TR-M-0047

Construction of Facial Tissue using Cyberware and Computer Tomography Data

ファイサル　ズバイル　クェレシ
Faisal Zubair Qureshi

1999.8.13

ATR 知能映像通信研究所

Construction of Facial Tissue using Cyberware and Computer Tomography Data.

Faisal Zubair Qureshi
ATR International
ATR Media Integration & Communications Research Laboratories
2-2, Hikaridai, Seika-cho, Soraku-gun
Kyoto 619-02, Japan

August 12, 1999

1. Introduction

Physically-based facial modeling has proven promising for facial animation. These models, consisting of mathematical abstractions for skin tissue, muscles and skeletal sub-surface, allow a wide range of expressions at high refresh rates. Geometery of the face is captured using stereo photogrammetery or optical surface scanners and used to setup the physically-based model. Facial expressions are dependant on the mechanical properties of the skin. These properties, in turn, depend upon elastin, collagen and ground susbtance [LS86]. The mechanics of the skin also depend upon the tissue thickness at various parts of the face. At forehead, the skin tissue is only few millimeters thick where as around cheeks, it is much more dense. Similarly, facial tissue has a variety of sub-cutaneous attachments. To correctly model the physical properties of the skin tissue, we need to take into consideration all the above things. In a realistic physically-based facial model we need to include both skin tissue and the underlying skeletal bone sub-surface.

Waters ([WT90]) proposed a way to estimate skeletal sub-surface geometery from the epidermis data. Although this simplification is good enough for animation purposes , it under-estimates the influence of the varying skin thickness. Also, lack of an accurate skull model makes jaw motion both difficult and unrealistic. Data from optical scanners and/or stereo photogrammetery contains information about the color and geometery of facial skin exo-surface and computer tomography (CT) data can be used to get skin thickness and underlying skeletal information. Using these data sets, we can compute skin thickness at various points on the face and build a more accurate skull-bone model.

In this report, we propose a way to combine information from the optical surface scanner (in our case cyberware range and reflectance data) and CT data to construct a physically-based facial model. This model takes into account the variations in skin thickness and creates more accurate facial expressions.

In order to be able to use information from the cyberware data and CT data effectively, we need to spatially align these data sets. This process is known as *registration.* To this end: skin surface constructed from cyberware data and skin surface extracted from CT data are registered together in an interactive environment. The

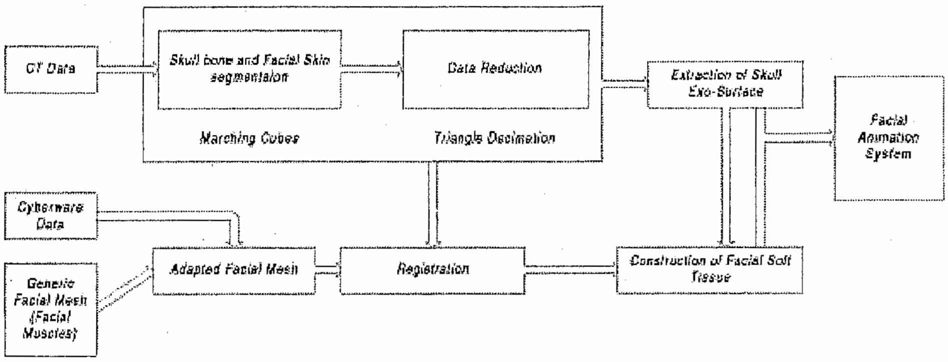
CT Data
Skull bone and Facial Skin segmentaion
Data Reduction
Marching Cubes
Triangle Decimation
Extraction of Skull Exo-Surface
Facial Animation System
Cyberware Data
Generic Facial Mesh (Facial Muscles)
Adapted Facial Mesh
Registration
Construction of Facial Soft Tissue

Figure 1.1: Block Diagram

user provides corresponding points from both surfaces, which the system uses to compute an affine transformation matrix. The later is in turn used to register the two surfaces. Once surfaces are registered, we have effectively registered optical surface scanner data with computer tomography data. The next step is to setup a skin model between the skin surface and the skull exo-surface. We currently use the skin model proposed by Waters, Terzopoulos and Lee ([YDK95], [Lee93]). This skin tissue model consists of prismatic volume elements. We shall discuss an algorithm for automatic generaion of prismatic units, given the skin surface and the skull model. The prismatic units can then be used to compute the skin thickness at various points, or can be imported in University of Toronto, physically-based facial animation system developed by Lee ([YDK95], [Lee93]) for animation. Figure 1.1 shows a flowchart of the system operation.

2. Related Work

A brief account of previous work on facial modelling follows.

Parke in 1972 ([Par72]) used key-framing to generate facial expressions, where facial expressions are generated by interpolating two or more captured facial expressions. This approach was unable to handle the immense variety of facial expressions. In 1974, Parke ([Par74], [Par82]) introduced parameteric models where each facial expression is specified by a set of parameter values and new facial expressions can be generated by interpolating these paramters. Parameters values for different facial expressions have to be specified carefully, otherwise it might result in unrealistic facial expressions. During 1980's models, whose paramters were anatomically inspired, were intorduced ([Pla80], [PB81], [Wat87], [Wai89], [MTPT88]). In these models, parameters were based on facial muscle structures. Anatomically based models can incorporate facial action coding schemes ([EF77]) to synthesize different expressions in a straightforward manner.

Geometric facial models tend to think of facial skin as a geometric surface with zero thickness. It is particularly hard to mimic subtle deformations of skin like wrinkling and bulging ([Kom88]) with this model of skin. Finite element based models of the skin have been developed and used in surgical applications ([DSCP94]). These models are computationally expensive and for animation purpose a simple deformable lattice serves the purpose ([YDK95], [Wat92], [WT92], [WT90]). In the area of cranio-facial surgery elaborate models for skin tissue and under-lying skull structure were developed. Kevin et al uses Cyberware data and CT data to setup a facial model although this model has no muscles. Waters ([Wat92]) used CT data to setup tissue model with accurate skin thickness information.

3. Extraction of Bone and Skin Surface from CT data

CT data consists of contiguous slices from an array of three dimensional Hounsfield values. From this data a number of geometric representations for different structures can be extracted. We use the *Marching Cubes* algorithm ([LC87]) to extract a polygon representaion of different iso-surfaces. The algorithm converts an array of density data into a number of polygons by a sequential tessellation of logical voxel constructed from 8 samples. Different thresholds results in different iso-surfaces. By carefully, selecting a threshold we can isolate the bone and the skin surfaces. Figure 3.1 shows the skull and figure 3.2 shows the skin constructed from two different CT data sets using this algorithm. The skull and skin has on the order of 500k polygons each. Because this amount of data is difficult to handle, we use *Triangle Decimation* algorithm ([SZL92]) to reduce the number of triangles to a more manageable 50k polygons. Currently, we use the generic skull shown in figure 3.3 .

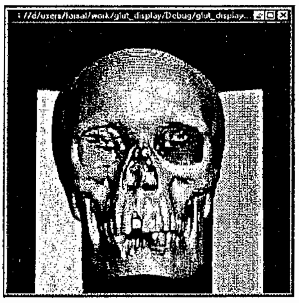

Figure 3.1: Skull extracted from CT data using Marching Cubes.

The extracted skin surface is used for registration of CT data with Cyberware data. We will discuss registration in the next section.

In addition to creating large data sets, *Marching Cubes* algorithm generates inner and outer surfaces of the skull bone. For the purpose of animation, we are only interested in the outer surface. Figure 3.3 shows that skull contains holes around eye orbits, nasal cavities, upper part of mandible and lower part of jaw. These features are anatomically correct but they pose unique difficulties in computing thickness between the skull exo-surface and the skin surface. It is desirable to have a continuous surface representation for skeletal foundation and skin because large cavities can create undesirable motion artifacts when simulated.

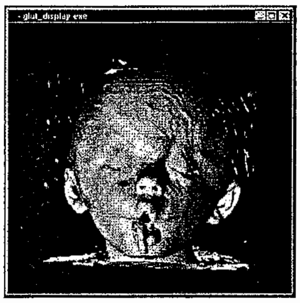

Figure 3.2: Facial skin extracted from CT data using Marching Cubes

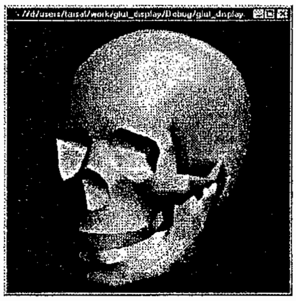

Figure 3.3: Generic skull model

3.1. Extracting a Continuous Skull Surface

Human face maps conveniently into the cylinderical coordinate system because human face, for the most part, is convex. We map the skull exo-surface in cylindrical coordinate system. A point (x, y, z) in cartesian coordinate system can be represented as (r, θ, y) in cylindrical coordinate system. Points sampled at the skull exo-surface can be mapped onto cylindrical coordinate system to generate a 2D array of r values called height map. θ varies from θ_{min} to θ_{max}[1] along the width of the height map; y varies from y_{min} to y_{max} along the height of the height map and each cell (θ, y) contains the corresponding r value. Figure 3.4 shows a height map created by sampling point on the outer surface of skull in figure 3.3.

Actual skull contains holes and these holes are mapped onto black regions in the image (figure 3.4). The user can paint and draw on this image to generate mask images. Different kinds of information can be encoded in these mask images and algorithm can use these masks to generate a continuous and smooth skull exo-surface.

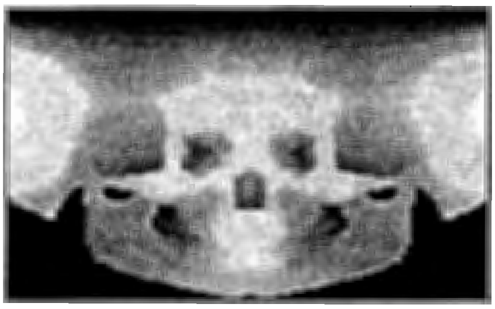

Figure 3.4: Cylinderical image map of skull

3.2. Radial Mapping

To generate these height maps: the skull is rotated such that y-axis intersects the skull at the top and passes right between the jaws. The skull is also translated along $x - z$ plane to minimize radial line of sight shadows. Depending upon various parameters entered by the user, a set of equally spaced planes parallel to $x - z$ plane, is generated. These planes are intersected with the skull to generate a set of contours. For each plane, which represents an row in the height map, $(\theta_{\text{max}} - \theta_{\text{min}})/N_r$ rays are fired from the center of the plane; the point where Y-axis intersects this plane. Number of rays per plane, N_r, is equal to the width of the height map. Intersection of these rays with the corresponding contours is computed and for each ray, the intersection point having maximum radius is stored. Thus, height map is composed of maximum radii for each intersected ray. Intersected points are stored in a N_r*N_p*3 array (N_r is the number of rows and N_p is the number of planes).

If there is no intersection; r is considered zero, therfore holes in the skull maps to black (0) regions in the gray level image, see figure 3.4. Once height map and intersected point array are filled, algorithm interpolates the missing values i.e., points

[1] θ varies from 0 to 2π. 0 correponds to negative $z - axis$.

where there was no intersection or points where intersected value is unacceptable like regions around eye orbits etc.

3.3. Algorithm for Closing Holes

3.3.1. Input:

Skull bone surface extracted from CT data, *dist*, *ip_array* and differents masks.

3.3.2. Output:

Skull bone exo-surface. (T and *ip_array* together define the skull bone exo-surface)

3.3.3. Symbols:

L_b	Set of (r, θ, y) points.
dist	Height map.
ip_array	Set of sampled points on the skull exo-surface. (It are also indexed as (θ, y)).
$N_{(0,\theta',y')}$	Neighbourhood of $(0, \theta, y)$. A set of points.
Ω	Pair (r, c) can be converted into a valid index in *ip_array* using $\Omega()$ operator. $\Omega(r, c) = c + r * N_r$.
T	A set of triangles. (a triangle is a triplet (a, b, c)).
p	A point (x, y, z).
p_{cyl}	A cylinderical poin (r, θ, y)

3.3.4. Steps:

1. $L_b = \Phi$, $T = \Phi$.
2. $\forall(\theta, y)$, $p = ip\_array(\theta, y)$ if $p = (0, 0, 0)$ then $L_b = L_b \cup \{(0, \theta, y)\}$.
3. If a mask is available then use mask information to decide bad nodes and insert bad nodes into set L_b.
4. Sort L_b in order of increasing badness. A bad vertex having least number of bad neighbouring vertices has least badness.
5. While L_b is not empty, do steps 4, 5 and 6:
    1. $\forall p_{cyl}, p_{cyl} \in L_b$, do steps
        1. Setup set $N_{(0,\theta',y')}$ where $\theta' = \theta \pm \Delta\theta$ and $y' = y \pm \Delta y$. Here, $\Delta\theta = (\theta_{\max} - \theta_{\min})/N_r$ and $\Delta y = (Y_{\max} - Y_{\min})/N_p$.
        2. $dist = \sum_{N_{(0,\theta',y')}} dist(\theta', y')$.
        3. If $dist \neq 0$; subtract $(0, \theta, y)$ from the set L_b. $dist(\theta, y) = dist$. $ip\_array(\theta, y) = CYL2CAR(dist, \theta, y)$.
6. $\forall r, r\epsilon[\theta_{\min}, \theta_{\max}]$ and $\forall c, c\epsilon[Y_{\min}, Y_{\max}]$, $T = T \cup \{(\Omega(r, c), \Omega(r+1, c), \Omega(r, c+1)), (\Omega(r+1, c), \Omega(r+1, c+1), \Omega(r, c+1))\}$.

Figure 3.5 shows skull exo-surface generated using this algorithm. The holes are closed and the skull bone has zero thickness. Figure 3.9 shows *use mask* used in generating skull exo-surface shown in 3.6.

3.4. Filling the Holes

The holes are closed, however, regions around eye orbits, nasal cavity and upper mandible still have cavities in them. These cavities are smoothed by carefully using mask images. The user paints the undesireable regions like eye orbits and nasal cavity etc. with black color. Thus, making these points invalid and algorithm calculates values for these points using neighbouring points. This results in smoothing the holes. Figure 3.7 shows the surface generated using *bad mask*. Multiple masks can also be used. Figure 3.8 shows the surface generated after using both *bad mask* and *use mask*. Figure 3.10 shows *bad mask* used to create exo-surface shown in figure 3.7. These masks were generated using distance map (figure 3.4).

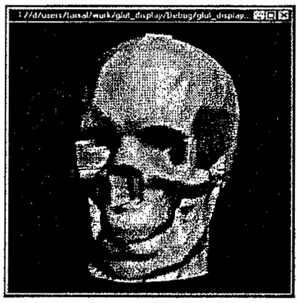

Figure 3.5: Skull exo-surface generated using above algorithm

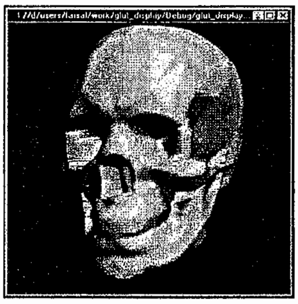

Figure 3.6: Skull exo-surface generates using use mask

4. Registration

For facial animation purpose, the two most important parts of human face are the facial tissue and the underlying facial bones. A complete physically-based facial an-

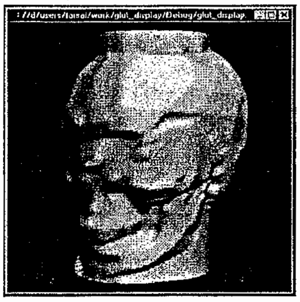

Figure 3.7: Skull exo-surface generated using bad mask

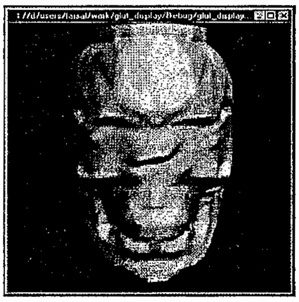

Figure 3.8: Skull exo-surface generated using both use mask and bad mask

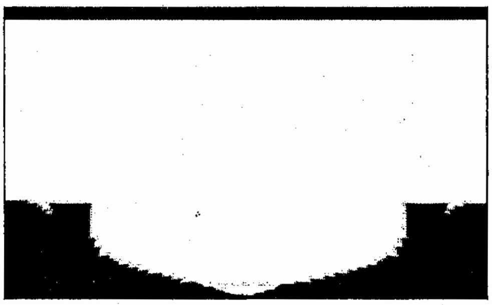

Figure 3.9: Use Mask

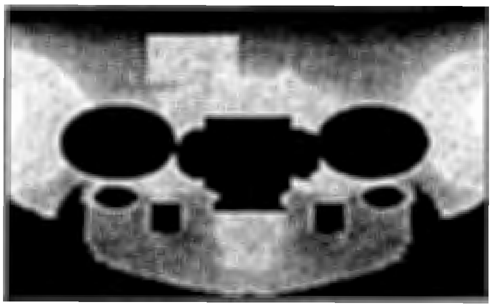

Figure 3.10: Bad Mask

imation model should not only include information of the soft tissue but also have knowledge of the underlying bone structure. Information about the soft tissue and the bone structure can not be captured by any single modality. In our case, color and texture of the skin can be captured by a Cyberware scanner where as skin tissue thickness and the underlying facial bone structures can be captured using CT data.

These data sets have to be registered together. Here, the assumption is that cyberware data and CT data comes from the same person. We adapt the generic facial mesh to the cyberware data using algorithm proposed by Lee in [Lee93] and extract the facial skin surface from the CT data as described in section 3.3. Once we have these two surfaces: extracted facial skin surface and the adapted facial mesh, we use global affine registration to spatially align them. We use point correspondences on both surfaces to compute 4x4 affine transformation matrix. User can interactively pick points on both surfaces and sets up the correspondence pairs. The system computes the transformation matrix and apply it to facial mesh surface to spatially align them. Thus, both surfaces are now registered.

In our case, it is enough to compute affine transformation matrix since both the surface are assumed to be from the same person and have similar local structure. Therefore, we only need scaling, translation and rotation to correctly align these two surfaces. Figure 4.1 and figure 4.2 show two similar generic facial meshes before and after registration respectively.

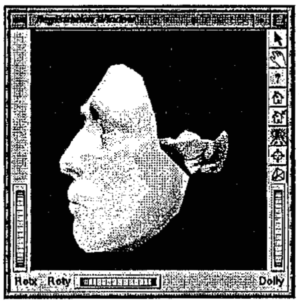

Figure 4.1: Two generic facial meshes before registration

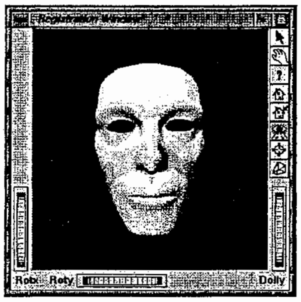

Figure 4.2: The generic facial meshes after registration

5. Facial Tissue Construction

We propose an algorithm for automatic construction of facial tissue model comprising of prismatic volume elements. This algorithm computes the thickness of the skin and uses this information to generate prismatic volume elements. It assumes that CT data and cyberware data are properly registered and that we skull exo-suface is already computed. This algorithm can construct facial tissue model even if the skull bone is not smooth but in this case user has to provide more information. Moreover, we need smooth skull exo-surface for animation purposes any way.

The algorithm computes skin thickness using generic facial mesh, which can be adapted to an individual face using algorithm described in [Lee93], and skull exo-surface constructed by algorithm described in section 3. Number of prismatic elements created are equal to the number of triangles in the generic facial mesh. Essentially, triangles comprising the facial mesh are extended inwards towards the skull exo-surface; generating prisms. The length of a prism depends upon the thickness of the skin in that region. Algorithm computes the thickness at various vertices of the facial mesh. If it is not possible to compute thickness at a vertex, it uses information from the neighbouring vertices to compute the skin thickness for that vertex. Once thickness value for all the vertices in the facial mesh are calculated, it sets up the prism elements. This thickness information can be used in Lee ([Lee93], [YDK95]) facial animation system to produce better facial animations.

5.1. Algorithm

5.1.1. Input:

Facial mesh, Skull Exo-surface, Vertex Info (e.g., a mask)

5.1.2. Output:

th_arr, S_p, inv_prisms.

5.1.3. Symbols:

th_arr	Thickness array
S_p	Set of prisms
inv_prisms	Set of inverted prisms only
S_v	Set of all vertices of facial mesh.
S_{Bv}	Set of vertices of facial mesh.
$\xi(v)$	Ray Operator. It generates a ray starting at vertex v along the negative normal of vertex v.
wall surface	Skull exo-surface.
seed surface	Facial mesh surface.
$INT\_PT(v)$	The point on *wall surface* where ray $\xi(v)$ intersects it.
$INT\_TRI(v)$	The triangle of *wall surface* with which ray $\xi(v)$ intersects.
S_t	Set of all triangles of facial mesh.
v	A vertex.
t	A triangle, a triangle is a set of three vertices i.e., $t = \{v, v', v''\}$.
p	A prism, a prism is a set of six vertices i.e., $t = \{v, v', v'', u, u', u''\}$.

5.1.4. Steps:

1. $S_{Bv} = \Phi$, $S_p = \Phi$.
2. Identify all the problem vertices of the facial mesh and put them in set S_{Bv}.
3. $\forall v,\ v \in S_v - S_{Bv}$, intersect $\xi(v)$ with *wall surface.* If there was a valid intersection, store intersection point, $INT\_PT(v)$, and triangle $INT\_TRI(v)$ of *wall surface* with which ray intersected. If there was no intersection $S_{Bv} = S_{Bv} \cup \{v\}$.
4. If $cos^{-1}\left(\frac{DIR(\xi(v)) \circ NORM(INT\_TRI(v))}{\|DIR(\xi(v))\|.\|NORM(INT\_TRI(v))\|}\right) \geq \frac{\pi}{2}$, then this intersection is unaccptable; $S_{Bv} = S_{Bv} \cup \{v\}$.
5. If user has specified an upper limit $T_{\max}$ of thickness then $\forall v,\ v \in S_v - S_{Bv}$, if $\|v - INT\_PT(v)\| > T_{\max}$ then $S_{Bv} = S_{Bv} \cup \{v\}$.
6. $\forall v,\ v \in S_{Bv}$ if user has specified a thickness T_v for v^2 then $INT\_PT(v) = -NORM(v) * T_v$. Subtract v from S_{Bv}.
7. While S_{Bv} is not empty, do steps 8,9 and 10:
    1. Sort S_{Bv} in increasing order of vertex badness. (A vertex with least number of bad numbers has least badness)
    2. Set up N_v.
    3. $INT_PT(v) = (-1) \sum_{N_v} NORM(v) * \|v' - INT_PT(v')\|$where $v' \in N_v$ and $v' \in S_v - S_{Bv}$.
    4. If $\|INT\_PT(v)\| > 0$ then subtract v from S_{Bv}.
8. $\forall t, t \in S_t$ if $t \subseteq S_v - S_{Bv}$ then $S_p = S_p \cup \{t, INT\_PT(t)\}^3$.

[2] User can specify thickness for nose and eyes vertices or for any other vertex for that matter.

[3] If $t = \{a, b, c\}$ then $INT\_PT(t) = \{INT\_PT(a), INT\_PT(a), INT\_PT(a)\}$.

9. $\forall v, v \in S_v$, $th\_arr(v) = v - INT\_PT(v)$.

10. $\forall p, p \in S_p$ if p is inverted $inv\_prism = inv\_prism \cup p$.

Figure 5.1 shows a facial tissue model generated by this algorithm. This model can be imported into any 3D modelling tool like maya and inverted prisms can be corrected. Number of inverted prisms increase with the increase in skin thickness. In the tests we have conducted, out of 810 prisms, no more than 10 to 15 are inverted. Currently, this system iteratively reduces the skin thickness, mainting the thickness ratio between different vertices, to correct the inverted prisms. Another advantage of this approach is, since the animation system becomes increasingly un-realistic and un-stable with increasing skin thickness so we can achieve better animation results by decreasing the thickness. Reducing the thickness does not adversly effect the animation results because thickness ratio between different parts of face like cheeks, nose and eyes is maintained.

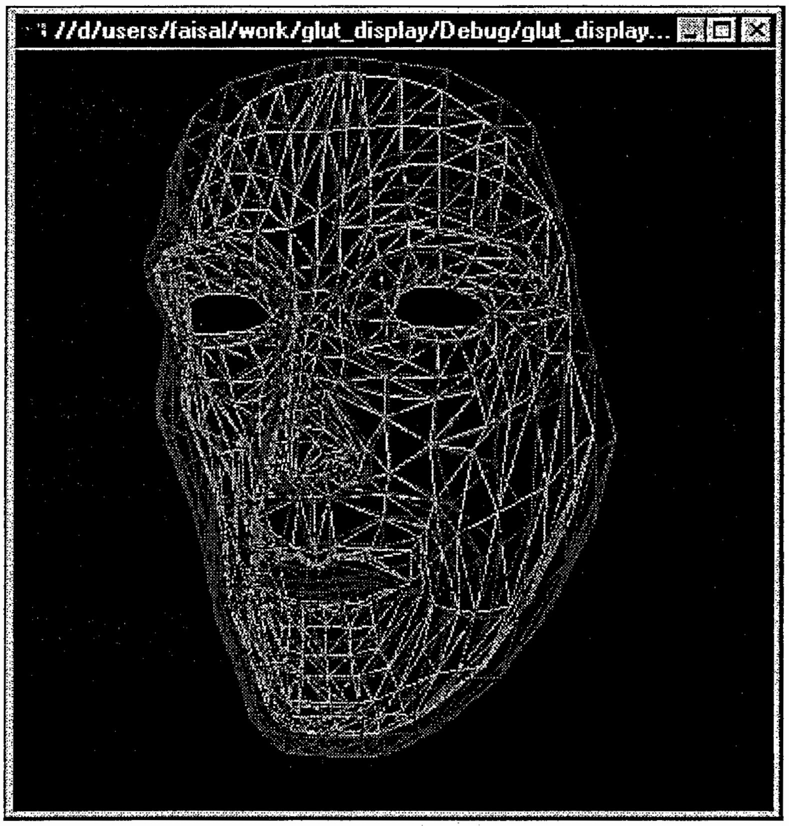

Figure 5.1: Skin tissue generated by above algorithm

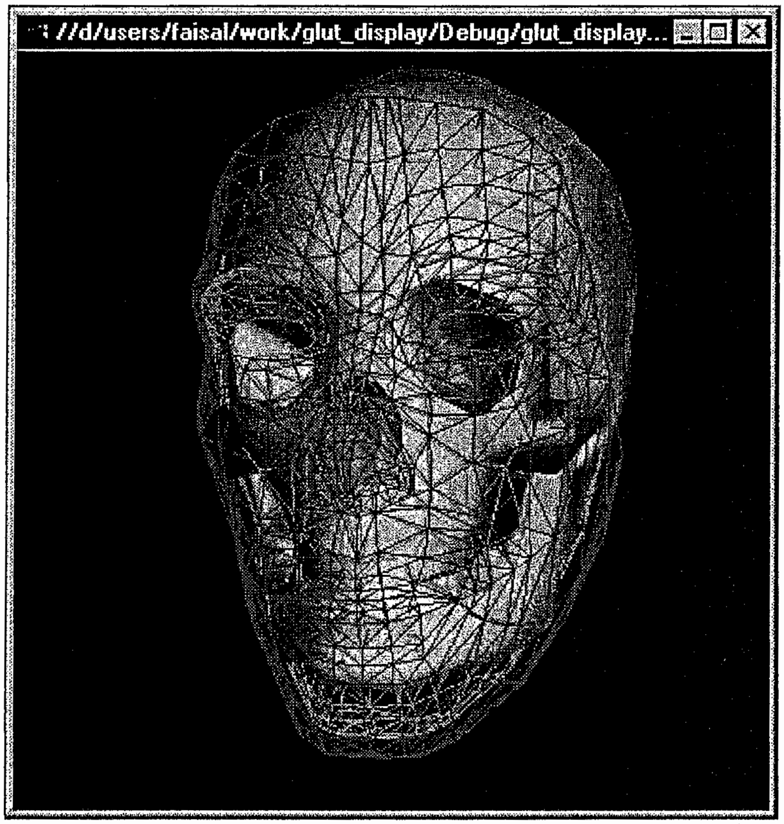

Figure 5.2: Skin tissue with the underlying skull bone

6. Result

Due to lack of time, we were unable to perform tests on real CT data. We used a generic skull model and a generic facial mesh to test our algorithms. Since these generic models were not from the same person and no skin information was associated with the generic skull model, we performed registration manually. Once the generic facial mesh and the generic skull model were registered, algorithm described in 3.3 computed a continuous and smooth skull exo-surface (Figure 3.8) using masks shown in figure 3.9 and 3.10. Algorithm, described in 5.1, then computed the skin tissue thickness and setup the skin tissue model (figure 5.1).

7. Conclusions and Future Work

In current physically-based models for facial animation importance of variation in skin thickness and underlying skull structure is underestimated. We have proposed a way to automatically setup physically-based facial animation models having correct skin thickness and a model for the underlying skull structure using Cyberware range, reflectance data and CT data of the face. This system holds promise for better facial animations. In future, we wants to address following issues:

- For now, skin thickness at a vertex is the average of skin thickness at neighbouring vertex. We treat all the neighbouring vertices equally. We can assign weights to neighbouring vertices, depending upon curvature of surface and their distance from the vertex where thickness is being calculated.
- Develop an algorithm to correct the inverted prisms without reducing the overall skin thickness.
- Speed up the facial skin tissue generation process so that it can be done in real time.
- Develop a gui front-end to the system.
- Develop a system so that generated facial tissue model can be automatically imported in Lee ([Lee93],[YDK95]) facial animation system.
- Conduct more tests on actual data to study the performance of the system.

References

[DSCP94] H. Delingette, G. Subsol, S. Cotin, and J. Pignon. A craniofacial surgery testbed. Technical report, INRIA, Rapport de recherge, N2199, Sophia-Antipolis, 1994.

[EF77] P. Ekman and W. V. Friesen. Manual for the facial action coding system. Technical report, Palo Alto: Consulting Psychologists Press, 1977.

[Kom88] K. Komatsu. Human skin model capale of natural shape variation. *Visual Computer*, 3:365–271, 1988.

[LC87] W. E. Lorensen and H. E. Cline. Marching cubes: High resolution 3d surface construction algorithm. In *Computer Graphics*, volume 21, pages 163–169, 1987.

[Lee93] Yuencheng Victor Lee. The construction and animation of functional facial models from cylinderical range reflectance data. Master's thesis, University of Toronto, Department of Computer Science, 1993.

[LS86] W. Larrabee and D. Sutton. A finite element model of skin deformation. In *Biomechanics of skin and soft tissue: A review*, volume 96, pages 399–405. 1986.

[MTPT88] N. Magnenat-Thalmann, E. Primeau, and D. Thalman. Abstract muscle action procedures for face animation. *Visual Computer*, 3:290–297, 1988.

[Par72] F. I. Parke. Computer generated animation of faces. Master's thesis, Department of Computer Science, University of Utah, 1972.

[Par74] F. I. Parke. *A parameteric model for human faces.* PhD thesis, Department of Computer Science, University of Utah, 1974.

[Par82] F. I. Parke. Parameterized models for facial animation. *IEEE, Computer Graphics Applications*, 2(9):61–68, 1982.

[PB81] S. M. Platt and N. I. Badler. Animating facial expressions. *Computer Graphics*, 15(3):245–252, 1981.

[Pla80] S. M. Platt. A system for computer simulation of the human face. Master's thesis, University of Pennsylvania, 1980.

[SZL92] W. J. Schroeder, J. A. Zarge, and W. E. Lorensen. Decimation of triangle meshes. In *ACM SIGGRAPH*, volume 26, pages 65–70, July 1992.

[Wai89] C. Waite. The facial action control editor, face: a parametric facial expression editor for computer generated animation. Master's thesis, Massachusetts Institute of Technology, 1989.

[Wat87] Keith Waters. A muscle model for animating three dimensional facial expression. *Computer Graphics*, 22(4):17–24, 1987.

[Wat92] Keith Waters. A physical model of facial tissue and muscle articulation derived from computer tomography data. In *Visualization in Biomedical Computing*, volume 1808, pages 574–583. SPIE, October 1992.

[WT90] Keith Waters and Demetri Terzopoulos. A physical model of facial tissue and muscle articulation. In *First conference on visualization in biomedical computing*, pages 22–25, May 1990.

[WT92] Keith Waters and Demetri Terzopoulos. A computer synthesis of expressive faces. In *Phil. Trans. R. Soc. Lond.*, number 335, pages 87–93. 1992.

[YDK95] Yuencheng Lee, Demetri Terzopoulos, and Keith Waters. Realistic modeling for facial animation. *SIGGRAPH*, pages 55–62, 1995.