COMPUTER ANIMATION AND VIRTUAL WORLDS
Comp. Anim. Virtual Worlds (2012)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cav.1450

SPECIAL ISSUE PAPER

Position-based facial animation synthesis

Marco Fratarcangeli*

Department of Computer and Systems Science 'A. Ruberti', Sapienza University of Rome, Rome, Italy

ABSTRACT

We propose an integrated facial dynamics model addressing the animation of 3D humanoid faces in real time. The computational model mimics facial motion by reproducing the layered anatomical structure of a human head including the bony structure, overlapping facial muscles and the skin. The model is flexible enough to animate face meshes of various shape, connectivity, and scale. Different from previously proposed approaches based on mass–spring networks, overshooting problems are avoided by simulating the dynamics through a position-based scheme, which allows for real-time performance, control, and robustness. Experiments demonstrate that convincing expressive facial animation can be interactively prototyped on consumer class platforms. Copyright © 2012 John Wiley & Sons, Ltd.

KEYWORDS

facial animation; deformable bodies; muscle model; virtual humans

Supporting information may be found in the online version of this article.

***Correspondence**

Marco Fratarcangeli, Department of Computer and Systems Science 'A. Ruberti', Sapienza University of Rome, Rome, Italy.
E-mail: frat@dis.uniroma1.it
http://www.dis.uniroma1.it/∼frat/

1. INTRODUCTION

Animation of virtual faces has been an active research field in computer graphics for almost 40 years [1]. The applications of facial animation include such diverse fields as character animation for films and advertising, computer games, video teleconferencing, user-interface agents, and avatars, as well as facial surgery planning. In character animation, it is of critical importance to accurately reproduce facial motion because it is one of the prime sources of emotional information.

The complex and sophisticated structure of the human head involves the motion, deformation, and contact handling between bio-tissues that are viscoelastic, nonlinear, anisotropic, and structurally heterogeneous. This makes it hard to formulate a mathematical model able to represent the biomechanical inner workings of the face. Nonetheless, high accuracy and precision are required because, as humans, we are used to observing and decoding facial expressions from the moment we are born, and we are expert at easily detecting the smallest artifacts in a virtual facial animation.

We propose a muscle-based approach for facial animation based on the stable and robust position-based dynamics (PBD) schema introduced in [2]. This allows for avoiding the overshooting problem typical of force-based systems solved explicitly, and it enables real-time performance and interactive application.

The main contribution resulting from our work concerns an anatomical model, which is able to generate facial expressions and can be applied to a wide range of input face meshes varying in shape, connectivity, and dimensions.

Different from other approaches [3,4], our muscle model is adaptive, and it is not bound to a specific template skin mesh and muscle map. The complete facial model allows for any number of muscle and passive tissue layers; muscles can actively contract, and each layer influences the deformation of the layers on top of it, but not those underlying it.

The contraction of each muscle is controlled by a single, scalar parameter; this allows for finding easily a mapping between muscle contractions and the motion of facial landmarks tracked from a human subject. We use this mapping for transferring the motion from real actors to virtual heads and evaluate the quality of the produced animations.

2. RELATED WORK

For a comprehensive overview of the methods proposed in literature to achieve facial animation, please refer to one of the many surveys on the topic, for example, [5,6]. Here, we focus on muscle model-driven facial animation pioneered more than 30 years ago [7,8].

Among the most advanced techniques currently used for muscle-based facial animation are the finite element

algorithms and mass–spring models. Finite element models of facial muscles are used to build an accurate biomechanical musculature model. These methods enable to reproduce with accuracy highly realistic facial expressions (e.g., [9–12]), but they have a high computational cost and set-up time.

Other anatomically based models use mass–spring formalism and may lead to interactive applications [3,4,13,14]. These methods allow for interactivity, but the behavior of the network is sometimes difficult to predict and it may depend from the topology and scale of the skin mesh.

In particular, Kähler *et al.* [4] devised a computational representation of the anatomical structure of a real head, complete with the skull, a single muscular layer, and skin, in order to model a general template of a virtual face. The muscle map can be interactively designed by the artist through an editing tool that is not hard-coded to the template mesh. The different anatomical parts are connected together, resulting in the final template face model. To represent different humans, the template model adapts its shape and appearance, while conserving spatial dimensions and connectivity.

3. POSITION-BASED DYNAMICS

Although our anatomical model is still based on the simulation of a particle system, like in mass–spring networks, the physical simulation is computed using the PBD introduced by Müller [2].

Position-based dynamics allows for imposing nonlinear constraints of a geometric nature on a deformable surface, as in the case of volume preservation of the whole surface or of maintaining distance between two nodes of the mesh during deformation. This permits the modeling of the virtual anatomical structures without the use of internal or external forces, which simplifies the deformable model and produces unconditionally stable simulations, as a result of the elimination of the overshooting problem.

A geometric constraint can be expressed as $C_j(\mathbf{p}_1, \ldots, \mathbf{p}_n) \geq 0$. During the simulation, given the current spatial configuration $\mathbf{p}$ of the set of particles, we want to find a correction $\Delta\mathbf{p}$ such that $C(\mathbf{p} + \Delta\mathbf{p}) = 0$. In order to be efficiently solved, the constraint equation is approximated by

$$C(\mathbf{p} + \Delta\mathbf{p}) \approx C(\mathbf{p}) + \nabla_{\mathbf{p}} C(\mathbf{p}) \cdot k\,\Delta\mathbf{p} = 0 \qquad (1)$$

where $k \in \{0, \ldots, 1\}$ is the stiffness parameter. The solution of the set of the resulting nonlinear constraints governing the dynamic system, is computed through an iterative Gauss–Siedel solver.

In this context, stiffness k can be considered as the speed with which the particle positions converge towards a legal spatial state, that is, a state in which all the constraints are satisfied. By tuning the value of k, we control how much a constraint is stringent during the evolution of the system. For example, a distance constraint between two particles with $k = 0.5$ behaves similar to a spring, whereas with $k = 1.0$ behaves nearly like a stiff rod.

For an exhaustive explanation of the PBD method, including how to specify and solve the various constraint types, please refer to [2,15].

4. SCATTERED DATA INTERPOLATION

In different parts of the following description of model building process (i.e., Sections 5.1 and 7), we use a scattered data interpolation function for morphing a mesh M such that the position of a subset of its vertices P matches with another set Q not belonging to M. In mathematical terms, for given position pairs (p_i, q_i), $(i = 1, \ldots, n; p_i \in \mathbb{R}^3, q_i \in \mathbb{R}^3)$, we find $G : \mathbb{R}^3 \longrightarrow \mathbb{R}^3$, $G \in C^1$ so that

$$q_i = G(p_i) \qquad i = 1, \ldots, n \qquad (2)$$

By applying $G(p)$ to all the vertices of M, we obtain $p_i = q_i, i = 1, \ldots, n$, and the remaining vertices of M deformed accordingly. We define $G(p)$ as a sum of radial basis functions (RBFs) based on Hardy multiquadrics, which proved to be effective in the context of surgical correction and growth simulation of humanoid heads. The reader can refer to [16,17] for the definition details.

5. MUSCLE MODEL

In this section, we describe the adaptive model, which can simulate linear and circular muscles, by imitating their macroscopic behavior when they contract or elongate.

We first describe *action lines*, which allow for controlling muscle activation by using a single scalar parameter expressing the magnitude of the muscle contraction. Then, we describe the construction and control of the volumetric muscles out of action lines.

5.1. Geometric Shape

The initial shape of a muscle in its rest state is defined by a closed contour C lying over existing anatomical structures. Next, a template hexahedral mesh is morphed such that it matches C (Figure 1). The deformed mesh is used to build the geometrical constraints, which govern the muscle dynamics. In the following section, we first explain how to define C through piecewise linear curves (namely, *action lines*) and, then, how to build the muscle mesh through the morphing of a template mesh.

5.1.1. Action Lines.

An *action line* is a piecewise linear curve $A \in \mathbb{R}^3$ lying over a set of triangulated surface meshes. The purpose of the action lines is twofold: (1) they define the bottom contour of the muscle geometry during the simulation, and (2) they provide a mechanism to control the active contraction of the muscle itself, as explained in Section 5.2.

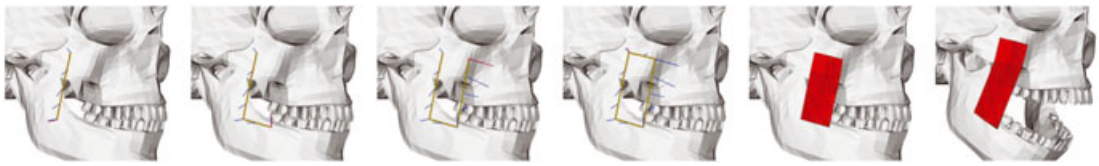


Figure 1. The closed contour C is drawn upon underlying structures, in this case the skull. Then, a hexahedral mesh M is morphed to fit C. M is passively deformed as the skull moves.

A *surface point* is a point $sp \in S$, where S is a triangulated surface in $\mathbb{R}^3$. The surface point sp is uniquely described by the homogeneous barycentric coordinates w.r.t. the vertexes of the triangular facet to which it belongs.

The relevant attributes of a surface point are the *position* and the *normal* attributes; both of them are obtained as the linear combination of the barycentric coordinates combined with the corresponding attributes of the triangle vertexes. When these latter are displaced because of a deformation of the triangle, the new position and normal of sp are also updated, using the new attributes of the vertexes.

Each segment forming the action line is defined by two surface points; thus, an action line is described by the ordered list of its surface points. When the underlying triangles deform, the surface points on them are displaced and the action line deforms accordingly. A graphical example of this mechanism is shown in Figure 2.

5.1.2. Specification of Muscle Geometry.

Using action lines, composed by surface points, we define a muscle contour C, which is formed by four action lines $A_i, i = 0, \ldots 3$, two latitudinal and two longitudinal lines. The end points of the latitudinal lines are connected to the end points of the longitudinal ones, as shown in Figure 3. Note that the surface points are defined over the barycentric coordinates of existing triangles. So, a contour C may be defined only over already existing meshes.

C is sampled at regular intervals separately on each action line; the sample step for each action line is defined according to the Euclidean norm measured on the segments:

$$step = \frac{\sum_{i=0}^{N} \|sp_i - sp_{i-1}\|}{N}, N \in \{w, h\} \qquad (3)$$

where N assumes the value w or h, depending on the type of the sampled action line, longitudinal or latitudinal, respectively.

Each sample is characterized by its position s and normal n_s, similarly to the surface points. The position s_i and the normal vector n_{si} of each sample on an action line are found through a simple linear combination of the corresponding surface point attributes.

Because the number of samples is w for each latitudinal action line and h for each longitudinal line, the cardinality of the set of samples of C is $2(w + h)$. We build a triangulated, oriented hexahedral mesh $M \in \mathbb{R}^3$ with exactly h longitudinal section and w latitudinal sections (Figure 4). The contour of one of the sides of the mesh is arbitrarily chosen as the *bottom* contour.

We define a bijective mapping between the samples on the actions lines of C and the contour of the hexahedral mesh. From this mapping, we compute a scattered data interpolation $G(p)$ (Section 4). Applying $G(p)$ to M, we obtain the mesh M' lying on top of the surfaces where C is defined. The size and orientation of the initial mesh M are not important and, thus, are not specified because they do not influence the final shape of M'. This is because the RBF-based algorithm is invariant w.r.t. affine transformations, such as translation and rotation. The thickness of the mesh M' can thus be adjusted by displacing the vertices on the upper surface along their normals.

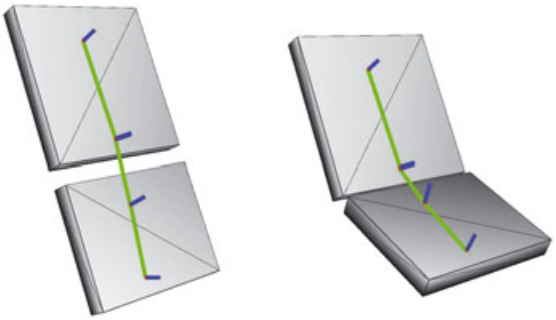


Figure 2. When the bottom icosahedron rotates, the surface points and, thus the action line, update accordingly.

5.2. Muscle Dynamics and Control

The muscle mesh is the basis for defining the geometric constraints that govern the dynamics of the muscle itself. The expressive facial muscles are of two basic types: linear and sphincter. The action produced by these muscles is always along the tangent of their fibers. We use a generic template muscle model in which the fibers are represented by the longitudinal action lines forming the contour C. The constraints on the template model and its geometry are adapted according to the type of muscle that has to be simulated. In Sections 5.2.1 and 5.2.2, we describe how muscle contraction is controlled through the modulation of the sampling step of the longitudinal action lines.

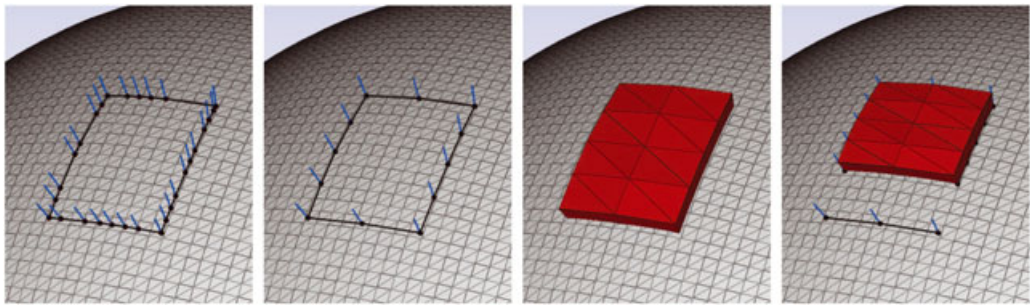


Figure 3. (a). A contour C composed by four action lines layed upon a curved, triangulated surface. Blue segments represent the surface points' normal. (b) The contour is sampled (h = 5, w = 3). (c) The hexahedral muscle mesh is morphed to fit C, and it is constrained to the position of the sample points. (d) Changing the sampling step, the sample points *slide* on the action lines, and the muscle contracts.

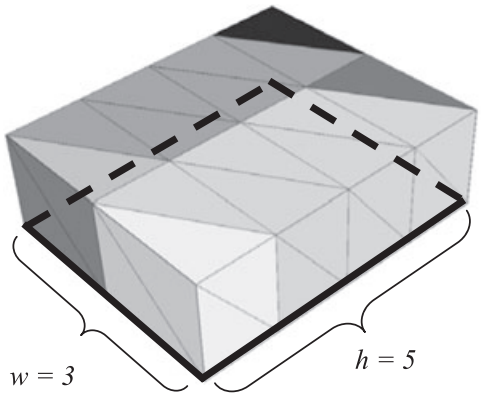


Figure 4. Hexahedral mesh in its initial state. This example has $h = 5$ latitudinal sections and $w = 3$ longitudinal sections.

First, a network of *distance* constraints is built as shown in Figure 5. This network alone is not sufficient to describe the dynamics of the muscle. In order to mimic the bulging behavior of the muscle when contracting and to avoid face inversions, for each pair of triangulated faces f_i, f_j sharing the same edge e_{ij}, a *bending* constraint is defined. A bending constraint conserves superficial tension and reacts whenever there is a compression, or an elongation, perpendicular to the edges of the triangulated muscle mesh M'. A further *volume preserving* constraint is defined over all the particles of the muscle geometry, making the muscle thicker as a result of compression and thinner due to elongations.

The softness of the muscular deformable body depends on the stiffness of each individual constraint, regardless of its kind and from the number of Gauss–Siedel iterations performed within the numerical integration [2]. Because this latter parameter is fixed and set to the lowest possible value ensuring interactivity (as specified in Section 3), each type of muscle defines its own set of stiffness constraints as described in the following two sections.

5.2.1. The Linear Muscle.

A linear muscle model represents a sheet muscle, a flat and thin muscle, whose fibers are arranged in parallel threads. When it contracts, the fibers shorten by equal amounts and pull the skin surface from the insertion site towards the origin site, which can be regarded as fixed.

Given the contour C, as defined in Section 5.1, and the action lines $A_i, i \in \{0, \ldots 3\}$, we consider the latitudinal action lines A_3 and A_1 as, the *origin* site and the *insertion* site of the muscle, respectively. The longitudinal action lines A_0 and A_2 represent the fibers of the muscle and define the direction of contraction.

Each particle $\mathbf{p}_i$ of the muscle model, which corresponds to the sample s_j on the longitudinal action lines A_1 and A_3, is bound to its corresponding sample through a *position* constraint (Section 3). If the sample s_j is displaced, the dynamics of the corresponding particle $\mathbf{p}_i$ will change the position p_i attempting to match this displacement. The new position of the samples is computed by using Eq. 3 with the sampling step:

$$step' = step(1 - c) \tag{4}$$

where $step$ is defined in Eq. 3 and $c \in [0, 1]$ is the *contraction factor*. If $c = 0$, the muscle is in rest position. If $c = 1$, it is undergoing the greatest contraction possible.

The constraints, with the corresponding stiffness, involved in the dynamics of the sheet muscle are the following: $k^{\text{up}}_{\text{latitudinal}} = 1.0, k^{\text{up}}_{\text{longitudinal}} = 0.1, k_{\text{connection}} = 1.0, k_{\text{bend}} = 0.4, k_{\text{volume}} = 1.0, and\, k_{\text{position}} = 0.2$. The stiffness of the remaining constraints is set to 0; these constraints are thus disabled and not considered further in the simulation. Note the low stiffness value of the position constraints: although the difference between the expected position and the actual position remains acceptable, such a low value allows the muscle to slide over the underlying surfaces.

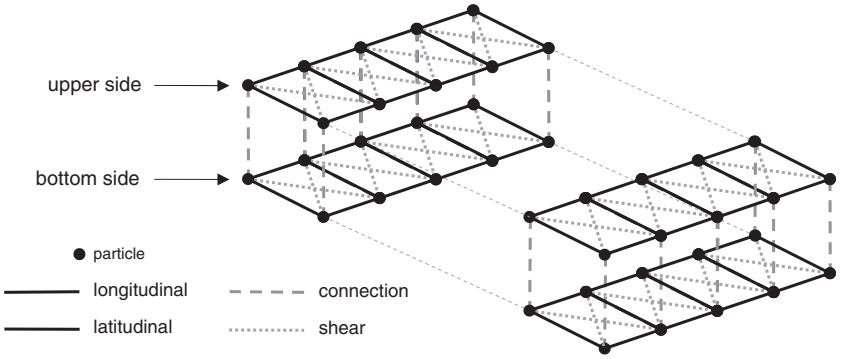


Figure 5. Distance constraints involved in muscle model dynamics.

5.2.2. The Sphincter Muscle.

The sphincter muscle model is used to simulate the *orbicularis oris*, which is the muscle surrounding the oral orifice.

In the sphincter muscle model, the two longitudinal action lines, A_1 and A_3, are both closed curves (for both, the first surface point corresponds to the last) and define the inner and the outer contours of the muscle, respectively. The remaining latitudinal action lines, A_0 and A_2, are coincident and both connect the first surface point of A_1 to the first surface point of A_3.

In the sphincter muscle model, there is no fixed site of origin. In fact, while contracting around a virtual center $\mathbf{o}$, the body of this kind of muscle slides on the underlying anatomical structures.

Similarly to the way in which other authors have approached the problem in the context of interactive physically based facial animation [3,13], we model the contraction of the sphincter muscle through the use of the parametric ellipsoidal function. The ellipsoid constructed by choosing the best fit for the action line samples in their current position in accordance with the least square method, whereas the position of the virtual center $\mathbf{o}$ is computed as their average position.

An ellipsis is defined to compute the sample displacements and, thus, the motions of the muscle. For each pair $\left\langle s_i^1, s_j^3 \right\rangle$ of action line samples,

$$x = \left(s_j'^3 - o\right) \cdot dir_a$$
$$y = \left(s_j'^3 - o\right) \cdot dir_b$$

so that

$$\delta = \left(o - s_i^1\right)(1-c)\cos\left(\frac{\pi}{2}\left(1 - \frac{\sqrt{y^2b^2 + x^2a^2}}{ab}\right)\right)$$
$$s_i^1 = s_i^1 + \delta$$
$$s_j^3 = s_i^3 + \delta$$

where dir_a and dir_b are respectively the directions of the semi-major and semi-minor axis of the ellipses, (x, y) is the coordinate of $s_j'^3$ in the coordinate frame of dir_a and dir_b, and δ is the displacement of the samples and, thus, the muscle motion. The constraints, with the corresponding stiffness, involved in the dynamics of the sphincter muscle are as follows: $k^{\text{up}}_{\text{latitudinal}} = 0.2, k_{\text{connection}} = 1.0, k^{\text{up}}_{\text{shear}} = 0.3, k^{\text{low}}_{\text{shear}} = 0.3, k_{\text{bend}} = 1.0, k_{\text{volume}} = 1.0$, and $k_{\text{position}} = 0.2$. Like in the case of the sheet muscle, the stiffness of the remaining constraints is set to 0, and thus, they are disabled.

6. THE FACIAL MODEL

The facial model is organized into layers. The deepest layer is the bony structure, which is the skull. Next are the successive layers of muscle, one lying below the other. The last, most superficial layer is the skin, represented by the face mesh that is to be animated. The deepest layer influences the more superficial ones, propagating the deformation until the skin moves, producing animation. Multiple muscle layers allow the artist to design muscle maps reflecting the arrangements of the real head. This leads to a wider degree of expressiveness compared with approaches allowing only a single muscular layer. Before discussing the musculature and its influence upon the shape of the skin surface, we briefly describe the influence of the articulated, rigid skull.

6.1. The Skull

The skull is represented by a triangulated surface mesh $\in \mathbb{R}^3$, where the other anatomical parts are laid. Figure 8(a) illustrates an example of the mesh used in the experiments. The face model is not limited to the use of this particular skull mesh.

The skull is divided into two components: the upper skull and the mandible. The mandible constitutes the complete lower jawbone. The motions of the mandible are modeled through the use of rigid transformations applied to the jaw mesh vertices. The lowering and elevation of the jaw are represented as rotation around the *pitch* axis, the small amount of lateral movement as rotation around

the *yaw* axis, and finally, protrusion and retraction as a translation along the *protrusion* axis, thus making the total number of degrees of freedom equal to three.

A set of 31 landmarks is defined on the skull surface (Figure 8(a)). The landmarks belong to a subset of MPEG-4 Facial Definition Points [18]. Their purpose is twofold: (1) they are useful in defining the pitch and yaw rotation axes and the direction of protrusion of the mandible, and (2) they provide a spatial reference for computing the morphing function used to fit the skull and the muscle map into the facial skin mesh (Section 7).

6.2. The Muscle Map

The muscle map is made up of overlapping muscles, and it is positioned over the skull. The muscle model presented in Section 5 is used to build the facial muscles, which are responsible for expression, and the structural muscles, which are used to support other muscles and the skin. Each layer influences the deformation of the layers on top of it, but not those underlying it. Figure 6 presents the different layers forming the particular muscle map used in the experiments.

The muscle map comprises 25 linear muscles and one circular muscle, the orbicularis oris.

Some of the linear muscles do not actively move but are just passively deformed by the underlying structures. Their function is to provide structural support to other muscles, particularly the *masseter*, in which the *risorius* muscle originates, and the fatty tissue under the cheeks, on top of which are the *zygomaticus major* muscle, the *levator labii*, and part of the *orbicularis oris*. The last supporting muscle is the *platysma*, under the chin bone: it is useful as a support for the skin when the jaw rotates; otherwise, the skin vertices would enter in the lower part of the face producing an unrealistic and unpleasant effect.

Each simulated muscle is linked to the underlying structures through position constraints following the position of surface points. Thus, when an anatomical structure deforms (muscle or bone), all the surface points lying on it move as well, which in turn influences the motion of the above-linked structures. For instance, when the jaw, which is part of the deepest layer, rotates, all the deformable tissues that, totally or partially, lay on it will be deformed as well and, so on, in a chain reaction, which eventually deforms the skin (Section 8). In Figure 7, the example muscle map is deformed by rotating the jaw and contracting the *frontalis* bellies. All the aforementioned muscles are deformed accordingly.

7. CONSTRUCTION PROCESS

Once the musculoskeletal system has been defined as specified in the previous sections, it can be used to animate the input skin mesh. The skull and the muscle map must first be morphed in order to fit into the skin shape. The next step (Section 8) involves anchoring the skin to the bones and muscles so that it follows their deformation and becomes animated.

The fitting algorithm uses RBFs (as done in Section 5.1.2), to morph the musculoskeletal model. Figure 8(a) shows the pairs of landmarks for a skull and a skin mesh, used in the process. These pairs have been empirically defined, through the application of the morphing algorithm to several skin meshes and the visual assessment of areas where the skull mesh did not properly fit into the skin. Green dots represent the initial set of landmarks provided during input. Red dots are the landmarks obtained through the enhancement process.

The landmarks on the skull surface may be covered by the muscles. In such a case, the landmarks are brought directly onto the muscle surface through a cylindrical projection. A ray is cast from the outside of the skull, towards the inner cylindrical projection axis, and if it intersects a muscle, it is assigned to it. Before the ray-tracing procedure, the thickness t of the muscles is

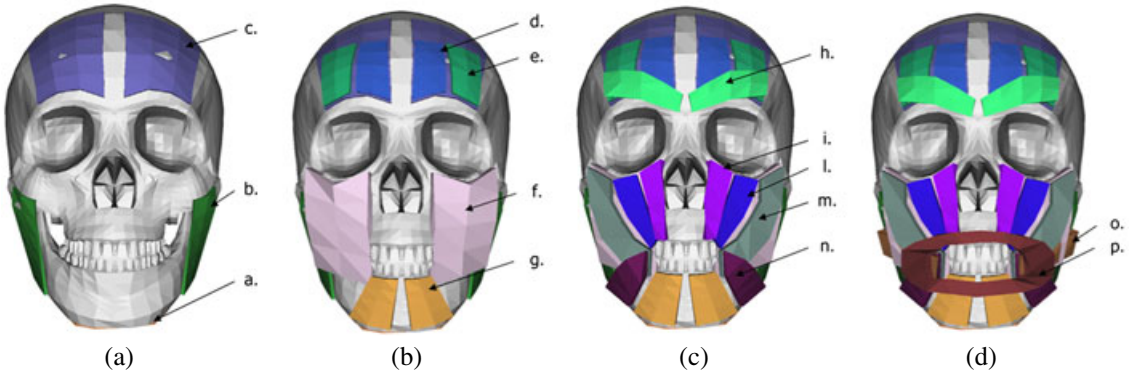


Figure 6. Muscle Map. (a) *a*, platysma; *b*, risorius; *c*, frontalis. (b) *d*, frontalis inner; *e*, frontalis outer; *f*, fatty tissue; *g*, depressor anguli. (c) *h*, corrugator supercilii; *i*, levator labii inner; *l*, levator labii outer; *m*, zygomaticus major; *n*, depressor anguli. (d) *o*, risorius; *p*, orbicularis oris.

DOI: 10.1002/cav

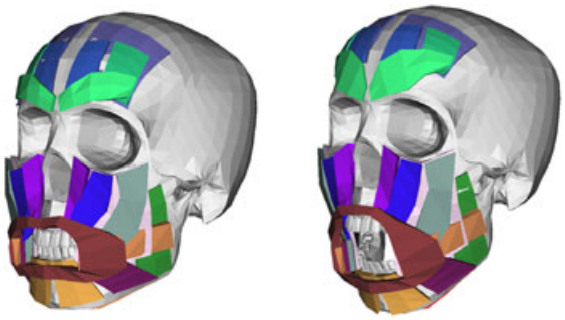


Figure 7. The jaw depresses while the frontalis bellies contract; the whole muscle map is deformed accordingly.

recomputed in accordance with the skull's dimensions after the morphing procedure

$$t = \frac{d_{\text{head}}}{d_{\text{skull}}} t \tag{5}$$

where d_{head} and d_{skull} are the diagonal of the axis-aligned bounding box, which embed the skull and the skin mesh, respectively. By projecting the skull's landmarks onto the muscles, with their thickness, and running again the fitting algorithm, the muscle map is accommodated just beneath the skin surface.

The rear section of the skull is not required to fit into the skin mesh because the deformation of the skin is not propagated in the back of the head but happens in the area of the jaw and the muscle map, as explained in the following section.

8. SKIN

The skin is modeled as a network of geometrical constraints, in a similar way to the muscles. The skin is built starting from the triangulated mesh provided in input.

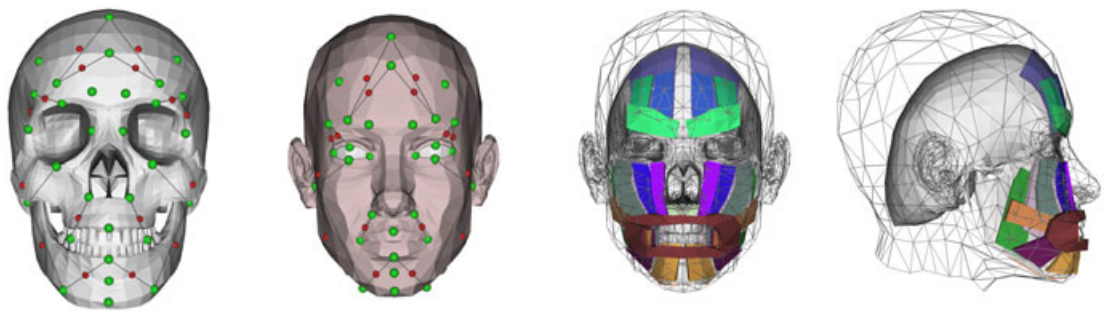
A stretching constraint is placed along each edge of the triangular faces. For each pair of adjacent triangles, a further stretching constraint is placed along the opposing vertices. This latter constraint penalizes to bending and twisting stresses. The target length is the Euclidean distance among the particles in rest state. A bending constraint is defined on the two faces, and the target angle is the dihedral angle among the two faces in rest state. A triangle area preservation is also imposed on each triangular face. Finally, a volume preservation constraint is defined over all the particles belonging to the skin. The target volume is the initial volume of the mesh in rest state. The value of the stiffness for each kind of constraint is found empirically by assessing the visual quality of the animation.

The empirical values, which produce the most plausible results are as follows: $k^{\text{stretching}} = 0.15, k^{\text{bending}} = 0.95, k_{\text{area}} = 0.95, k_{\text{anchor}} = 0.2$, and $k_{\text{volume}} = 1.0$.

After the skull and the muscle map are fitted onto the skin mesh, further constraints are defined to bind the skin to the underlying muscoloskeletal structure. For each particle $\mathbf{p}$ in the skin mesh, a ray is casted along the normal, that is towards the outer direction. In fact, after the fitting, portions of some muscles may stay outside the skin. By projecting in the outer direction, the skin vertexes are first bound to these muscles. If no intersection is found, then another ray is casted in the opposite direction of the normal, towards the inner part of the head. The ray is tested against the muscles from the most superficial to the deepest one. If the ray does not intersects any muscle, then the skull is tested. The normal of the skin particle is considered as the average normal among the normals of the star of faces to which the particle belongs.

If an intersection is found, then it is defined a surface point sp on the intersected triangular face in the position where the ray intersects the face. A particle $\mathbf{q}$ is added to the system, and it is bound through a position constraint to sp. A stretching constraint is defined between $\mathbf{p}$ and $\mathbf{q}$.

When the bones and the muscles move, the position of the surface points will change accordingly. The set of added particle $\{q\}$ is updated as well because it is bound to the surface points through the corresponding position constraint and will displace the skin particles.



(a) (b)

Figure 8. (a) Set of reference points used to compute the morphing function for fitting the skull mesh into the skin. Green dots represent reference points provided in input; red ones are computed by ray tracing. (b). The result of the morphing process. The front part of the skull and the muscle map, that is the movable part of the musculoskeletal system, are fitted inside the skin mesh.

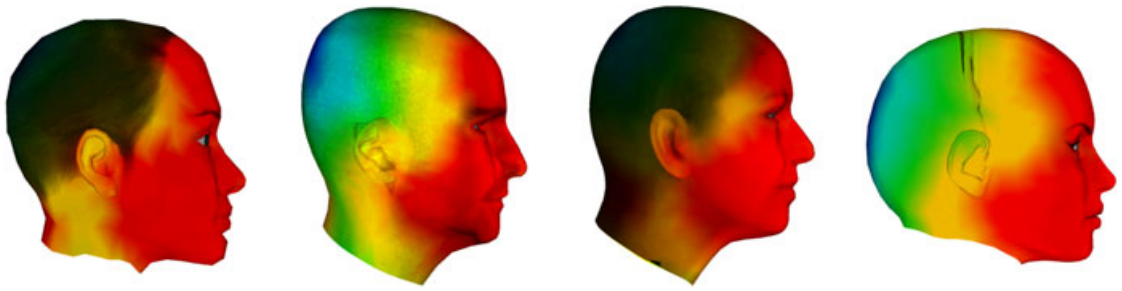


Figure 9. Influence of the musculoskeletal structure on the skin. Red are zones under direct control of the musculoskeletal structure, blue with no control at all.

Because the main part of facial animation happens in the frontal part of the face, where the main part of the muscles and the jaw are, the deformation of the skin is not propagated to the back of the head. This is achieved by computing the normalized geodesic distance from all the skin particles, which are anchored to a muscle or to the jaw, to the remaining skin particles. The scalar values field produced by this process is called *influence map*, and it is used as a weight for the displacement of the skin particles. Figure 9 shows the influence map as a color map on some input skin meshes used in the experiments.

9. IMPLEMENTATION AND RESULTS

The entire facial model has been implemented as a real-time software framework, created in C++ and OpenGL. The test platform is a consumer class platform, an Intel Core2 Duo CPU at 2.40 GHz, with 2 GB of main memory using only one core. The system is tested through visual evaluations of the various facial expressions produced with different target skin meshes and, particularly, the basic expressions defined in [19]: anger, disgust, fear, joy, sadness, and surprise. Time performance is also measured and reported in Table I.

In the preprocessing phase, the most demanding step is the definition of the muscle map, which can be designed in less than one working day and then be used for all the input skin meshes. Once the muscle map is ready, the time required for fitting the musculoskeletal structure inside the skin is less than 1 second, whereas binding it to the skin from the inside, which involves ray tracing for each skin vertex, takes less than 5 seconds per each input skin mesh.

Table I. Facial dynamics computation time. *V*, number of vertices; *F*, number of faces; *C*, number of constraints; *t*: physical simulation time for each animation frame.

Mesh	*V*	*F*	*C*	*t* (millisecond)
Mesh_A	1466	2822	16 461	31
Mesh_B	2502	4908	28 785	28
Mesh_C	4604	9160	53 834	45
Mesh_D	4904	9541	55 687	50
Skull + muscles	4297	8352	4919	6

Figure 10 shows several face models built from the input skin mesh as they are performing basic facial expressions. Further results are available in the accompanying videos, where we also show an interactive editor, which assists the artist in sketching the muscles and the passive tissues directly on the bones and already existing muscles.

To further validate the expressiveness of the facial model, we mapped tracked motion from real faces to the virtual muscle contractions and visually evaluated the resulting animation. We used videos of actors performing basic expressions taken from the MMI-Facial Expression Database collected by Valstar and Pantic [20]. The actual tracking was performed using a facial tracking software package [21] commercially available.

The output of the tracking software is a description of the facial motion encoded using Facial Animation Parameters (FAPs), defined in the MPEG-4 Facial and Body Animation standard [18].

Each parameter describes a basic action that a face can perform, such as closing an eyelid or stretching a corner lip. Most of these actions have been defined considering the displacement caused by the contraction of a single muscle. The magnitude of facial parameters is expressed in normalized units of measure (Facial Animation Parameter Units, FAPU); thus, the magnitude value is independent from the tracked face morphology.

Because we used a muscle map similar to the one in a real face and the value of the muscle contractions is normalized between 0 and 1 by construction, we built a mapping between the normalized FAPs and muscle contractions. For each normalized FAP, we define which muscles contract to synthesize the facial action described by the FAP. We are able to precisely map the muscle contraction to the magnitude of a FAP because we control the muscle using the contraction c (Section 5.2), which tunes the muscle fiber length. This would be much more difficult to do if our muscle model would be based on forces rather than positions. Figure 11 shows some of the obtained results. The corresponding animations are available in the accompanying videos.

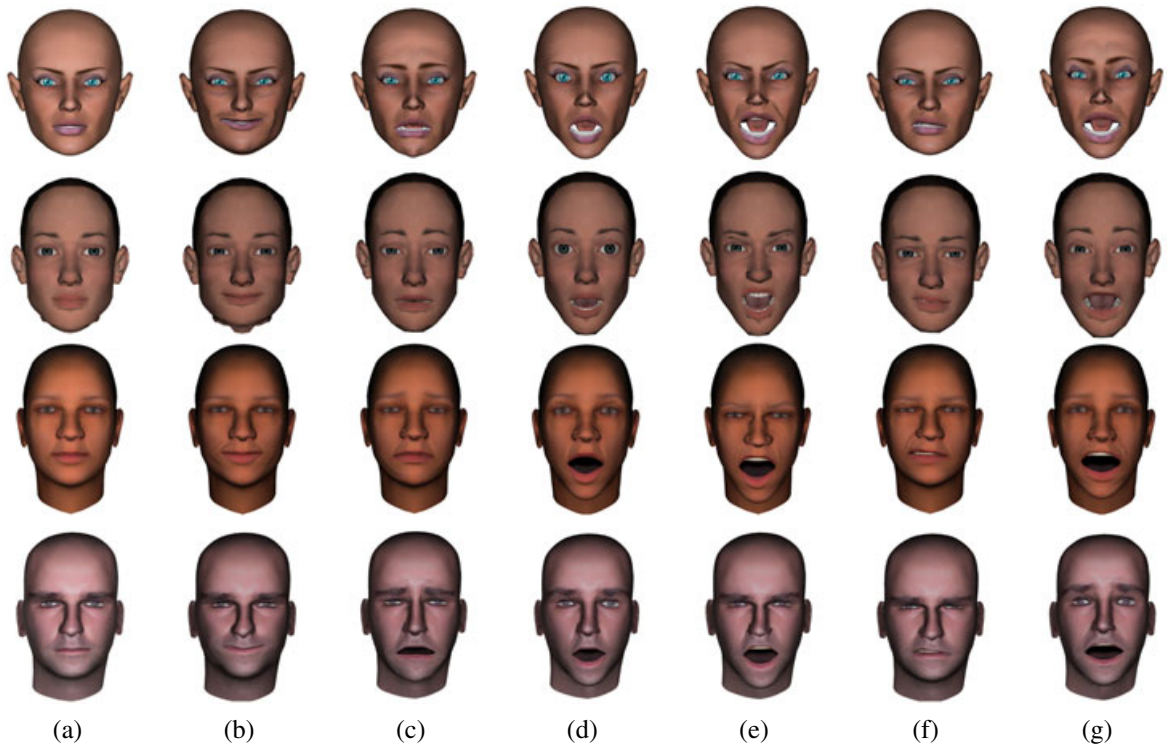


Figure 10. Test input faces performing basic expressions. (a) Neutral; (b) joy; (c) sadness; (d) surprise; (e) anger; (f) disgust; (g) fear.

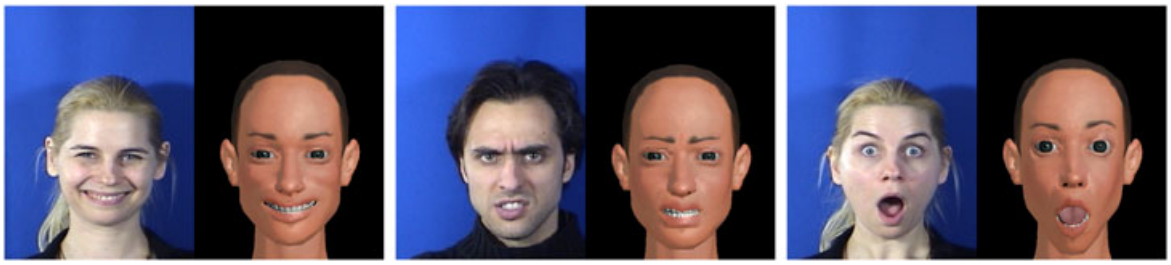


Figure 11. Facial tracking applied to muscle contractions.

10. LIMITATIONS AND FUTURE WORK

We have presented an anatomically based muscle model, which is not restricted to a specific input skull or skin mesh. Unlike past approaches [3,4], it is not bound to a template mesh and allows for any number of facial muscle and passive tissue layers. These layers interact and slide upon each other, forming an overlapping structure. The given muscle model is deformed according to its own active contraction, which is controlled by a single scalar parameter, or due to the motion of the underlying anatomical structures. The skin is deformed by muscular and bony motion.

We have modeled computationally inexpensive yet expressive anatomical elements by using the PBD approach for the facial animation setting. This simulation approach has proven to be effective at modeling complex dynamics in a manner guaranteeing stability, which is of major importance in interactive applications. Because the stiffness values of the constraints are normalized between zero and one and are dimensional, the parameters are suitable for the creation of virtual models of heads characterized by various scale and shape.

The Gauss–Seidel iterative method used to work out the dynamics of the model may lead to some instability when the number of constraints increases. However, this can be solved by increasing the number of iterations in the

numerical integration, which however may lead to slower frame rate.

ACKNOWLEDGEMENTS

This paper describes research carried out under the EUFP7 ICT 247870 NIFTI project.

REFERENCES

1. Parke FI. Computer generated animation of faces. In *ACM Annual Conference (ACM72)*. ACM, New York, NY, USA, 1972; 451–457.
2. Müller M, Heidelberger B, Hennix M, Ratcliff J. Position based dynamics. In *Virtual Reality Interactions and Physical Simulations (VRIPhys)*, Madrid, November 6-7 2006.
3. Lee Y, Terzopoulos D, Waters K. Realistic modeling for facial animation. In *Computer Graphics and Interactive Techniques (SIGGRAPH95)*. ACM, New York, NY, USA, 1995; 55–62.
4. Kähler K, Haber J, Yamauchi H, Seidel H-P. Head shop: generating animated head models with anatomical structure. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA02)*. ACM, New York, NY, USA, 2002; 55–63.
5. Parke FI, Waters K. *Computer Facial Animation*, (2nd edn). A. K. Peters, Ltd., 2008.
6. Haber J, Terzopoulos D, Magnenat-Thalmann N, Blanz V. *Facial Modeling and Animation—Eurographics 2003 Tutorial Notes*. Eurographics, Granada, Spain, 2003.
7. Platt SM, Badler NI. Animating facial expressions. *Computer Graphics and Interactive Techniques* 1981; **15**(3): 245–252.
8. Terzopoulos D, Waters K. Physically-based facial modeling, analysis, and animation. *Journal of Visualization and Computer Animation* 1990; **1**(2): 73–80.
9. Gladilin E, Zachow S, Deuflhard P, Hege HC. Anatomy-and physics-based facial animation for craniofacial surgery simulations. *Medical and Biological Engineering and Computing* 2004; **42**(2): 167–170.
10. Sifakis E, Neverov I, Fedkiw R. Automatic determination of facial muscle activations from sparse motion capture marker data. *ACM Transactions on Graphics (TOG)* 2005; **24**(3): 417–425.
11. Teran J, Sifakis E, Irving G, Fedkiw R. Robust quasistatic finite elements and flesh simulation. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA05)*. ACM, New York, NY, USA, 2005; 181–190.
12. Sifakis E, Selle A, Robinson-Mosher A, Fedkiw R. Simulating speech with a physics-based facial muscle model. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA06)*. Eurographics Association, Switzerland, 2006; 261–270.
13. Zhang Y, Prakash EC, Sung E. Efficient Modeling on an anatomy-based face and fast 3D facial expression synthesis. *Computer Graphics Forum* 2003; **22**: 159–169.
14. Kähler K. *3D Facial Animation- Recreating Human Heads with Virtual Skin, Bones, and Muscles*. Verlag, 2007.
15. Müller M. Hierarchical position based dynamics. In *Virtual Reality Interactions and Physical Simulations (VRIPhys2008)*, Grenoble, November 2008.
16. Micchelli CA. Interpolation of scattered data: distance matrices and conditionally positive definite functions. *Constructive Approximations* 1986; **2**: 11–22.
17. Fang S, Raghavan R, Richtsmeier J. Volume Morphing Methods for landmark based 3D image deformation. In *SPIE International Symposium on Medical Imaging*, Newport Beach, CA, February 1996; volume 2710, 404–415.
18. Pandzic IS, Forchheimer R (eds). *MPEG-4 Facial Animation—the Standard, Implementation and Applications*, (1st edn). John Wiley & Sons, LTD, Linköping, Sweden, 2002.
19. Ekman P, Friesen WV, Ellsworth P. *Emotion in the Human Face*. Pergamon Press, Elmsdorf, NY, 1972.
20. Valstar MF, Pantic M. Induced disgust, happiness and surprise: an addition to the MMI facial expression database. In *Int'l Conference Language Resources and Evaluation, Workshop on EMOTION*, Malta, May 2010; 65–70.
21. Visage Technologies, January 2012. http://www.visagetechnologies.com.

AUTHORS' BIOGRAPHY

Marco Fratarcangeli is an Assistant Professor in Computer Science at the Department of Computer and Systems Science 'A. Ruberti' at University of Rome 'La Sapienza', Italy, where he also received his MSc (Laurea) in 2003 in Computer Engineering. He obtained his PhD in 2009 in the field of physically based animation of virtual faces. Between 2004 and 2006, he was a research assistant at the Image Coding Group at the Linköping University, Sweden. From 2008 to 2011, he worked as technical lead at Taitus Software, focusing on the development of cross-platform visualization tools for planning and analysis of Earth Observation missions. His research interests focus on physically based simulation of deformable bodies, face and body animation for virtual characters, collision handling, and geometric algorithms.