Hierarchical Face Modeling and Fast 3D Facial Expression Synthesis

Yu Zhang[†], Edmond C. Prakash[††] and Eric Sung[†]

[†]School of Electrical and Electronic Engineering, [††]School of Computer Engineering
Nanyang Technological University, Singapore 639798

Abstract. This paper presents a new hierarchical facial model that conforms to the human anatomy for realistic and fast 3D facial expression synthesis. The facial model has a skin/muscle/skull structure. The deformable skin model uses a kind of nonlinear springs to directly simulate the nonlinear visco-elastic behavior of soft tissue, and a new kind of edge repulsion springs is developed to prevent model collapse. The incorporation of the skull extends the scope of facial motion and facilitates facial muscle construction. The construction of facial muscles is achieved by using an efficient muscle mapping approach that ensures different muscles to be located at the anatomically correct positions. For computational efficiency, we devise an adaptive simulation algorithm which uses either a semi-implicit integration scheme or a quasi-static solver to compute the relaxation by traversing the designed data structures in a breadth-first order. The algorithm runs in real-time and has successfully synthesized realistic facial expressions.

1 Introduction

The human face is endowed with a complex anatomical structure; there are a multitude of subtle expressional variations on the face; moreover, we as humans have an uncanny sensitivity to facial appearance. Due to these factors, until now, synthesizing realistic 3D facial expressions is still a tedious and difficult task. In order to tackle this challenge, two different aspects, namely the *structure modeling* and the *deformation modeling*, have to be addressed: the first one relates to the development of accurate representations of the 3D geometry of the facial structure. The second one applies to the efficient deformation of the facial skin shape to generate flexible expressions.

1.1 Background and Previous Work

Since the pioneering work of Parke [13] in the early 70's, researchers have investigated techniques to generate realistic face models and animation [15]. Early works restricted themselves to geometric models and simulation methods, such as key-frame interpolation animation [11, 13], parametric models [2, 4, 7, 14, 20], abstract muscle actions [19] and control-point driven animation [8, 17]. To extend the range of possible facial deformations, during the last decade, some researchers turned to physically-based methods, using either a particle system or a continuous system. Platt et al. [16] developed an early structural facial model in which the skin is modeled as a tension net and muscles are groups of linear elastic arcs underlying skin surface. Based on Platt's model, Essa et al. [6] developed a system to estimate muscle actuation corresponding to a given expression using feedback control theory. Terzopoulos et al. [18] have extended the physical model, introducing a deformable lattice structure model for the facial tissue. Lee et al. [12] adapted generic polygonal facial representations to the range data and derived facial motions from the behaviors of the muscle structures. The facial model developed by Wu et al. [21] focuses on the plastic and viscoelastic properties of the skin to generate wrinkles and skin aging effects. The finite element method is also employed for more accurate calculation of skin deformation, especially for potential medical applications [9, 10].

However, there are several issues with existing facial models that still lack adequate solutions:

Spring Approximation: The facial models based on the particle system use linear springs. Though this assumption simplifies somewhat the equation of motion at each node, it is undesirable for accurate simulation of the real tissue that has a nonlinear stress-strain relationship. Moreover, there is a tendency of spring mesh to collapse under relatively large compression, which is due to the fact that linear springs used can be compressed fully.

Skull Base: Skull plays an important role in the articulation of the jaw. The incorporation of the skull structure into facial model can also facilitate the construction of contractile muscles at anatomically correct positions. However, in previous work the skin-skull interface has not been emphasized.

Facial Muscle Construction: Automatic construction of facial muscles in the face remains a problem. In some previous work, although the skin and muscle structures have been modeled, the correctness of the locations of facial muscles can not be verified due to either the absence of the underlying skull or the interactive muscle insertion. Particularly, the latter requires the user to go through a series of trials in the 3D space, clearly making it the bottleneck in face modeling.

Fast Simulation: Research on physical facial animation has generally relied on explicit numerical integration (such as Euler's method or Runge-Kutta methods) to advance the simulation. Unfortunately, such methods may lead to slow simulation since very small time steps are required to ensure stability. Moreover, for the fast computation of the underlying deformation and force model, efficient data structures and algorithms that can process high-resolution models have not been developed.

1.2 Our Approach

The goal of our work is to achieve a realistic facial expression synthesis from the anatomical perspective and executing in an acceptable time. In this paper, we propose a new hierarchical facial model which incorporates a physically-based approximation to facial skin, a set of anatomically-motivated facial muscle actuators and underlying skull. The skin is modelled by taking into account the nonlinear stress-strain relationship of the soft tissue. In order to prevent model collapse, a kind of edge repulsion springs is used to apply additional pressure forces to the skin nodes. With such springs, the triangle element topology of the skin mesh can be preserved effectively during the dynamic simulation without a heavy computational load. The 3D facial model incorporates a skull structure which extends the scope of facial motion and facilitates facial muscle definition. The facial muscle construction is achieved by using an efficient muscle mapping approach that ensures the muscles to be automatically located between the skin and skull layers. For efficient simulation, we propose an adaptive simulation algorithm. It employs either a dynamic or a quasi-static solver by taking advantages of the facts that facial deformations are local. By propagating forces in an ordered fashion through the facial mesh, we in effect concentrate computational load to the facial regions that undergo significant deformations. The algorithm runs at an interactive rate with flexib facial expressions to be synthesized.

This paper is organized as follows: Section 2 presents the biomechanical facial skin model. Section 3 describes the skull incorporation; a muscle mapping approach for facial muscle construction is also illustrated. Section 4 elaborates on our adaptive simulation algorithm used for relaxation. The simulation results are shown in Section 5. Section 6 gives concluding remarks and our future work.

2 Physical Facial Skin Model

We have constructed a face model that consists of 7,373 vertices and 13,200 triangles. Additional geometrical models of the eyes have been combined alone with the surface model to enhance the overall realism (see Fig 1).

In order to physically simulate the deformation of the facial skin tissue, we use the mechanical law of the mass-

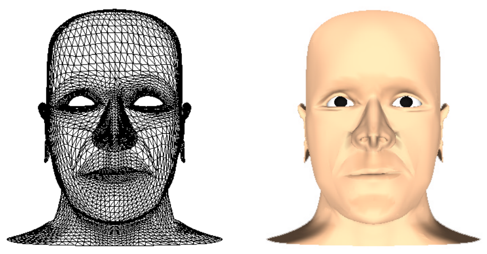


Figure 1: (a) Wire frame face model; (b) Face model with Gouraud shading.

spring system. A spring mesh is created from the adaptively reduced triangle mesh: each vertex corresponds to a point mass with mass density m and along each edge of a triangular element there is a spring connecting two adjacent mass points. Since such kind of springs construct the facial mesh, we call them *structural spring* (SS). In order to simulate nonlinear deformation of the skin, we use a nonlinear function to describe the stress-strain relationship directly. Suppose an arbitrary soft-tissue point $\mathbf{x}_i$ is connected to one of its neighbors $\mathbf{x}_j$ by a spring with rest length d_{ij}. Let $\Delta\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$, we introduce a function $K(\mathbf{x}_i, \mathbf{x}_j)$ to modulate a constant spring stiffness k_0:

$$K(\mathbf{x}_i, \mathbf{x}_j) = (1 + (\mid \Delta\mathbf{x}_{ij} \mid - d_{ij})^2)^\alpha k_0 \tag{1}$$

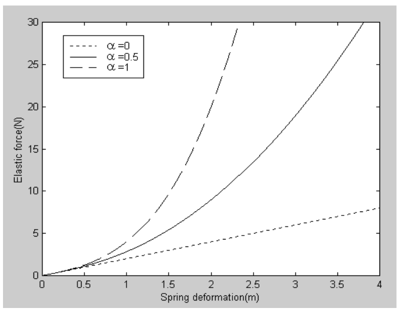
and the spring force generated by an SS is:

$$f_{SS}(\mathbf{x}_i) = -K(\mathbf{x}_i, \mathbf{x}_j)\frac{(\mid \Delta\mathbf{x}_{ij} \mid - d_{ij})}{\mid \Delta\mathbf{x}_{ij} \mid}\Delta\mathbf{x}_{ij} \tag{2}$$

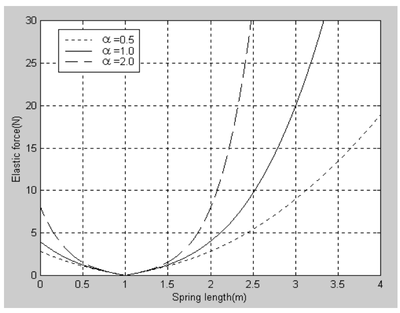
In Eq. 1 α is the *nonlinearity factor* controlling the modulation. By assigning different values to α, function $K(\mathbf{x}_i, \mathbf{x}_j)$ can be chosen to model linear or nonlinear stress-strain relationship. Fig. 2 (a) illustrates the stress-strain relationship for different values of α.

However, the deformable skin model arising from the previous definition can exhibit undesirable behavior under certain conditions. The worst aspect of this is the tendency of the skin model to collapse under relatively large compression. In order to prevent model collapse, we introduce a new type of spring named *edge repulsion spring* (ERS). Consider a mass point $\mathbf{x}_0$ with a neighborhood formed by $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$, $\mathbf{x}_4$, $\mathbf{x}_5$, $\mathbf{x}_6$, as shown in Fig. 3. In each triangular element composed by $\mathbf{x}_0$ and two neighboring mass points, we insert an ERS.

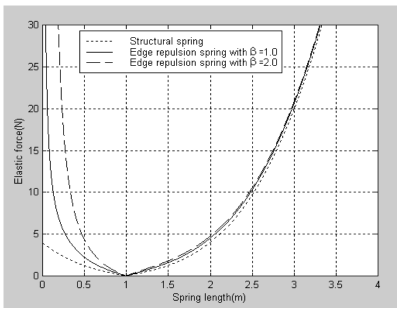
The implementation of this kind of springs is done by attaching the springs linking each vertex $\mathbf{x}_i$ with $\mathbf{p}_j$ which is the projection of $\mathbf{x}_i$ onto one of its opposite edges. The reasoning behind introducing this kind of spring is that as the standard element is compressed its resistance to further compression reduces since the springs become less and less



(a)



(b)



(c)

Figure 2: (a) The stress-strain relationship of SS with different values of α ($k_0 = 2.0$); (b) Magnitude of the force generated by SS with different values of α ($k_0 = 2.0, d_{ij} = 1.0$); (c) Magnitude of the force generated by the SS and ERS ($\alpha = 1.0, k_0 = 2.0, d_{ij} = 1.0, \varsigma = 1.0$).

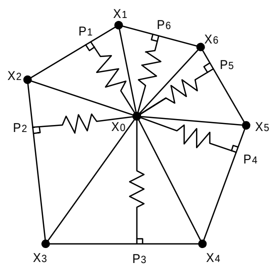


Figure 3: The ERSs of $\mathbf{x}_0$.

aligned with crush direction. However, the force exerted by the ERS is always away from the edge and therefore applies constraint on the motion of the mass point. But even the insertion of these edge repulsion springs can not eliminate the element collapse under a suitably large force if the spring force function of the SS (Eq. 2) is employed. This is due to the fact that the elastic force resisting compression is finite as spring length equals to zero (see Fig. 2 (b)), which means such springs can be compressed fully. In reality as the spring length tends towards zero the spring will exert reaction forces which tend to infinity. In order to model this nonlinear effect, we revise Eq. 2 to obtain:

$$f_{ERS}(\mathbf{x}_i, \mathbf{p}_j) = (-K(\mathbf{x}_i, \mathbf{p}_j)(\mid \Delta\mathbf{x}_i\mathbf{p}_j \mid - d_{ij}) + \frac{\varsigma d_{ij}^{\beta}}{\mid \Delta\mathbf{x}_i\mathbf{p}_j \mid^{\beta}} - \varsigma)\frac{\Delta\mathbf{x}_i\mathbf{p}_j}{\mid \Delta\mathbf{x}_i\mathbf{p}_j \mid} \qquad (3)$$

where

$$\Delta\mathbf{x}_i\mathbf{p}_j = \mathbf{x}_i - \mathbf{p}_j \qquad (4)$$

d_{ij} is the rest length of ERS, ς is the *repulsion scaling factor*, β controls the magnitude of the repulsion force generated by the ERS and hence the name *repulsion strength factor*. Eq. 3 combines the term for an SS with a nonlinear term which tends to infinity as the spring length tends to zero. The term ς is also needed to ensure that the spring force is zero when an ERS is in its rest state.

A graph of the force exerted by an ERS as a function of the spring length with different values of β is illustrated in Fig. 2 (c), together with the force generated by an SS. It can be seen that the reaction force becomes infinite as the length of the compressed ERS tends towards zero. The new mass-spring mesh has been constructed with each triangular element consisting of three SSs along the edges and several ERSs for each vertex. The new elastic force function for each mass point thus is given by:

$$\mathbf{F}_{ela}(\mathbf{x}_i) = \sum_{j \in \mathcal{N}_i} f_{SS}(\mathbf{x}_i, \mathbf{x}_j) + \sum_{j \in \Omega_i} f_{ERS}(\mathbf{x}_i, \mathbf{p}_j) \qquad (5)$$

where $\mathcal{N}_i$ and Ω_i are the index set of neighboring mass points and edge projecting points of $\mathbf{x}_i$ respectively.

3 Skull Model and Muscle Construction

3.1 Geometric Model of Skull and Jaw

We use a generic skull model to map general anatomical attributes to the facial surface. It consists of an immovable skull and a rotating jaw. In contrast to the facial expressions caused by muscle activation, the jaw rotation is obtained by manipulating the skull geometry. In accordance with the activation of the real jaw, the jaw model can rotate around X and Y axes and can move to a parallel direction alone the Z axis. The motion of the jaw is realized by a 3D coordinate transformation.

When this generic skull model is available, the next step is to adapt it to fit the face mesh geometry. Since this task cannot be fully automated, user interaction is necessary. In the process of skull fitting, currently affine transformations - rotation, translation and nonuniform scaling are applied on the skull model by the user interactively. Fig. 4 shows facial skin mesh and the shape of the skull after alignment. While it is generally not possible to match a generic skull to different human faces in this way, the approximation is good enough for defining facial muscles and assigning skin regions to the skull or jaw.

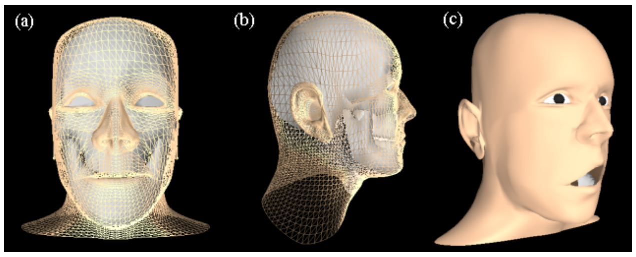


Figure 4: (a) and (b): Front and side views of generic skull model aligned with facial skin mesh; (c) Simulated jaw movement.

3.2 Automatic Facial Muscle Construction

In our previous work [22], we have developed three kind of muscle models to simulate the distribution of the muscle force applied on the skin. We use a muscle mapping approach for facial muscle construction. By using OpenGL, we first save a bitmap from the color buffer. It records the RGB values of the facial surface. We then specify a set of key points on this bitmap to identify the ideal locations of the facial muscles that should be designed on it (see Fig. 5). Based on the Facial Action Coding System (FACS)[5], we select 23 major functional facial muscles to simulate facial expressions. For a linear or sheet muscle, the positions of the attachment and insertion points of its central muscle fiber completely define the location of the muscle. We mark the attachment and insertion points of the linear and sheet muscles by using two different colors. In Fig. 5, red key points are muscle insertion points, the green ones are muscle attachment points. For each muscle, its attachment and insertion points are connected by a white line. For a sphincter muscle, we mark two blue points to represent the ends of its two semi-axes (see Fig. 5). The positions of the key points are marked once on the reflectance image and the resulting image is named *facial muscle image*.

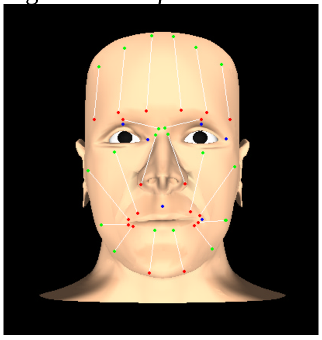


Figure 5: Facial muscle image with marked muscle positions.

Once the marks are all made, the texture coordinates of each facial mesh vertex in the facial muscle image are calculated based on an orthographic projection and the facial muscle image is mapped automatically to the 3D face. In order to locate all marked points on the 2D image, we use a cylindrical projection to map rendered 3D face to a 2D image plane. In the resulting 512×512 *cylindrical facial muscle image* (as shown in Fig. 6 (a)), each pixel value represents the surface color of the texture mapped facial surface in cylindrical coordinates with corresponding longitude (0-360 degrees) and latitude. As we have marked key points in distinct colors, we can easily detect them and calculate their image positions in the cylindrical facial muscle image.

To calculate 3D positions of red key points representing muscle insertion points we create a mapping from the 3D facial mesh to 2D image space by using the same cylindrical projection (see Fig. 6 (b)). All the image coordinates (s, t) of the facial surface vertices are automatically calculated. Each detected red key point p can be located inside one of the triangles of the projected triangular mesh. The image position of p can be represented by its barycentric coordinates $\beta = (\beta_1, \beta_2, \beta_3)$ inside this triangle $\triangle(v_1, v_2, v_3)$:

$$p = \sum_{i=1}^{3} \beta_i v_i \qquad (6)$$

with $\sum_i^3 \beta_i = 1$ and $\beta_i \geq 0$ for all i. Each triangle in the 2D image space corresponds to a triangle on the 3D facial surface. The 3D position of p can be calculated as a linear interpolation of the positions of three vertices, weighted using its barycentric coordinates in the 2D triangle:

$$P = \sum_{i=1}^{3} \beta_i V_i \qquad (7)$$

where P, V_i are the 3D positions of the key point p and vertex v_i in 2D image space respectively. By applying cylindrical projection to the skull model (see Fig. 6 (c)) and using the same position interpolation approach, we can calculate the 3D positions of muscle attachments points that should be located on the skull surface. To automatically find the correspondence between the attachment and insertion points of each muscle, we use a simple image processing algorithm not worth being described here to trace the white line connected between two key points in the cylindrical facial muscle image.

By exploiting the relative image positions of six blue key points (e.g., two key points representing *Orbicularis Oris* have smaller v coordinate values et al.), we can easily identify which key point corresponds to which sphincter muscle. For each blue key point, its 3D positions on both the facial skin and skull surfaces are computed using the method described above. Its final position is obtained by averaging these two positions. Once the 3D positions of all the key points of facial muscles are calculated and the correspondence between key points and muscles are found, the facial muscles are automatically constructed. Fig. 7 shows

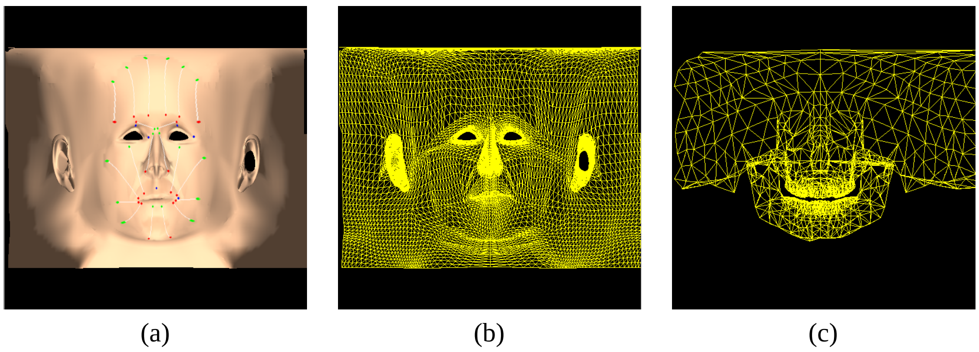


Figure 6: Mapping between the 3D space and 2D image plane based on a cylindrical projection: (a) cylindrical projection of the texture mapped facial model; (b) cylindrical projection of the facial mesh; (c) cylindrical projection of the skull mesh.

all the facial muscles automatically constructed in our facial model using this approach. In this figure, for each muscle, the dark and light points represent muscle insertion point and muscle attachment point respectively, the line connecting between them represents the central muscle fiber.

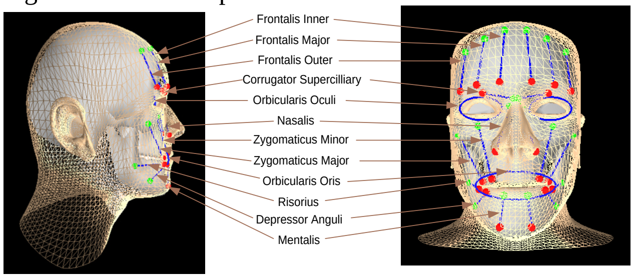


Figure 7: Automatically constructed muscles in the face.

4 Facial Dynamics and Numerical Simulation

Based on the Lagrangian dynamics, the deformable facial model equations of motion can be expressed in 3D vector form by the second-order ordinary differential equation (ODE) of type:

$$\mathbf{M}\frac{d^2\mathbf{x}(t)}{dt^2} + \mathbf{D}\frac{d\mathbf{x}(t)}{dt} + \mathbf{K}\mathbf{x}(t) = \mathbf{F}_{mus}(\mathbf{x}(t)) \tag{8}$$

We can take the elastic force expression as an external force $\mathbf{F}_{ela}(\mathbf{x}(t), \mathbf{K}) = \mathbf{K}\mathbf{x}(t)$, and take $\mathbf{F}_{ela}$ to the right hand side of the Eq. 8. This new form of the equation will simplify the formulation procedure.

$$\mathbf{M}\frac{d^2\mathbf{x}(t)}{dt^2} + \mathbf{D}\frac{d\mathbf{x}(t)}{dt} = \mathbf{F}_{mus}(\mathbf{x}(t)) - \mathbf{F}_{ela}(\mathbf{x}(t), \mathbf{K}) \tag{9}$$

Given *n* nodes $\mathbf{x}$ represents a 3n vector of nodal displacement, $\mathbf{M}$, $\mathbf{D}$ and $\mathbf{K}$ are $3n \times 3n$ diagonal matrices describing the mass, damping and stiffness between nodes in the skin mesh respectively. $\mathbf{F}_{mus}$ and $\mathbf{F}_{ela}$ are vectors of dimension 3*n* and represent muscular and elastic forces respectively.

The above system is solved by iterative stepwise processing of each individual equation. To simplify notation, we will only consider the governing equation for a single mass m_i. In order to derive the method we divide the second order ODE for a single mass m_i into a system of first ODEs by introducing the velocity function $\mathbf{v}_i(t)$.

$$\begin{cases} \frac{d\mathbf{x}_i(t)}{dt} = \mathbf{v}_i(t) \\ \frac{d\mathbf{v}_i(t)}{dt} = \frac{\mathbf{F}_{mus}(\mathbf{x}_i(t)) - \mathbf{F}_{ela}(\mathbf{x}_i(t), \mathbf{K}) - \mathbf{D}\mathbf{v}_i(t)}{m_i} \end{cases} \tag{10}$$

It has been shown that implicit integration methods are superior to explicit ones in case of stiff equation systems [1, 3]. They enhance the simulation efficiency by taking large time steps while maintaining the stability of the integration. Using an implicit method we describe the calculation of $\mathbf{x}_i^{n+1}$ and $\mathbf{v}_i^{n+1}$ depending on the given states $\mathbf{x}_i^n$ and velocity $\mathbf{v}_i^n$ at time t_n, and the new, unknown states $\mathbf{x}_i^{n+1}$ and $\mathbf{v}_i^{n+1}$ at time t_{n+1}.

$$\begin{cases} \mathbf{v}_i^{n+1} = \mathbf{v}_i^n + \Delta t \frac{\mathbf{F}_{mus}(\mathbf{x}_i^{n+1}) - \mathbf{F}_{ela}(\mathbf{x}_i^{n+1}, \mathbf{K}) - \mathbf{D}\mathbf{v}_i^{n+1}}{m_i} \\ \mathbf{x}_i^{n+1} = \mathbf{x}_i^n + \Delta t \mathbf{v}_i^{n+1} \end{cases} \tag{11}$$

where

$$\mathbf{F}_{ela}(\mathbf{x}_i^{n+1}, \mathbf{K}) = \sum_{j \in \mathcal{N}_i} P(\mathbf{x}_i^{n+1} - \mathbf{x}_j^{n+1}) + \sum_{j \in \Omega_i} Q(\mathbf{x}_i^{n+1} - \mathbf{p}_j^{n+1}) \tag{12}$$

$$P = -K(\mathbf{x}_i^{n+1}, \mathbf{x}_j^{n+1}) \frac{|\ \Delta\mathbf{x}_{ij}^{n+1}\ | - d_{ij}}{|\ \Delta\mathbf{x}_{ij}^{n+1}\ |} \tag{13}$$

$$Q = (-K(\mathbf{x}_i^{n+1}, \mathbf{p}_j^{n+1})(|\Delta\mathbf{x}_i^{n+1}\mathbf{p}_j^{n+1}| - d_{ij}) + \frac{\varsigma d_{ij}^{\beta}}{|\Delta\mathbf{x}_i^{n+1}\mathbf{p}_j^{n+1}|^{\beta}} - \varsigma) \frac{1}{|\Delta\mathbf{x}_i^{n+1}\mathbf{p}_j^{n+1}|} \tag{14}$$

$$\Delta\mathbf{x}_{ij}^{n+1} = \mathbf{x}_i^{n+1} - \mathbf{x}_j^{n+1}, \quad \Delta\mathbf{x}_i^{n+1}\mathbf{p}_j^{n+1} = \mathbf{x}_i^{n+1} - \mathbf{p}_j^{n+1} \tag{15}$$

$$K(\mathbf{x}_i^{n+1}, \mathbf{x}_j^{n+1}) = (1 + (|\ \Delta\mathbf{x}_{ij}^{n+1}\ | - d_{ij})^2)^{\alpha} k_0 \tag{16}$$

$$K(\mathbf{x}_i^{n+1}, \mathbf{p}_j^{n+1}) = (1 + (|\ \Delta\mathbf{x}_i^{n+1}\mathbf{p}_j^{n+1}\ | - d_{ij})^2)^{\alpha} k_0 \tag{17}$$

Note that, one has to solve a nonlinear system of equations for the variables $\mathbf{x}_i^{n+1}$ and $\mathbf{v}_i^{n+1}$ at every time-step. To bypass the computation of this system, we substitute $\mathbf{x}_i^{n+1}$ in the upper row of Eq. 11 by insertion of the bottom equation of (11) and solve it for the velocity $\mathbf{v}_i^{n+1}$. In this case, the solution of the initially implicit method can be expressed explicitly. With some algebra we get:

$$\begin{cases} \mathbf{v}_i^{n+1} = \\ \frac{m_i \mathbf{v}_i^n + \Delta t(\mathbf{F}_{mus}(\mathbf{x}_i^{n+1}) - \sum_{j\in\mathcal{N}_i} P(\mathbf{x}_i^n - \mathbf{x}_j^{n+1}) - \sum_{j\in\Omega_i} Q(\mathbf{x}_i^n - \mathbf{p}_j^{n+1}))}{m_i + \Delta t D + (\Delta t)^2(\sum_{j\in\mathcal{N}_i} P + \sum_{j\in\Omega_i} Q)} \\ \\ \mathbf{x}_i^{n+1} = \mathbf{x}_i^n + \Delta t \mathbf{v}_i^{n+1} \end{cases} \tag{18}$$

The second equation in Eq. 18 only depends on the variable $\mathbf{v}_i^{n+1}$ and can therefore be computed after evaluating the first equation. But in the first equation, the functions of muscular force and elastic force still depend on $\mathbf{x}_i^{n+1}$. Since they are nonlinear functions, it turns out to be too expensive to extract $\mathbf{x}_i^{n+1}$ at every time-step. To simplify evaluation, we estimate $\mathbf{x}_i^{n+1}$ using an explicit Euler step.

$$\mathbf{x}_i^{n+1} = \mathbf{x}_i^n + \Delta t \mathbf{v}_i^n \tag{19}$$

By inserting this estimated position value into the first equation in Eq. 18 we then solve for new velocity $\mathbf{v}_i^{n+1}$. Given $\mathbf{v}_i^{n+1}$, we trivially compute $\mathbf{x}_i^{n+1}$. This leads to a semi-implicit approach that solves of the direction of the mass' motion implicitly and for its magnitude explicitly.

From the analysis of various expressions generated with muscle's contraction we learned that the major part of the tissue deformation occurs in the direct vicinity of the muscle influence area (the reader is referred to our previous work [22] for its definition). In other words, mass nodes further from this region take a smaller influence on the nodal displacement. This observation suggests an adaptive simulation. For an efficient implementation of this feature, we divide node set V of facial skin model into three subsets: V_d, V_s and V_q. *Dynamic node set* V_d represents the set of nodes that are in the muscle influence areas. The equilibrium position of each node in V_d is calculated by using semi-implicit integration method described earlier to solve Eq. 9. In the facial model, there are a large number of nodes that remain static in the facial animation. They are grouped into the *static node set* V_s. In the offline precomputation, we calculate the deformation of the entire skin mesh by using the semi-implicit method. If the final displacement of a node is less than a small pre-specified threshold, it's then considered as a node of V_s. *Quasi-static node set* V_q is a collection of nodes that receive no muscle forces but are still displaced to their new positions due to the propagation of unbalanced forces through the mesh. The displacement of each node in V_q is then calculated by using the following quasi-static algorithm:

1. Acquire the positions of all the dynamic nodes.
2. Compute the residual force $\mathbf{f}(\mathbf{x}) = \mathbf{F}_{ela}(\mathbf{x})$ exerted on each quasi-static node.
3. Displace the quasi-static node along the acquired residual force.

This scheme will be most advantageous when nodes are traversed in an order starting at the dynamic nodes and expending towards the quasi-static nodes farther away from any dynamic node. In our simulation, the traversal of the nodes operates in a wave-propagation order which is precomputed by a breadth-first scan of the skin mesh as shown in Fig. 8. In this figure, q denotes the topological distance of a node from the outermost dynamic node measured in terms of the smallest number of edges between them. The outcome of node ordering is a list of nodes such that if index i appears before index j then the topological distance q_i is less than or equal to q_j.

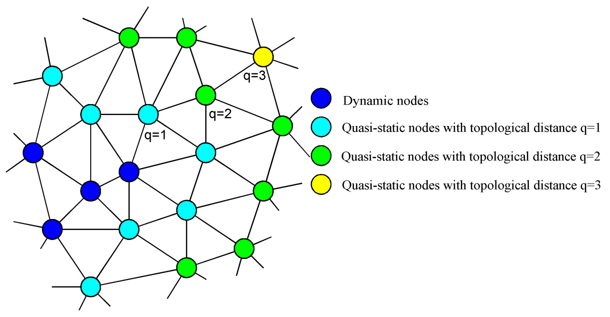

q=3
q=2
q=1
Dynamic nodes
Quasi-static nodes with topological distance q=1
Quasi-static nodes with topological distance q=2
Quasi-static nodes with topological distance q=3

Figure 8: Breadth-first traversal strategy for quasi-static simulation.

The data structure of three node subsets enables adaptive simulation to be implemented efficiently. The synthesis of different expressions only requires re-grouping three node sets and re-invoking quasi-static node ordering to compute a new node list. In this way, the number of nodes dynamically or quasi-statically treated (hence the computational cost) adapts automatically.

5 Results

The implementation of our approach has been integrated into a facial animation system which is programmed with C++/OpenGL and runs on an Intergraph Zx10 with dual Pentium III 730MHz, 512MB memory. In the simulation, the facial model with physical attributes is deformed to create various expressions. Fig. 9 illustrates the neutral face and typical expressions synthesized by our system and compares each of them with the corresponding actual one generated by a subject (leftmost image of each example). For clarity, different views of the face model are shown. The sequences of images presented in Fig. 10 show several frames of the dynamic deformation of the face in synthesizing each expression.

Expressions	N_d	N_q	N_s	T_p(s)	Computational time of an iteration (ms)			Frame-rate (fps)
					T_d	T_q	T_r	
Happiness	1048	2027	4298	20.6	16.7	7.0	5.0	34.8
Anger	1882	2294	3197	24.2	29.4	8.0	5.0	23.6
Sadness	2193	2163	3017	26.7	34.2	7.5	5.0	21.4
Disgust	2016	2118	3239	22.7	32.2	7.3	5.0	22.5
Fear	1562	2208	3603	21.6	25.0	7.6	5.0	26.6

Table 1: System performance. Notation: number of dynamic nodes (N_d), number of quasi-static nodes (N_q), number of static nodes (N_s), precomputing time (T_p), dynamic simulation time (T_d), quasi-static simulation time (T_q) and rendering time (T_r).

Table 1 shows the quantitative evaluation of the system performance by breaking the simulation into different processes. The time consumption of the quasi-static simulation and rendering is relatively small compared to that of the dynamic simulation. For the expression generated by the contractions of more muscles such as sadness, although the number of dynamic nodes increases, the framerate can still reach about 21 fps. We can clearly verify that the proposed adaptive simulation algorithm of section 4 gracefully accelerates the simulation speed as a function of N_d and N_q.

6 Conclusion and Future Work

A hierarchical facial model constructed from anatomical perspective for realistic 3D facial expression synthesis has been proposed. The deformable skin model takes into account the nonlinear stress-strain relationship of the real skin and uses a kind of edge repulsion spring to prevent mesh collapse. The incorporation of the skull extends the scope of facial motion and facilitates facial muscle construction. By using a muscle mapping approach, different facial muscles are constructed at the anatomically correct positions in an efficient way. An adaptive simulation algorithm has been proposed to accelerate numerical simulation. It uses either a semi-implicit integration method or a quasi-static solver for the relaxation, thus concentrating computational cost to the facial regions that undergo significant deformations.

We would like to extend the single-layer skin to a volume model by using tetrahedral mesh which gives more geometrical flexibility for the efficient modeling of the complex anatomical structure. Currently the fitting of the generic skull model to the skin mesh is achieved by interactively adjusting the parameters of spatial transformation. A more sophisticated approach will be developed for an automatic and more precise fit. Additionally, in order to synthesize more complex and individualized expressions, we will track the actual facial motion and apply modal analysis techniques to find natural control parameters.

References

[1] D. Baraff and A. Witkin, "Large steps in cloth simulation", *Proc. SIGGRAPH'98,* vol.32, pp. 43-54, July 1998.

[2] D. DeCarlo, D. Metaxas, and M. Stone, "An anthropometric face model using variational techniques", *Proc. SIGGRAPH'98,* vol.32, pp. 67-74, July 1998.

[3] Desbrun, M., Schroder, P., Barr, A. "Interactive animation of structured deformable objects", *Proc. Graphics Interface'99,* Kingston, 1999.

[4] S. DiPaola, "Extending the range of facial types", *Journal of Visualization and Computer Animation,* 2(4): 129-131, 1991.

[5] P. Ekman and W. V. Friesen, *Facial Action Coding System,* Consulting Psychologists Press Inc., 577 College Avenue, Palo Alto, California 94306, 1978.

[6] I. Essa and S. Basu, "Modeling, tracking and interactive animation of facial expressions and head movements using input from video", *Proc. Computer Animation'96,* pp. 68-79, June 1996.

[7] T. Goto, M. Escher, C. Zanardi, and N. Magnenat-Thalmann, "MPEG-4 based animation with face feature tracking", *Proc. Erographics Workshop on Computer Animation and Simulation'99,* pp. 89-98,1999.

[8] P. Kalra, A. Mangili, N. Magnenat-Thalmann, D. Thalmann, "Simulation of facial muscle actions based on rational free form deformations", *Proc. EUROGRAPHICS'92,* pp. 59-69. Cambridge, 1992.

[9] E. Keeve, S. Girod, P. Pfeifle, and B. Girod, "Anatomy-based facial tissue modelling using the finite element method", *Proc. IEEE Visualization'96,* pp. 21-28, 1996.

[10] Koch R., Gross M., Carls F., Buren D., Fankhauser G. and Parish Y., "Simulating facial surgery using finite element models", *Proc. SIGGRAPH'96,* vol.30, pp. 421-428, August 1996.

[11] Cyriaque Kouadio, Pierre Poulin, Pierre Lachapelle, "Real-time facial animation based upon a bank of 3D facial expression", *Proc. Computer Animation'98,* pp. 128-136, 1998.

[12] Y. Lee, D. Terzopoulis, and K. Waters, "Realistic modeling for facial animation", *Proc. SIGGRAPH'95,* vol.29, pp. 55-62, August 1995.

[13] F. I. Parke, *Computer generated animation of faces,* Master's thesis, university of Utah, Salt Lake City, June 1972.

[14] F. I. Parke, "Parameterized models for facial animation", *IEEE Computer Graphics and Application,* 2(9): 61-68, November 1982.

[15] F. I. Parke and K. Waters, *Computer Facial Animation,* AK Peters, Wellesley, MA, 1996.

[16] Platt S., Badler N., "Animating facial expressions", *Proc. SIGGRAPH'81,* vol.15, pp. 245-252, 1981.

[17] H. Tao and T. S. Huang, "Explanation-based facial motion tracking using a piecewised bezier volume deformation model", *CVPR'99,* pp. 611-617, 1999.

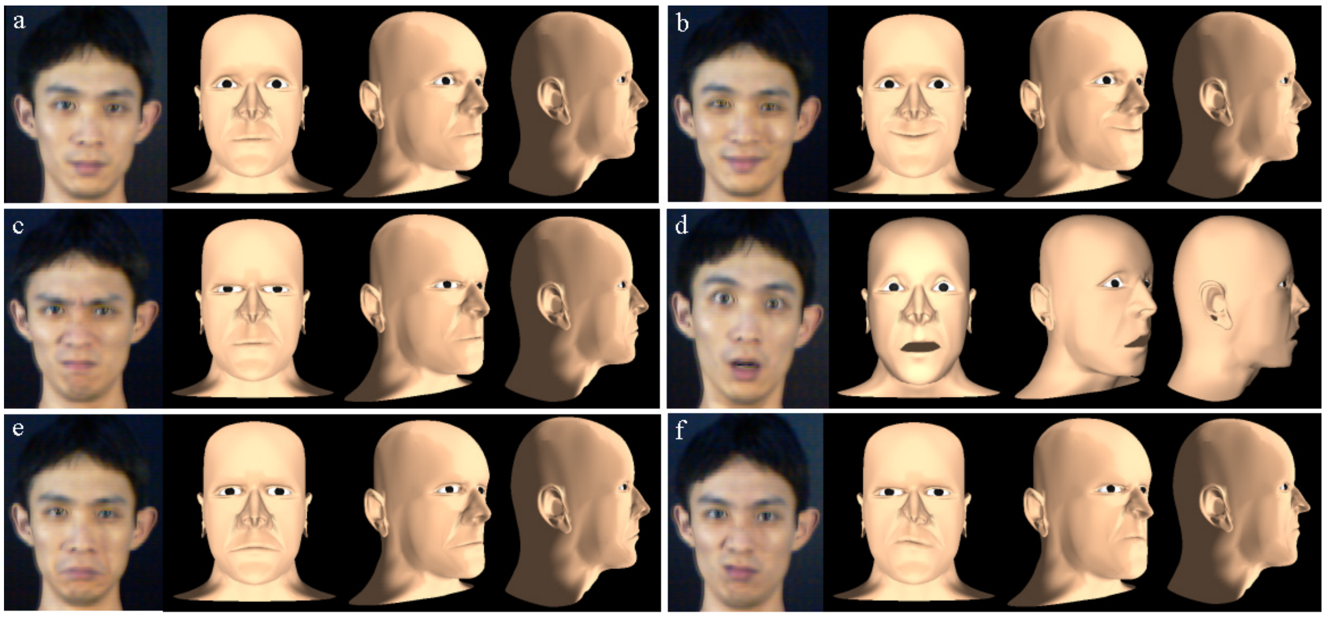


Figure 9: Synthesized typical expressions compared with the actual ones: (a) neutral face; (b) happiness; (c) anger; (d) surprise; (e) sadness and (f) disgust (3 views of each synthesized expression).

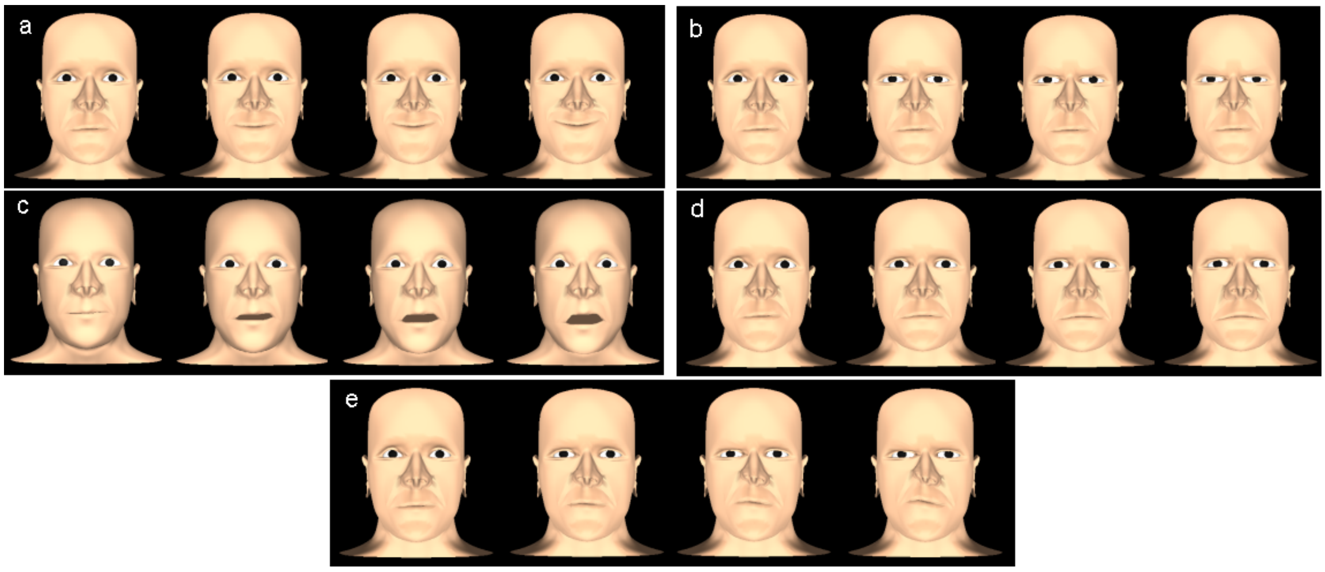


Figure 10: Deformation of the face in synthesizing different expressions. The leftmost image corresponds to the initial state of the face and the rightmost image to the steady state. (a) happiness; (b) anger; (c) surprise; (d) sadness and (e) disgust.

[18] D. Terzopoulus and K. Waters, “Physically-based facial modeling, analysis and animation”, *The Journal of Visualization and Computer Animation* , vol.1, pp. 73-80, 1990.

[19] N. Magnenat-Thalmann, E. Primeau, and D. Thalmann, “Abstract muscle action procedures for human face animation”, *The Visual Computer,* 3(5):290-297, 1988.

[20] N. Magnenat-Thalmann, H. Minh, M. deAngelis and D. Thalmann, “Design, transformation and animation of human faces”, *The Visual Computer,* 5:32-39, 1989.

[21] Y. Wu, P. Kalra, L. Moccozet, and Nadia M. Thalmann, “Simulating wrinkles and skin aging”, *The Visual Computer*, 15(4):183-198, 1999.

[22] Y. Zhang, E. C. Prakash and E. Sung. “Animation of facial expressions by physical modeling.” *Proc. Eurographics 2001, Short Presentations*, pp. 335-345, Sept. 2001.