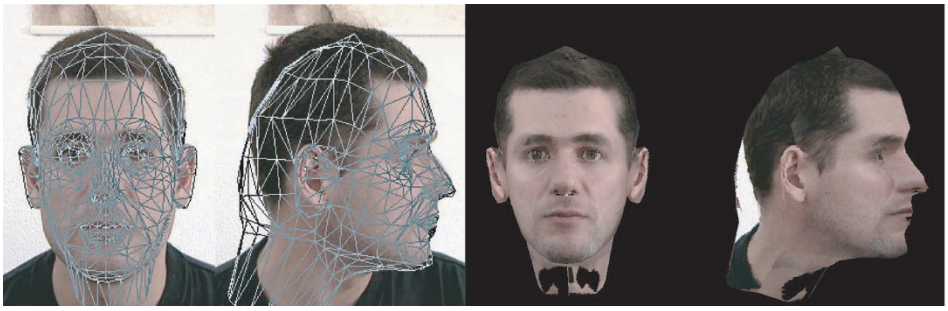


(b) RBF adaptation.

Figure 13: *Dimitris* test images.

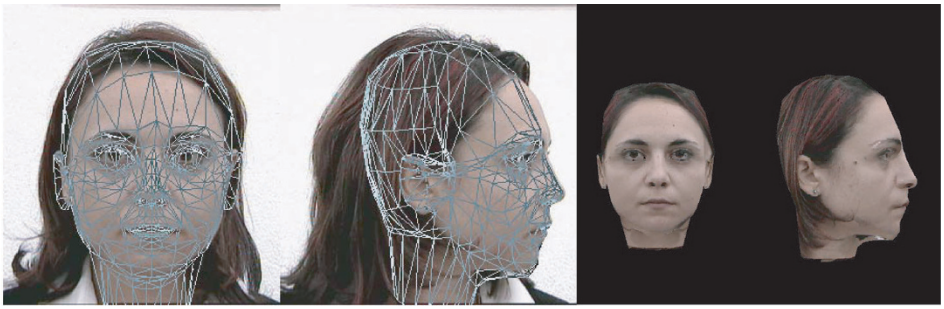


(a) FPD adaptation.

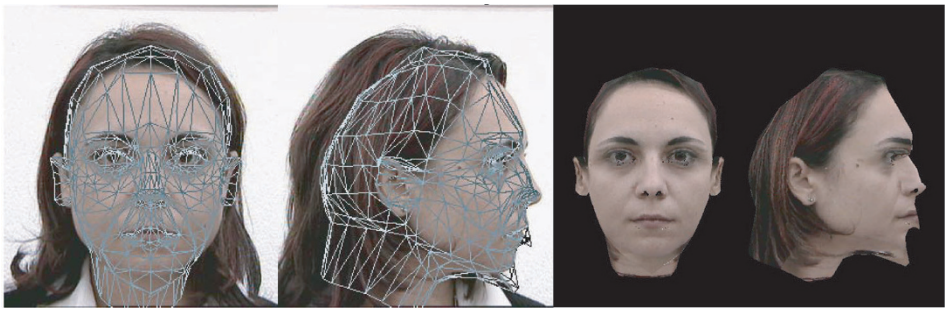


(b) RBF adaptation.

Figure 14: *Ngramm* test images.



(a) FPD adaptation.



(b) RBF adaptation.

Figure 15: *Venny* test images.

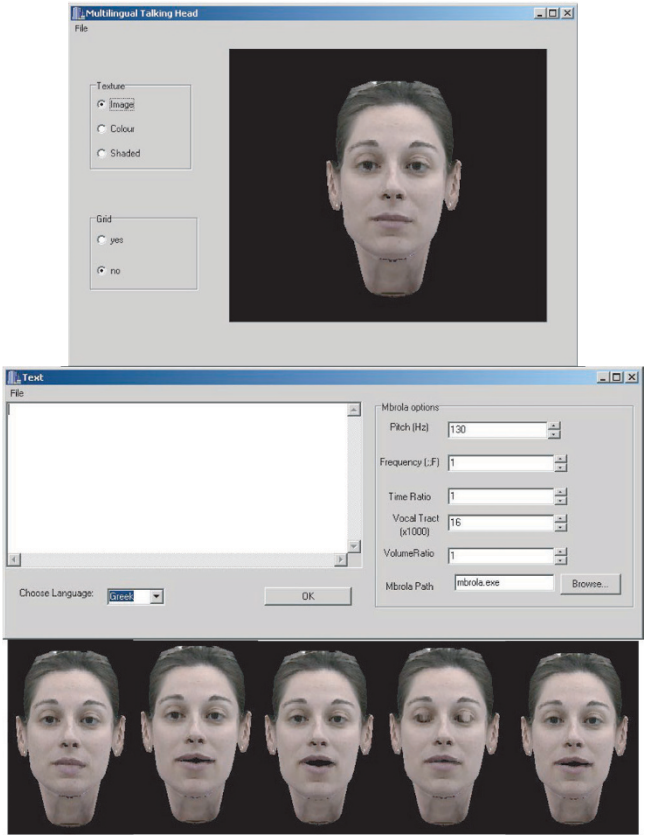


Figure 16: TTS Talking head viewer.

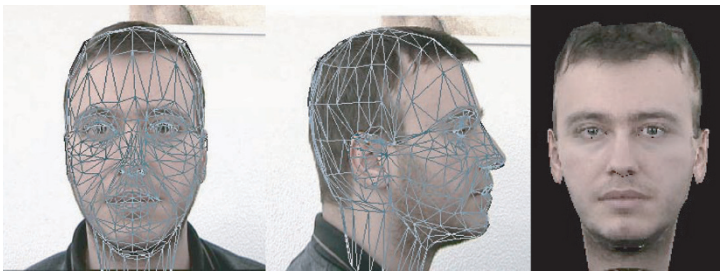


Figure 17: FPD adaptation on *Nikolas* test image.

Figure 18: MPEG-4 emotions on *Nikolas* test image.

APPENDIX

The system of the four equations given in (1) can be rewritten as

$$\begin{aligned} f \cdot x + (X_F - X_{F_0}) \cdot z &= 0, \\ f \cdot y + (Y_F - Y_{F_0}) \cdot z &= 0, \\ -(X_P - X_{P_0}) \cdot x + f \cdot z &= -(X_P - X_{P_0}) \cdot b + b \cdot f, \\ -(Y_P - Y_{P_0}) \cdot x + f \cdot y &= -(Y_P - Y_{P_0}) \cdot b. \end{aligned} \tag{A.1}$$

This system can be given in the form

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{b}, \tag{A.2}$$

where

$$\mathbf{A} = \begin{bmatrix} f & 0 & X_F - X_{F_0} \\ 0 & f & Y_F - Y_{F_0} \\ -(X_P - X_{P_0}) & 0 & f \\ -(Y_P - Y_{P_0}) & f & 0 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 0 \\ 0 \\ -(X_P - X_{P_0})b + f \cdot b \\ -(Y_P - Y_{P_0})b \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}. \tag{A.3}$$

The solution to this system is found by the least squares method by the calculation of

$$\mathbf{x} = (\mathbf{A}^{\mathbf{T}} \cdot \mathbf{A})^{-1} \cdot \mathbf{A}^{\mathbf{T}} \cdot \mathbf{b}. \tag{A.4}$$

REFERENCES

[1] P. Eisert and B. Girod, "Analyzing facial expressions for virtual conferencing," *IEEE Computer Graphics and Applications*, vol. 18, no. 5, pp. 70–78, 1998.

[2] Y. Lee, D. Terzopoulos, and K. Waters, "Realistic modeling for facial animation," in *Computer Graphics (Proc. SIGGRAPH '95)*, pp. 55–62, Los Angeles, Calif, USA, August 1995.

[3] F. Lavagetto and R. Pockaj, "The facial animation engine: toward a high-level interface for the design of MPEG-4 compliant animated faces," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 9, no. 2, pp. 277–289, 1999.

[4] R. Pockaj, M. Costa, F. Lavagetto, and C. Braccini, "MPEG-4 facial animation: an implementation," in *Proc. International Workshop on Synthetic-Natural Hybrid Coding and 3-D Imaging*, pp. 33–36, Santorini, Greece, September 1999.

[5] W.-S. Lee, M. Escher, G. Sannier, and N. Magnenat-Thalmann, "MPEG-4 compatible faces from orthogonal photos," in *International Conference on Computer Animation*, pp. 186–194, Geneva, Switzerland, May 1999.

[6] L. Moccozet and N. Magnenat-Thalmann, "Dirichlet free-form deformations and their application to hand simulation," in *International Conference on Computer Animation*, pp. 93–102, IEEE Computer Society, Geneva, Switzerland, 1997.

[7] T. Sederberg and S. R. Parry, "Free form deformation of solid geometric models," in *Proc. SIGGRAPH '86*, pp. 151–160, Dallas, Tex, USA, August 1986.

[8] F. Pighin, J. Auslander, D. Lischinski, D. Salesin, and R. Szeliski, "Realistic facial animation using image based 3D morphing," Tech. Rep. UW-CSE-97-01-03, University of Washington, Seattle, Wash, USA, May 1997.

[9] L. Zhang, "Automatic adaptation of a face model using action units for semantic coding of videophone sequences," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 8, no. 6, pp. 781–795, 1998.

[10] T. Dutoit, *An Introduction to Text-To-Speech Synthesis*, Kluwer Academic, Dordrecht, Netherlands, 1996.

[11] Digimask—The Face of the Future, http://www.digimask.com.

[12] LifeFX—The face of the Internet, http://www.lifefx.com.

[13] N. M. Brooke, "Computer graphics synthesis of talking faces," in *Talking Machines: Theories, Models, and Designs*, G. Bailly, C. Benot, and T. R. Sawallis, Eds., pp. 505–522, North Holland, Elvsevier, Amsterdam, 1992.

[14] B. DeGraph and M. Wahrman, "Mike, the talking head," *Computer Graphics*, vol. 11, pp. 15–17, 1988.

[15] J. Ostermann, M. Beutnagel, A. Fischer, and Y. Wang, "Integration of talking heads and text-to-speech synthesizers for visual TTS," in *Proc. 5th International Conference on Spoken Language Processing (ICSLP-98)*, Sydney, Australia, November 1998.

[16] MBROLA project, http://tcts.fpms.ac.be/synthesis/mbrola.html.

[17] MBRDICO project, http://tcts.fpms.ac.be/synthesis/mbrdico/.

[18] A. Al-Qayedi and A. F. Clark, "An algorithm for face and facial-feature location based on grey-scale information and facial geometry," in *Proc. IEEE International Conference on Image Processing and its Applications*, vol. 2, pp. 625–629, Manchester, UK, July 1999.

[19] T. C. Chang and T. S. Huang, "Facial feature extraction from color images," in *12th International Conference on Pattern Recognition*, vol. 2, pp. 39–43, Israel, October 1994.

[20] M. Reinders, P. J. L. van Beek, B. Sankur, and J. C. A. van der Lubbe, "Facial feature localization and adaptation of a generic face model for model based-coding," *Signal Processing: Image Communication*, vol. 7, no. 1, pp. 57–74, 1995.

[21] N. Sarris and M. G. Strintzis, "Construction of a 3D facial model from two image views," in *IEEE International Symposium on Intelligent Signal Processing and Communication Systems*, Honolulu, Hawaii, November 2000.

[22] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*, vol. II, Addison-Wesley, Reading, Mass, USA, 1993.

[23] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quart. Appl. Math.*, vol. 2, no. 2, pp. 164–168, 1944.

[24] D. W. Marquardt, "An algorithm for least squares estimation of non-linear parameters," *SIAM J. Appl. Math.*, vol. 11, pp. 431–441, 1963.

[25] The minpack library, Netlib repository, http://www.netlib.org/minpack/.

[26] G. A. Abrantes and F. Pereira, "MPEG-4 facial animation technology: survey, implementation, and results," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 9, no. 2, pp. 290–305, 1999.

[27] F. Pighin, J. Hecker, D. Lischinski, R. Szeliski, and D. H. Salesin, "Synthesizing realistic facial expressions from photographs," *Computer Graphics*, vol. 32, pp. 75–84, August 1998, Annual Conference Series.

[28] Esopos, Hellenic TTS System, http://esopos.ee.auth.gr.

[29] V. Pagel, K. Lenzo, and A. W. Black, "Letter to sound rules for accented lexicon compression," in *Proc. 5th International Conference on Spoken Language Processing (ICSLP-98)*, Sydney, Australia, November 1998.

Nikos Grammalidis is an Associate Researcher in the Informatics and Telematics Institute. He received his B.S. and Ph.D. degrees in electrical and computer engineering from the Aristotle University of Thessaloniki, in 1992 and 2000, respectively. Prior to his current position, he was a Researcher in 3D Imaging Laboratory at the Aristotle University of Thessaloniki. His main research interests include image compression, 3D data processing, multimedia image communication, 3D motion estimation, stereo and multiview image sequence coding. His involvement with those research areas has led to the coauthoring of more than 10 articles in refereed journals and more than 25 papers in international conferences. Since 1992, he has been involved in more than 10 projects, funded by the EC and the Greek Ministry of Research and Technology.

Nikos Sarris was born in Athens, Greece in 1971. He received his Master of Engineering degree in computer systems engineering from the University of Manchester Institute of Science and Technology (UMIST), in 1995. He has worked as a Research Assistant in the Information Processing Laboratory of the Aristotle University of Thessaloniki for 4 years, where he participated in several national and international projects. Mr. Sarris has been a member of the Greek Technical Chamber as a Computer Systems and Informatics Engineer since 1996 and is currently working for the Informatics and Telematics Institute as a Research Fellow, while being a Ph.D. candidate in the Aristotle University of Thessaloniki. His research interests include 3D model-based image and video processing, image coding, image analysis and sign language synthesis and recognition.

Fani Deligianni received her B.S. degree (Diploma) in electrical and computer engineering from Aristotle University of Thessaloniki, Greece in 2000. In 2001, she has worked as a Researcher in the 3D Imaging Laboratory at the Aristotle University of Thessaloniki. Her main interests include 3D computer graphics, facial animation, and MPEG-4 coding. Ms. Deligianni is currently pursuing her M.S. degree in advanced computing at Imperial College, London.

Michael G. Strintzis received the Diploma degree in electrical engineering from the National Technical University of Athens, Athens, Greece in 1967, and the M.A. and Ph.D. degrees in electrical engineering from Princeton University, Princeton, NJ, USA in 1969 and 1970, respectively. He then joined the Electrical Engineering Department at the University of Pittsburgh, Pittsburgh, Pa., USA, where he served as Assistant Professor (1970–1976) and Associate Professor (1976–1980). Since 1980 he has been a Professor of electrical and computer engineering at the University of Thessaloniki, Thessaloniki, Greece, and, since 1999, Director of the Informatics and Telematics Research Institute, Thessaloniki. His current research interests include 2D and 3D image coding, image processing, biomedical signal and image processing, DVD, and Internet data authentication and copy protection. Dr. Strintzis is serving as an Associate Editor of the IEEE Transactions on Circuits and Systems for Video Technology since 1999. In 1984, Dr. Strintzis was awarded a Centennial Medal by the IEEE.