Construction and Simplification of Bone Density Models

Jianhua Yao, Russell Taylor

Computer Science Department, The Johns Hopkins University, Baltimore, MD

Abstract

This paper presents a hierarchical tetrahedral mesh model to represent the bone density atlas. We propose and implement an efficient and automatic method to construct hierarchical tetrahedral meshes from CT data sets of bony anatomy. The tetrahedral mesh is built based on contour tiling between CT slices. The mesh is then smoothed using an enhanced Laplacian algorithm. And we approximate bone density variations by means of continuous density functions written as smooth Bernstein polynomial spline expressed in terms of barycentric coordinates associated with each tetrahedron. We further perform the tetrahedral mesh simplification by collapsing the tetrahedra and build hierarchical structure with multiple resolutions. Both the shape and density error bound are preserved during the simplification. Furthermore a deformable prior model is computed from a collection of training models. Point Distribution Model is used to compute the variability of the prior model. Both the shape information and the density statistics are parameterized in the prior model. Our model demonstrates good accuracy, high storage efficiency and processing efficiency. We also compute the Digitally Reconstructed Radiographs from our model and use them to evaluate the accuracy and efficiency of our model. Our method has been tested on femur and pelvis data sets. This research is part of our effort of building density atlases for bony anatomies and applying them in deformable density based registrations.

Keyword: Hierarchical Tetrahedral Mesh, Density Function, Mesh Simplification, Prior Model

1. Introduction and Background

Medical images are usually complex, noisy and possibly incomplete, which makes the interpretation and analysis of medical image very difficult without prior knowledge of anatomy. Electronic atlas of anatomy provides potential solutions to these difficulties. The prior knowledge stored in the atlas can be used to resolve the ambiguity caused by anatomical complexity and variability, to provide tolerance to noisy or incomplete medical data, and to offer an automatic labeling of the anatomical structure [5, 6]. Many groups are developing electronic atlases as a reference database for consulting and teaching, for deformable registration, for assisted segmentation of medical images and for use in surgical planning. Researchers at INRIA have built atlases based on surface models and crest lines [1]. Cutting *et al* [2] have built similar atlases of the skull. These atlases are surface models with landmarks and crest lines and don't contain any volumetric density information. Chen *et al* [3] have built an average brain atlas based on statistical data of the voxel intensity values. Their atlases only contain intensity information and don't describe the geometrical features of the anatomy. Pizer *et al* have proposed a medial model representation called M-rep to represent the shapes of 3D medical objects [4]. One advantage of this approach is that it is easy to deform medial atoms to accommodate shape variations. One drawback is that the current representational scheme is primarily useful for exterior shapes, and more work will be needed to extend this work to volumetric properties. Cootes *et al* [5, 6] proposed an Active Appearance Model that incorporates both the shape variability and density variability using Principal Component Analysis method. They implemented their models on 2D MRI brain slices and 2D human face images. Their models have also been extended to 3D surface model by Fleute *et al* [7].

One goal of our current research is to construct a deformable *volumetric* density atlas for bony anatomy and apply this atlas to various applications. Our intent is to apply the atlas to intensity and shape-based deformable 2D/3D and 3D/3D registration methods and to exploit it for patient-specific procedure planning and intraoperative guidance. The atlas will provide a basis for representing "generic" information about surgical plans and procedures. The atlas will also offer infrastructure for study of anatomical variability. It may also provide an initial coordinate system for correlating surgical actions with results, as well as an aid in postoperative assessment. The atlas should include: 1) 3D parameterization of anatomical landmarks and surface shape; 2) model representations of "normal" densities and volumetric properties; and 3) statistical characterization of variability of parameters.

We adopt a hierarchical tetrahedral mesh structure to represent volumetric properties for several reasons. The tetrahedral meshes provide the greatest possible degree of flexibility. Furthermore, other meshes can be easily converted into tetrahedral mesh. Data structure, data traversal, and data rendering for tetrahedral mesh are more involved. And tetrahedral meshes have

Medical Imaging 2001: Image Processing, Milan Sonka, Kenneth M. Hanson, Editors,
Proceedings of SPIE Vol. 4322 (2001) © 2001 SPIE · 1605-7422/01/$15.00

the ability to better adapt to local structures. It is also convenient to assign properties to the vertices and tetrahedra. Computational steps such as interpolation, integration, and differentiation are fast and often can be done in closed form. Finite element analysis is often conveniently performed on tetrahedral meshes. The hierarchical structure of multiple resolution meshes can make the analysis and visualization more efficient. We build our model on bony anatomy such as pelvis and femur using their CT data sets, but this model can be easily extended to other anatomy.

The remainder of the paper is organized as follows. Section 2 elaborates the method to construct the tetrahedral mesh from bone contours and introduced the technique for tetrahedral mesh smoothing. Section 3 presents the method to assign a density function to the tetrahedron and evaluate its accuracy. Section 4 describes our method to perform tetrahedral mesh simplification based on edge collapsing and build a hierarchical structure. Section 5 shows the results using the density model to efficiently generate Digitally Reconstructed Radiographs (DRRs). The DRRs generated from the models are also used to evaluate accuracy and efficiency of our model. Section 6 introduces our method to build a prior model from a collection of training models. Finally section 7 discusses current status and future works.

2. Tetrahedral Mesh Reconstruction from Contours

There are several techniques to construct tetrahedral meshes from 3D volume data sets. The easiest way first subdivides the 3D space into cubicle voxels, then decomposes each voxel into five tetrahedra [8]. The drawback of this method is that the mesh generated is too dense and doesn't have any anatomical meaning. 3D Delaunay triangulation is another method. But this method is time consuming and requires some post-processing since the basic algorithm produces a mesh of the convex hull of the anatomical object. Boissonnat *et al* [9] proposed a method to construct tetrahedral mesh from contours on cross section. They first computed 2D Delaunay triangulation on each slice, then tiled tetrahedral mesh between sections and removed those external tetrahedra. But they didn't consider the intersection and continuity constraints between tetrahedra, and their meshes also don't have anatomical meaning.

Our tetrahedral mesh construction algorithm is derived from those surface mesh reconstruction algorithms [10]. The tetrahedral mesh is constructed slice by slice and layer by layer based on bone contours extracted in a separate algorithm. The running time is $O(n)$, where n is the total number of vertices in the model.

We chose a simple data structure to represent the tetrahedral mesh, consisting of a list of vertex coordinates and a list of tetrahedra. Each tetrahedron contains links that reference its four vertices and its four face neighbors. The face neighbors of a tetrahedron are those tetrahedra share a common face with this tetrahedron. The face neighbors are stored corresponding to the vertex it doesn't contain, i.e. face neighbor *1* doesn't contain vertex *1*, etc. It is easy to maintain and update this data structure. And the connectivity information can be easily retrieved from this data structure and takes $O(1)$ running time. Other information such as the density function is also stored in the data structure.

Figure 1 shows the flow chart to reconstruct the tetrahedral mesh from the bone contours extracted in each CT slice. First a separate algorithm is used to semi-automatically extract the bone contours and their medial axes. Then modeling steps including contour tiling, branching, metric function and constraints are performed to construct a tetrahedral mesh from the contours. Finally a smoothing step is taken to remove the noises in the CT data and artifacts during segmentation. The details are described in the following sections.

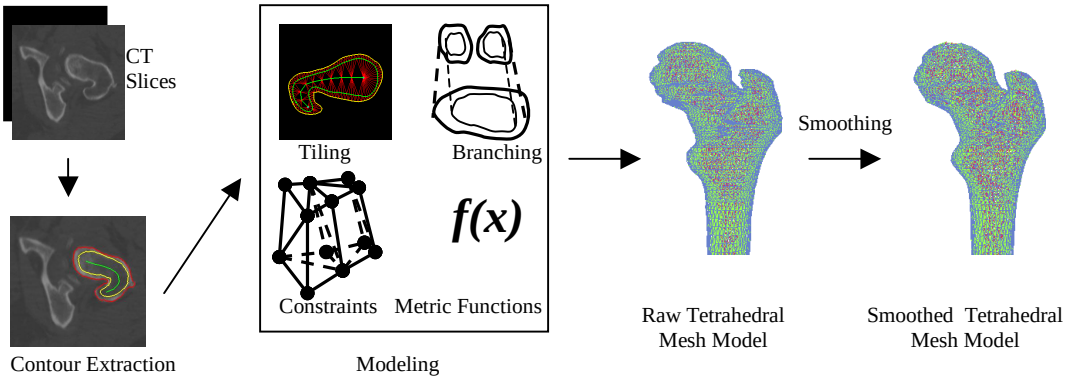


Figure 1. Tetrahedral Mesh Reconstruction from Contours

2.1 **Bone Contour Extraction:** Bones contain two basic types of osseous tissue: compact bone and spongy bone [11]. Compact bone is dense and looks smooth and homogeneous. Spongy bone is composed of little beams and has a good deal of open space. Figure 2 illustrates the bone structure. In our method, we apply a deformable "snake" contour model to extract exterior bone contours and boundaries between compact bone, spongy bone and Medullary Cavity. Provided the initial shape and position of the contour, the algorithm can automatically converge to the bone contour. The users only need to provide the initial contour for the very first slice. For any following slice, the contours generated on previous slice can be used as initial values. Contours are then fitted into smooth B-spline curves and re-sampled at desired resolution for the follow up tiling. During sampling, the contour segments with larger curvature are sampled at higher resolution, while those with smaller curvature at lower resolution. We also compute the medial axis of the contour and treat it as the innermost layer. This process is not fully automatic. The segmentation algorithm may generate bad results when two pieces of bones contact or too close to each other (e.g. the acetabular of the pelvis), hence manually adjustment need to be imposed.

2.2 **Tiling:** Tiling is the essential step in constructing the tetrahedral mesh. The idea is to subdivide the space between adjacent slices into tetrahedra, then connect the tetrahedra into a mesh. In previous paper, we proposed a 3D tiling strategy based on the tiling of four ordered contours on two adjacent slices. There are 32 distinct tiling patterns for each single step [12]. It is complex and time consuming to choose the optimal tiling pattern out of all those patterns. Furthermore, because of the tiling constraints (Section 2.4), the 3D tiling strategy discussed in [12] may not be able to find a feasible tiling solution in certain situations. In such case, we need to back up the process and start over again. Due to these disadvantages, we propose a new tiling strategy. In this scheme, we separate the complex 3D tiling procedure into two simpler procedures. In the first procedure we compute the 2D tiling between contours on each slice and the tiling between medial axes. In second step the 2D tiling of contours and the tiling of the medial axis can be 'knitted' to form a tetrahedral mesh. Hence each tiling step has at most four tiling patterns (by choosing the knit direction). And the constraint problem is also much simplified. Figure 3 illustrates this tiling scheme.

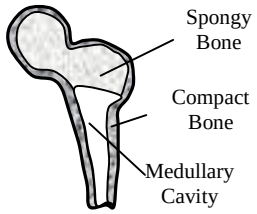


Figure 2. Bone Structure

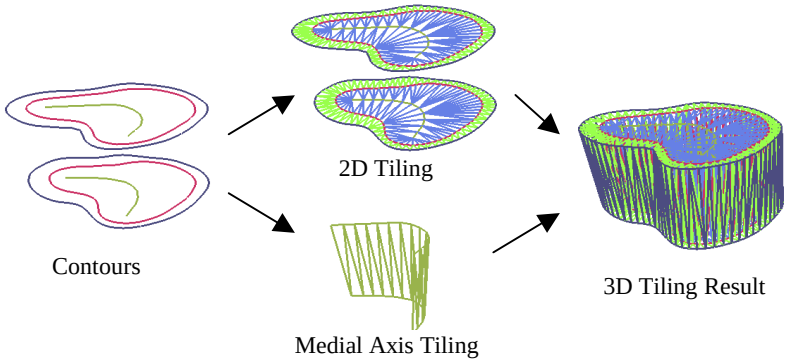


Figure 3. Tiling Strategy

2.3 Metric Functions: In order to get the optimal tiling pattern, a metric function must be evaluated for each candidate pattern. Several metric functions have been tested, including Maximize-Volume, Minimize-Area, Minimize-Density-Deviation, and Minimize-Span-Length. The metric function currently used in our algorithm is a combination of minimizing density deviation and span length.

2.4 Constraints: Some constraints must be imposed in order to form a manifold tetrahedral mesh. 1) Non-intersection between tetrahedra. This constraint is obvious for a valid spatial subdivision of a volume. When we select a tiling pattern, those newly generated tetrahedra shouldn't intersect with the old tetrahedra already in the mesh. 2) Continuity between slices. A triangle face on one slice should be shared by two tetrahedra in adjacent sections. This constraint is automatically fulfilled in our 2D tiling strategy (Section 2.2). 3) Continuity between layers. This constraint is similar to the continuity constraint between slices.

2.5 Correspondence Problems: The corresponding problem arises whenever there are multiple contours on one slice. We solved the correspondence problem by simply examining the overlap and distance between contours on adjacent slices. Manually interference may be provided in complex correspondence cases.

2.6 **Branching Problems:** There are two kinds of branching problems: branching between layers and branching between contours. The layer branching occurs when the numbers of layers of corresponding contours between adjacent slices are different. The layer branching can be converted to the tiling of three contours, which is a special case of the tiling of four contours. The contour branching occurs when the numbers of contours between adjacent slices are different. One method is to construct a composite contour that connects the adjacent contours at the closest points. This composite contour is then tiled with the single contour from the adjacent slice. Another method is to split the contour into several parts and tile each part with its counterpart on the other slice. We adopted the composite contour method in our model building strategy.

2.7 Tetrahedral Mesh Smoothing

Due to noises in the CT data and artifacts during segmentation, the tetrahedral mesh generated from our algorithm may not look smooth and may include some severely distorted elements. These distorted elements may cause numerical difficulties during later processing and visualization. One approach to remove the noises is to adjust the vertices' location without changing the mesh topology. This technique is called mesh smoothing. The most commonly used smoothing technique is Laplacian smoothing [13]. Laplacian smoothing is well known in image processing and finite element meshing. The basic idea is to replace the location of a vertex in a mesh with the average of the locations of vertices in the neighborhood. The process of averaging can be applied iteratively until the result is satisfied. The equation of Laplacian smoothing can be written as $v_i' = \frac{1}{|N_i|}\sum_{j \in N_i} v_j$, where N_i is the set of vertices in the neighborhood of v_i, v_j is location in the old mesh, v_i' is location in the smoothed mesh. Figure 4 is an illustration of Laplacian smoothing in 2D.

The Laplacian smoothing method is inexpensive. But it has several drawbacks. First it usually causes the shrinkage of the model (refer to Figure 5). Furthermore the operation is heuristic and does not guarantee improvements in the element quality. In fact, it is possible to produce an invalid mesh containing elements that are inverted or have negative volume (see Figure 6). We propose an enhanced Laplacian Smoothing algorithm to address those problems. To reduce shrinkage, instead of simply using the average location of neighborhood vertices, we project the average location to the mesh boundary and use it as the new location. In this way, we push the vertices of the smoothed mesh back toward its original boundary to keep the volume of the model. Figure 5 illustrates how this work in a 1D case. Now the new location can be written as $v_i' = proj(\frac{1}{|N_i|}\sum_{j \in N_i} v_j)$.

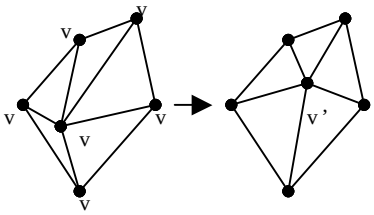


Figure 4. 2D Laplacian Smoothing

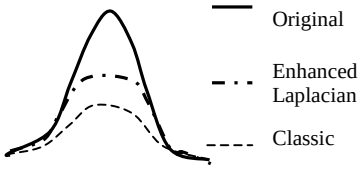


Figure 5. Shrinkage in Laplacian Smoothing

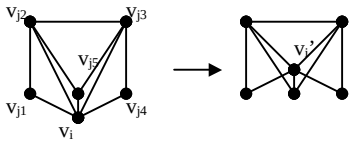
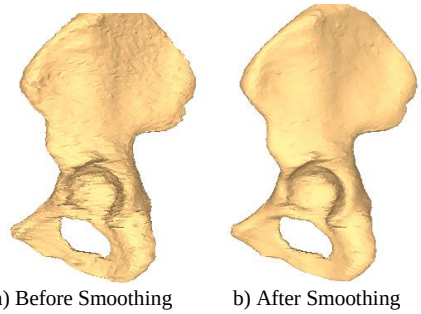


Figure 6. Invalid mesh caused by smoothing



a) Before Smoothing b) After Smoothing

Figure 7. Tetrahedral Mesh Smoothing

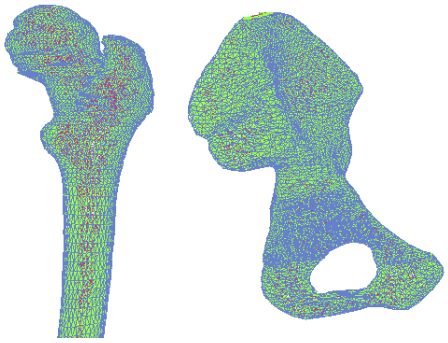


Figure 8. Femur and Pelvis Model

Because Laplacian smoothing is heuristic, it may produce invalid mesh containing intercrossed tetrahedra (see Figure 6 for an example in 2D case). The intercrossing can be detected by the flipping of tetrahedra (refer section 4.2 for details). One variant of Laplacian smoothing called "smart Laplacian" [13] accepts the new location only when it forms a valid mesh,

otherwise use its original location. This scheme works but too brute-forced so that can't smooth the mesh in some area. We proposed a variant method using iterative assignment to address the problem. In this scheme, if intercrossing occurs, instead of totally discarding the newly computed location, we compute a weighed location using the original location and newly computed location and treat it as a new location and test the intercrossing. The scheme can be written in the following equation: $v_i'^{(0)} = proj(\frac{1}{|N_i|}\sum_{j\in N_i} v_j)$, $v_i'^{(k)} = \alpha \cdot v_i + (1-\alpha)v_i'^{(k-1)}$,$0 \le \alpha \le 1$, the process can be applied iteratively until we find a valid location $v_i'^{(k)}$.

We are also investigating the technique of face and edge swapping to improve the connectivity and topology of the mesh, as well as the combination of swapping and smoothing. Figure 7 shows the results of a tetrahedral mesh of pelvis before smoothing and after smoothing.

2.8 Tetrahedral mesh models

In manifold tetrahedral mesh, the neighborhood of any face can be continuously deformed to a cone or half cone. In the cases of half cone, that face can be defined as a boundary face. The tetrahedral mesh generated by our method is guaranteed to be manifold and well behaved because of the constraints we impose during the tiling.

Figure 8 shows wire frame renderings of two tetrahedral mesh models produced by our method. Figure 8 (left) is a femur model. Figure 8 (right) is a half pelvis model. Table 1 lists some facts about these two models.

Model	Num of Vertices	Num of Tetrahedra	Num of Slices	Total Num of Voxels inside	Avg Num of voxels Per Tetra	Volume (mm^3)	Avg Vol. Per Tetra (mm^3)
Femur	6163	31,537	83	1,802,978	57.1	312,107	9.9
Pelvis	8219	32,741	110	1,941,998	59.3	347,070	10.6

Table 1. Facts about two tetrahedral mesh models

3. Density Function

We assign an analytical density function to every tetrahedron instead of storing the density value of every voxel in the model. The advantage of such a representation is that it is in explicit form and is a continuous function in 3D space. So it is convenient to integrate, to differentiate, to interpolate, and to deform.

Currently we define the density function as an n-degree Bernstein polynomial in barycentric coordinates. $D(\mu) = \sum_{i+j+k+l=n}^{n} C_{i,j,k,l} B_{i,j,k,l}^{n}(\mu)$, here $C_{i,j,k,l}$ is the polynomial coefficient, and $B_{i,j,k,l}^{n}(\mu) = \frac{n!}{i!\,j!k!l!}\mu_x^i\mu_y^j\mu_z^k\mu_w^l$ is a barycentric Bernstein basis. The definition of barycentric coordinate in a tetrahedron is as following. For any point *K* inside a tetrahedron *ABCD*, there exists four masses w_A, w_B, w_C, and w_D such that, if placed at the corresponding vertices of the tetrahedron, their center of gravity (barycenter) will coincide with the point *K*. (w_A, w_B, w_C, w_D) is called the barycentric coordinate of *K*. Furthermore barycentric coordinates are in a form of homogeneous coordinates where $w_A+w_B+w_C+w_D=1$, and the barycentric coordinate is defined uniquely for every point. By defining the density function on a symmetric and normalized coordinate system as barycentric system, it makes the density function independent of the shape, which is a great advantage for volume model deformation and prior models computing.

For each tetrahedron, we first get a sample of the voxels inside the tetrahedron, and obtain the voxel density via the CT data set. The number of sample voxels is more than twice of the number of polynomial coefficient and the sample voxels are evenly distributed within the tetrahedron. We then solve a least square equation to fit a Bernstein polynomial function of the density in the barycentric coordinates of those sample voxels. The least square problem can be written as $\min \sum_{\rho_i \in \Omega} \left(\left(\sum_{i+j+k+l=n}^{n} C_{i,j,k,l} B_{i,j,k,l}^{n}(\mu_{\rho_i}) \right) - T(\mu_{\rho_i}) \right)^2$, where Ω is the set of sample voxels, $\mu_{\rho i}$ is the barycentric coordinate of one sample voxel ρi, $T(\mu_{\rho i})$ is the density value from the CT data set. The problem can also be written in a matrix form as

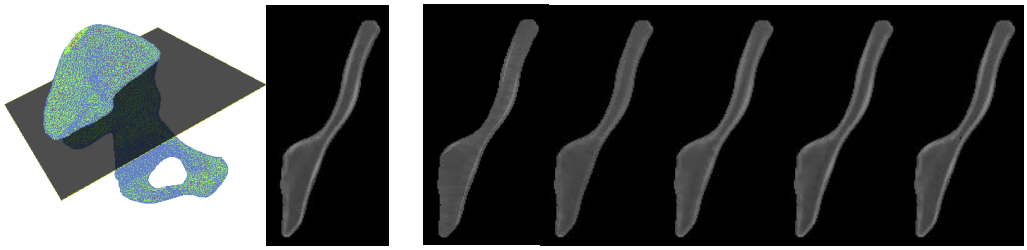
$$\begin{bmatrix} B_1(\mu_{\rho 1}) & B_2(\mu_{\rho 1}) & \dots & B_m(\mu_{\rho 1}) \\ B_1(\mu_{\rho 2}) & B_2(\mu_{\rho 2}) & \dots & B_m(\mu_{\rho 2}) \\ \vdots & \vdots & \vdots & \vdots \\ B_1(\mu_{\rho s}) & B_2(\mu_{\rho s}) & \dots & B_m(\mu_{\rho s}) \end{bmatrix} \cdot \begin{bmatrix} C_1 \\ C_2 \\ \vdots \\ C_m \end{bmatrix} = \begin{bmatrix} T(\mu_{\rho 1}) \\ T(\mu_{\rho 2}) \\ \vdots \\ T(\mu_{\rho s}) \end{bmatrix}$$

. Here s is the number of sample voxels and m is the number of coefficient in the density function, and $s>>m$. This is an over-constrained linear equation and can be solved using least square error method.

Once having the density function, we can compute the density at any point inside the tetrahedron by their barycentric coordinates. The higher the degree of the density function is, the more accurate the density function is. In order to measure the accuracy of the density function, we use CT data set as ground true and cut an arbitrary plane through the model and the CT volume, then compare the profile of the cutting plane (Figure 9). Table 2 is a comparison of polynomial degree via average density error of the density function of the images in Figure 9. Each tetrahedron contains hundreds of voxels. Using a density function with few coefficients also shows high storage efficiency (Table 3).

Degree	0	1	2	3	4	5	6	7	8
Coeff Number	1	4	10	20	35	56	84	120	165
Avg. Density Err (%)	3.291	1.583	0.766	0.442	0.298	0.216	0.167	0.149	0.128

Table 2. Degree of Density Function via Accuracy of Density Function of half pelvis model



Cutting Plane Ground Truth n=1 n=2 n=3 n=4

Figure 9. Degree of Density Function vs Accuracy of Density Function

4. Tetrahedral Mesh Simplification and Hierarchical Mesh Structure

Tetrahedral meshes with multiple resolutions are very desirable for some applications. We can build tetrahedral meshes with different level of details by sampling the contours at different resolution. But there would be no hierarchical connections between different scales of meshes. Our current modeling strategy is as following: first construct a fine model from the pre-segmented contours of bones. Then use a bottom-up approach to produce a hierarchy of tetrahedral mesh, each of which is within a specific error bound of its upper level. Edge collapsing was chosen to simplify the mesh. We put all the edges in a priority queue according to their cost functions. And each time we select the edge with smallest cost function for collapsing. The edge collapsing is performed by contracting one of its vertices to the other, while merging all the tetrahedra incident to the diminishing vertex (Figure 10 shows a 2D case of edge collapsing). The h^{th} level of mesh is obtained by simplifying the $h\text{-}1^{th}$ level of mesh. This allows one to proceed incrementally, taking advantage of the work done in previous simplifications. It also builds a hierarchical structure in which the vertices at the h^{th} level are a subset of the vertices at the $h\text{-}1^{th}$ level.

Our algorithm draws heavily on the work of [8]. Our main contributions are methods for preserving anatomical feature and density properties during the simplification. We also guarantee that no self-intersections occur in every level of the hierarchy.

4.1 Anatomical Feature Preserving: our model is built to adapt to the anatomical features (cortical bone walls). And the mesh is tiled layer by layer (Section 2.2). To preserve the anatomical features, edge collapsing across layers is prohibited.

4.2 Non-self-Intersection: flipping of a tetrahedron during edge collapsing can cause intersection between tetrahedra. In figure 11, tetrahedron $v_1v_2v_3v_4$ flips and intersects with $v_1v_2v_3v_5$. The normal flipping heuristic can be generalized in order to prevent the flipping of tetrahedra. The method is to analyze the signed volume of involved tetrahedra. If the volume of one of the neighboring tetrahedra changes sign, tetrahedron flipping occurs. Such edge collapsing should be forbidden.

4.3 Density Preserving: the ability to characterize the density property of a tetrahedron after the simplification is very important. As shown in figure 11, after edge collapsing and re-tiling, tetrahedron t_i in h-1^{th} level is replaced by t_j' in h^{th} level. The density function of a tetrahedron in h^{th} level is computed from those tetrahedra in h-1^{th} level it intersects with. For instance in figure 11, t_1' in h^{th} level intersects with t_1, t_2, t_3 in h-1^{th} level. Our way to compute new density function for t_j' is as following: first get a sample of voxels in t_j', then for each sample voxel locate its containing tetrahedron t_i in h-1^{th} level and compute its density, then fit a new Bernstein polynomial for t_j' using the sample (Section 3). In order to preserve the density function of t_j', we define the error metric of density function as $E(t_j') = \oint_{u \in t_j'} \left[D_{t_j'}(u) - D_{t_i}(v) \right]^2 du$, here v is the corresponding voxel of u in h-1^{th} level, and t_i is the containing tetrahedron of v. $E(t_j')$ is used as the cost function during edge collapsing and should be smaller than a pre-defined error bound for density preserving. If the error is too big, we can either discard the collapsing or elevate the degree of the density function to improve the accuracy.

Level	Num of Verts	Num of Tetras	Avg Density Diff (%)	Std Dev of Density Diff (%)	Avg Vol. Per Tetra (mm^3)	Avg Num of voxels Per Tetra	Storage in CT image (bytes)	Storage in Tetra model (bytes)
1	8219	32741	3.1%	2.2%	10.6	59.3	3,883,996	1,932,124
2	1272	6138	6.4%	3.8%	55.2	316.4	3,883,996	358,992
3	512	2438	8.8%	6.5%	137.9	796.6	3,883,996	142,672

Table 3. Facts about hierarchical half pelvis model in figure 12 (right)

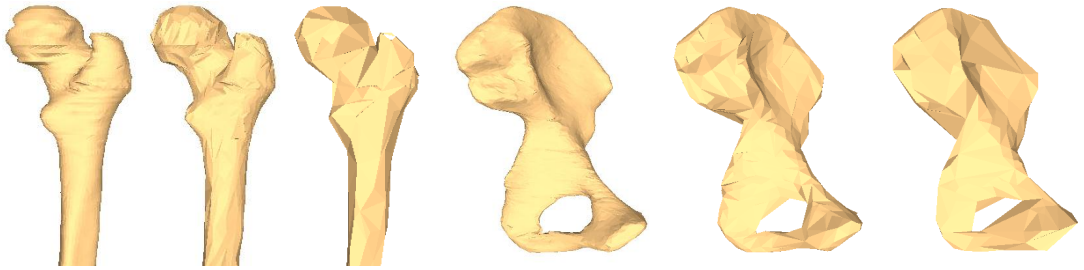


Figure 12. Multiple resolution femur model and pelvis model

Figure12 (left) shows a femur model with multiple resolutions, and figure 12 (right) shows a hierarchical pelvis model. Table 3 lists some facts about the hierarchical half pelvis model and the accuracy and storage efficiency of its density function.

5. Computing DRRs from Hierarchical Tetrahedral Mesh Density Model

We can employ a ray-casting algorithm to efficiently generate Digitally Reconstructed Radiographs (DRRs) through our tetrahedral mesh density model. When a ray passes through a 3D data set, it may go through hundreds of voxels but only a few tetrahedra. Because the density function is a Bernstein polynomial, its integral along a line can be computed in close form. Furthermore the neighborhood information is stored in the mesh, so next tetrahedron hit by the ray can be quickly retrieved from current hitting tetrahedron. And it is also fast to find the entry point of the casting ray using the neighborhood information. Meanwhile generating DRRs from CT data requires computing voxel convolution, which is a time consuming process [14]. So generating DRRs from our tetrahedral density model is very efficient, which is an important technique in 2D-3D intensity-based registration.

It is obvious that if we generate the DRRs from a coarser resolution mesh, the process can be speed up but DRR quality may be sacrificed. There is always a tradeoff between speed and quality. Using hierarchical mesh, we can choose the resolution of mesh according to different applications and different situations. Figure 13a is a DRR generated from a CT data using ray casting algorithm. Figure 13b, 13c, 13d are DRRs generated from our density model with different resolution (the pelvis models on Figure 12 right). The image size is 512*512. Table 4 shows the comparison of those DRRs in Figure 13. We used DRR generated from CT (figure 13.a) as the ground truth and compared the pixel intensities of those DRRs in figure 8. The running times are also compared on a Pentium III 400 PC with 256Mb memory. Because of the difficulty of visualizing 3D volumetric model, DRRs can also be used to evaluate the density and shape properties of our density model.

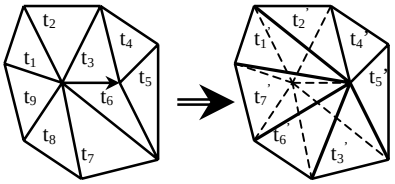


Figure 10 Edge Collapsing in 2D

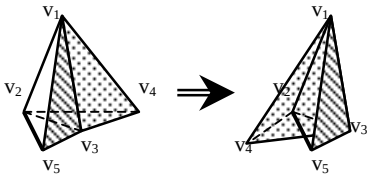


Figure 11 Tetrahedron Flipping

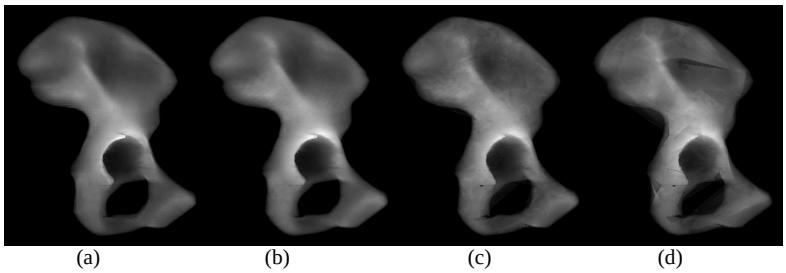


Figure 13. DRRs from CT data and multiple resolution density models in figure 12 (right)

(a) from CT (b) from density model with 32741 tetras (c) from density model with 6138 tetras (d) from density model with 2438 tetras

	Num of Tetra	Running time	Avg. elems Passed through	Avg Intensity Diff (%)	Std Dev of Intensity Diff (%)
CT Data set	N/A	29.4 s	132.6 voxels	N/A	N/A
Density Model	31537	9.2 s	43.1 tetras	3.2%	2.4%
Density Model	12719	5.6 s	21.8 tetras	7.6%	5.7%
Density Model	2448	1.9 s	7.3 tetras	14.4%	10.3%

Table 4. Comparison of DRRs generated from CT data set and multiple resolution density models

6. Prior models

The variability inherent in human anatomy makes medical image interpretation a difficult task. Prior knowledge drawn from a sufficiently large set of training models can be used to characterize the variability of anatomy. Bone density model is ideal for statistics analysis, because 1) the deformation of bone shape is relatively small; 2) the density variability of bone tissue is also small; 3) bone density distribution is relatively homogeneous. Cootes *et al* [5, 6] developed Active Appearance Model of brain ventricle using the Point Distribution Model (PDM) to characterize variability of brain. We employ and specialize the PDM model in our tetrahedral mesh model to describe the prior information of bony anatomy. Given a set of training model M_i, we compute the prior model in two steps: 1) model aligning; 2) statistics analysis and prior parameter computing.

6.1 Model Aligning

In our application, we first choose one model (says Model M_0) as the standard model, then register and deform the standard model to match other models in the training set. We use surface based method to align two models. We extract the exterior surface of the tetrahedral mesh and correspond two exterior surfaces in two steps. First an Iterative Closest Point (ICP) method is used to compute the rigid affine transformation. Then a deformable surface registration algorithm called Adaptive Focus Deformation Strategy proposed by Shen *et al* [15] is employed to deform the exterior surface of the standard model M_0 to that of model M_i.

Those interior vertices in the tetrahedral mesh can be morphed to their corresponding location according their relative coordinates in the eigen space formed by their neighbors on the exterior surface. For an interior vertex v of the standard model M_0, given its neighbors on exterior surface are $p_0 \ldots p_k$, and $p'_0 \ldots p'_k$ are the corresponding positions of $p_0 \ldots p_k$ in model

M_i, the corresponding position v' of v in model M_i can be computed as following. Assume $F(x,y,z,o)$ is the eigen space formed by $p_0 \ldots p_k$, here x, y, z are the coordinate axes and o is the origin. Similarly $F'(x',y',z',o')$ is the eigen space formed by $p'_0 \ldots p'_k$. The eigen space is computed using the eigen vector of covariance matrix. Given u is the local coordinate of v in F, i.e. $u.x = (v - F.o) \bullet F.x$, $u.y = (v - F.o) \bullet F.y$, $u.z = (v - F.o) \bullet F.z$. By assuming v' has the same relative coordinate in F' as v in F, v' can be computed as $v'x = (u \bullet F'x') + F'o'x$, $v'y = (u \bullet F'y') + F'o'y$, $v'z = (u \bullet F'z') + F'o'z$.

6.2 Statistics Analysis and Prior Parameter Computing

After registering and matching the models in the training set, modes of variability both for 3D shape and volumetric density properties are computed using Principal Component Analysis (PCA) method. PCA method has been applied extensively to 2D and 3D shape and 2D density values [5, 7]. Basically in PCA method, each model in the training set is first represented by a vector Y_i $(i=1..M)$. Then the covariance matrix S of these vectors Y_i is constructed and k most significant eigenvectors are computed and used to characterize the variability of the training set. k can be chosen so as to explain a given proportion of variability. In General k is significantly smaller than the size of the model vector Y_i. After that any model can be approximated by $Y \approx \bar{Y} + Pb$, where $P=(p_1, p_2, \ldots, p_k)$ is the matrix of the first k eigenvectors and $b=(b_1,b_2,\ldots,b_k)^T$ is a vector of weights, refer to as model parameters. By varying the elements of b we can deform the model. Limits on the values of b_i can be constrained within $\pm 3\sqrt{\lambda_i}$ (λ_i is the standard deviation of b_i) to ensure that the deformed model is similar to those in the training set and doesn't go wild.

The shape property of our model can be written as a vector Y_s using the coordinates of its vertices. $Y_s = (x_1, y_1, z_1, \ldots x_n, y_n, z_n)^T$, hence Y_s is a $3n$ dimension vector, where n is the number of vertices in the model. The density property of the model can also be described as a vector Y_d using the coefficients of the density function in each tetrahedron. $Y_d = (c_1^{(1)}, c_2^{(1)}, \ldots c_f^{(1)}, \cdots, c_1^{(t)}, c_2^{(t)}, \ldots c_f^{(t)})^T$, where f is the number of coefficients in the density function, and t is the number of tetrahedra in the model. We can apply PCA method to both the shape vector Y_s and density vector Y_d respectively. So now we have $Y_s \approx \bar{Y}_s + P_s b_s$ and $Y_d \approx \bar{Y}_d + P_d b_d$. We can correlate the shape vector and density vector using Cootes's method in Active Appearance model as following. First forge a combined vector $b=[W_s b_s, b_d]^T$, where W_s is a diagonal matrix of weights to compensate that difference between shape and density parameter. Then apply another PCA on the combined vector. But it is very difficult to get W_s in our model because b_s and b_d are from different domain (one is coordinate, one is polynomial coefficient). However by further examining the density function as a Bernstein polynomial on the barycentric coordinate of the tetrahedron, it is a symmetric function on a normalized coordinate system. So the density function is shape invariant. Hence we can deform the Y_s and Y_d independently and get the model vector Y by simply concatenating them, i.e. $Y=[Y_s, Y_d]^T$.

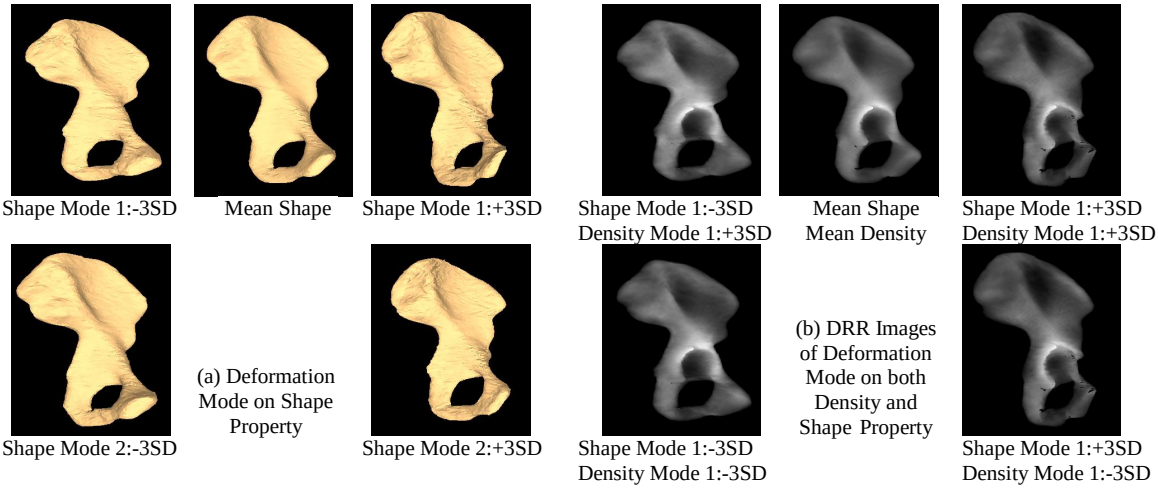


Figure 14. Deformation of Pelvis Prior Model

We have built the prior model for both the femur and the pelvis. The femur prior model is built from ten training models and the pelvis prior model from eight training models. Figure 14.a shows the exterior surface of the model after applying ± 3

standard deviations of the first two modes of the shape vectors to the mean model. Figure 14.b shows the DRR of the model after applying ± 3 deviations of the first mode of the shape vector and the first mode of the density vector respectively. We are working on getting more data and building a larger training database to capture more anatomical variability.

7. Discussion and Future Work

In this paper we have presented the first phase of our bone density atlas research. We proposed an efficient and automatic method to construct the tetrahedral mesh from bone contours and introduced an enhanced technique for tetrahedral mesh smoothing. We also assigned a density function to the tetrahedron and showed that it is reasonably accurate and efficient in storage. We then simplified the tetrahedral mesh based on edge collapsing scheme and build a hierarchical data structure to represent multiple resolutions of the density model. We also computed the prior statistical model using PCA method to characterize both the shape and density variability of the model.

We will continue our work on building the density atlas. We will investigate the deformation rule of the tetrahedral mesh. We are acquiring more data to build a larger database of the training models. We will register our statistical model to intra-operative patient data like C-Arm x-ray images. Ultimately we will apply the density atlas to various application such us deformable 2D-3D registration, surgical planning etc.

ACKNOWLEDGEMENTS

This work was partially funded by NSF Engineering Research Center grant EEC9731478. We thank Integrated Surgical System for providing the femur data and Shadyside Hospital for the pelvis data. We specially thank Dinggang Shen for the deformable surface matching algorithm. We also thank Stephen Pizer, Sandy Wells, Robert VanVorhis, Jerry Prince, and Chengyang Xu for their useful discussions.

REFERENCE

[1] G. Subsol, J.-P. Thirion, and N. Ayache, “First Steps Towards Automatic Building of Anatomical Atlas,” INRIA, France 2216, March, 1994

[2] C. B. Cutting, F. L. Bookstein, and R. H. Taylor, “Applications of Simulation, Morphometrics and Robotics in Craniofacial Surgery,” in *Computer-Integrated Surgery*, R. H. Taylor, S. Lavallee, G. Burdea, and R. Mosges, Eds. Cambridge, Mass.: MIT Press, 1996, pp. 641-662.

[3] M. Chen, T. Kanade, D. Pomerleau, and J. Schneider, “3-D Deformable Registration of Medical Images Using a Statistical Atlas,” Carnegie Mellon University, Pittsburgh, PA CMU-RI-TR-98-35, December, 1998.

[4] S. M. Pizer, “Object Shape in Medical Images,” A Tutorial for IEEE International Summer School on Biomedical Imaging 1999.

[5] T. F. Cootes and C. J. Taylor, “Statistical Models of Appearance for Computer Vision,” 2000.

[6] T. F. Cootes, C. Beeston, G. J. Edwards, and C. J. Taylor, “A Unified Framework for Atlas Matching using Active Appearance Models,” presented at IPMI, pp322-333, 1999.

[7] M. Fleute and S. Lavallee, “Nonrigid 3-D/2-D Registration of Images Using Statistical Models,” presented at MICCAI 1999, Cambridge, UK, pp 138-147, 1999.

[8] I. J. Trotts, B. Hamann, and K. I. Joy, “Simplification of Tetrahedral Meshes with Error Bounds,” *IEEE Transactions On Visualization and Computer Graphics*, vol. **5**, pp. 224-237, 1999.

[9] J.-D. Boissonnat and B. Geiger, “Three dimensional reconstruction of complex shapes based on the Delaunay triangulation,” Inst. Nat. Recherche Inf. Autom., Le Chesnay, France (INRIA) 1697, May, 1992.

[10] D. Meyers, S. Skinner, and K. Sloan, “Surfaces from Contours,” *ACM Transactions on Graphics*, vol. **11**, pp. 228-258, 1992.

[11] E. N. Marieb, *Human Anatomy and Physiology*, 2nd ed: the Benjamin/Cummings Publishing Company, 1992.

[12] J. Yao, “Tetrahedral Mesh Modeling of Density Data for Anatomical Atlases and Intensity-Based Registration,” presented at MICCAI 2000, Pittsburgh, PA, pp531-540, 2000.

[13] L. A. Freitag, “On Combining Laplacian and Optimization-based Mesh Smoothing Techniques”, *Trends in Unstructured Mesh Generation, ASME Applied Mechanics Division*, pp. 37-44, 1997.

[14] P. Lacroute, “Fast Volume Rendering Using a Shear-Warp Factorization of the Viewing Transformation,” in *Computer Science*. California: Stanford, 1995.

[15] D. Shen and C. Davatzikos, “Adaptive-Focus Statistical Shape Model for Segmentation of 3D MR Structures,” presented at MICCAI 2000, Pittsburgh, PA, pp.206-215, 2000.