Extrapolation of Missing Craniofacial Skeletal Structure via Statistical Shape Models

Project #1 Final Report, EN.600.646 Spring 2014

Robert Grupp
robert.grupp@gmail.com

Hsin-Hong Chiang
cblta@hotmail.com

Dr. Yoshito Otake (Mentor)
Ryan Murphy (Mentor)
Dr. Russell Taylor (Mentor)
Dr. Mehran Armand (Mentor)

May 9, 2014

1. Introduction

Statistical Shape Models (SSMs), or Statistical Atlases, are the primary topic of this project, however several sub-topics include surface and volumetric deformable registration, segmentation, feature extraction, and surface extrapolation. The goal is to design and implement a method for extrapolating missing anatomical craniofacial skeletal structure with the use of a SSM of the human cranium.

We intend for the algorithms developed as part of this project to be applied in the field of craniofacial surgery. The procedure of interest is craniofacial transplantation, which is the process of transplanting a donor's craniofacial soft tissue, and possibly bone structure, onto a patient that has been subject to some severe craniofacial deformation. The surgery aims to restore lost functionality to the patient, such as the ability to smell, speak, or eat solid food [1]. By allowing the patient to participate in society as a "normal" individual, the surgery may help alleviate psychosocial traumas developed by the patient upon their disfiguration [1]. Figure 1 shows preoperative and postoperative views of a transplant recipient. After undergoing face transplant surgery, "the patient was able to breathe through her nose, smell, taste, speak intelligibly, eat solid foods, and drink from a cup [2]." Once a potential donor has been identified, the decision to perform surgery must be made within a very short time frame (24-36 hours)[3]. Amongst other factors, the skeletal structure of the patient and donor is compared for compatibility via cephalometric measurements [3]. With these cephalometric measurements and medical images, such as CT imagery, surgeons develop a plan for specific cuts to the donor and patient. We propose for true

cephalometrics to be estimated via an extrapolation of the patient's missing facial bone structure. With this additional information, the surgeon may create a plan of patient and donor osteotomy cuts that results in a better postoperative outcome for the patient. It should be noted that the estimation of the patient's ideal skeletal structure is an attempt to maximize the aesthetic quality of the surgical result, and makes no guarantee regarding the postoperative biomechanics of the patient. It is plausible that higher aesthetic quality will imply "good" biomechanics, but this will need to be the topic of further study.

Figure 1: A craniofacial transplant recipient. Left preoperative photograph, middle preoperative CT, right postoperative photograph

SSMs have been a common topic in the medical imaging field since their introduction as Active Shape Models (ASMs) and Active Appearance Models (AAMs) by Cootes and Taylor [5] [6]. In this work we are concerned only with surface structure and therefore limit ourselves to ASM applications. SSMs have been used to intraoperatively estimate the surface of a patient's femur without any prior surface data [7]. This is done via the use of a tracking tool and generating a point cloud about a local region of the bone, which is matched to a SSM of the femur. The sparse collection of points is not used once the bone structure has been estimated. In our application, we shall start with a CT image of the patient as prior and a dense sampling of normal skeletal structure may be used as the input point cloud. This point cloud is then used to estimate the entire skeletal structure. Contrary to [7], the prior structure is a dense representation of the patient's true boney structure, therefore we wish to retain this structure while somehow integrating the estimate of the missing, or "unknown," structure. This idea has been explored previously by Chintalapani in order to extrapolate a full representation of a patient's pelvis given a partial CT volume as prior knowledge [8]. The major issue with the extrapolation performed in this case was the presence of a sharp, non-smooth, or "jaggy," transition

from the true structure of the patient to the extrapolated structure. Since it will result in inaccurate cephalometric estimates, this non-smooth transition is unsatisfactory for the purposes of the craniofacial transplant surgery, therefore we wish to develop a smooth extrapolation method. Due to the time limit of the semester, we were unable to complete the smooth extrapolation technique, however we were successful in the creation of a SSM of the skull and skin surfaces, with a basic extrapolation technique. The remainder of this report provides a detailed description of the data used, methods and algorithms used for data manipulation, and the corresponding results of this data manipulation.

2. Materials

To create the statistical shape model (SSM), training CT data was obtained from the The Cancer Imaging Archive (TCIA), specifically the "Quantitative Imaging Network (QIN) Head-Neck" [9] and "Head-Neck Cetuximab" [10] datasets. The TCIA data is available freely to the public and allowed us to bypass any approval process to process medical image data collected at a Johns Hopkins Medical Institution. Due to the nature of TCIA, each patient is suffering from some form of Cancer, however it is unknown if the Cancer had spread to a patient's bone and caused structural deformation. We obtained a total of 303 CT images, 191 from the QIN Head-Neck dataset and 112 from the Cetuximab dataset. This count includes multiple CT images from specific patients and images that do not have a complete scan of the head. After identifying images without complete coverage of the head, 59 images remained. Multiple CTs from a single patient were kept in the dataset, so that the instance with a more accurate registration would be used as training data for the SSM. The QIN Head-Neck dataset provided relatively comprehensive patient metadata, such as gender, age, and weight, however the majority of the Cetuximab dataset was missing this data. Unfortunately, of the 59 images chosen for processing, 54 images were from the Cetuximab dataset, and of these only 16 had valid values for the aforementioned metadata. Due to the small amount of complete CT images with valid metadata, we did not restrict our SSM training data to only include images with valid metadata annotations.

In order to evaluate our reconstructions on non-synthetically disfigured data, a CT image of a patient eligible for a full face transplant was obtained courtesy of Dr. Chad Gordon. The patient is a 28 year old male that had experienced severe trauma to the face and has undergone reconstructive surgery of the nose, mandible, and orbits via conventional autografts. The current skull of this patient is depicted in figure 2.

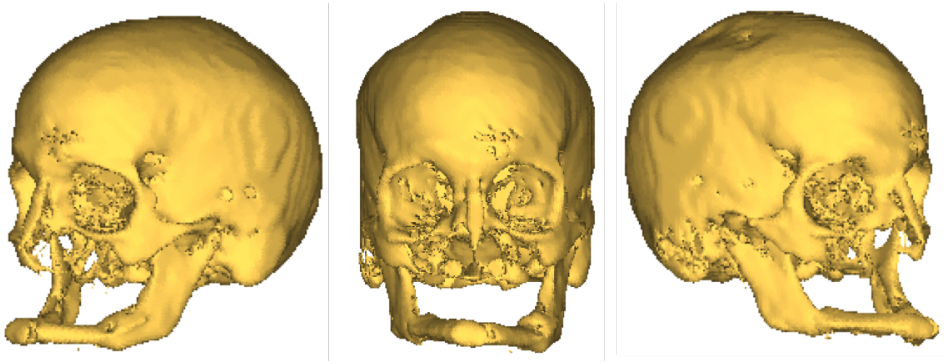


Figure 2: The skull surface of a patient that has experienced severe trauma to the face. The patient's mandible has been reconstructed with via an autograft of the fibula. The nasal bone was reconstructed via an autograft from a portion of the neurocranium (note the recessed portion on the right of the patient's neurocranium). The patient's orbits also appear abnormally large and the result of reconstruction. CT courtesy of Dr. Chad Gordon.

3. Methods

3.1. Overview

The algorithmic details and technical approaches for each stage of processing are described in this section. The core extrapolation processing implemented as a result of this study is depicted in figure 3. The extrapolation process relies on the existence of a Statistical Shape Model (SSM) of a normal and complete human skull or skin surface of the head. A CT image of a patient with some disfigurement is used to generate the appropriate surface mesh and any abnormal regions are manually masked out. This masked mesh is registered to the SSM, yielding an estimate of what the patient's normal surface should resemble. The abnormal regions, that were not valid in the patient's mesh, are then replaced with the appropriate estimates from the SSM.

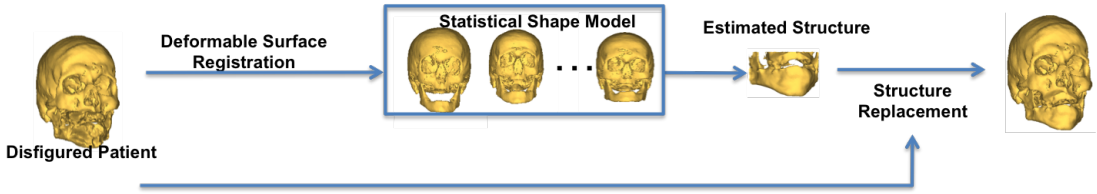


Figure 3: A high level overview of the extrapolation process.

A set of training images, and their corresponding surface meshes are used to build

the SSM. A Principal Component Analysis (PCA) is performed on the set of training meshes; this requires the set of training meshes to be topologically consistent. Topologically consistent implies that the same structure of triangles is used in each mesh, however the vertex values are allowed to vary. In order to obtain this training set a template mesh is chosen and each of it's vertices is displaced to match the corresponding anatomical locations of another image's surface. The correspondence between the template mesh and every other subject is obtained via a volumetric deformable registration from the template CT image to every other CT image. An overview of the steps required to construct a SSM are shown in figure 4. Each of the processing methods are now described in further detail.

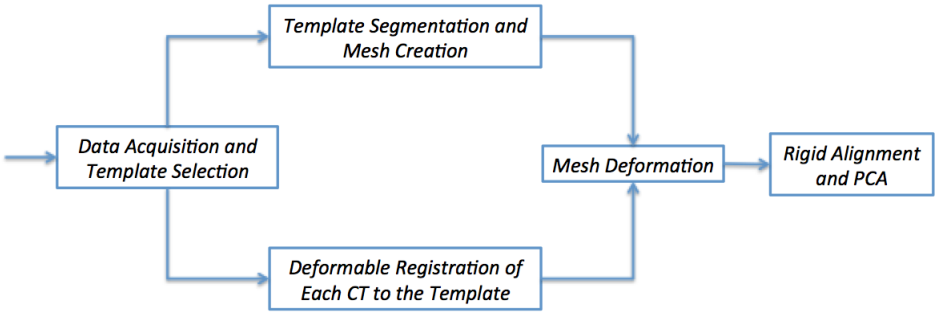


Figure 4: A high level overview of the SSM creation process.

3.2. Data Conversion

All data was obtained in the Digital Imaging and Communications in Medicine (DICOM) format[11], but was converted to the Neuroimaging Informatics Technology Initiative (NIFTI) file format [12] for additional processing. By converting to NIFTI format, we avoid the overhead of DICOM filesystem I/O, and instead work with a single file for each CT image. Each DICOM series was converted to Hounsfield Units (HU) and resampled to have an isotropic 2 mm voxel size via a B-Spline interpolator of order 3. In order to save space and processing time, DICOM conversion and resampling was performed in the same step. The 2 mm voxel size was chosen as a compromise between the ≈ 1 mm in-slice resolutions and the ≈ 3 mm spacing between slices. We chose to store intensity values in their 32-bit IEEE floating point representation when saving to NIFTI format.

After resampling, the images are cropped to have uniform dimensions. This is done first by manually identifying a tight bounding box about the most anterior and posterior portions of the skull, the top of the skull, and the bottom of vertebrae C4. Once bounding boxes for each image are calculated, a global bounding box is obtained by taking the maximum lengths in each dimension (coronal, sagittal, axial). The bounding boxes for each image are then expanded until the box is equal in size to the global bounding box. Each box dimension is expanded equally in the positive and negative dimensions. If an

image boundary is encountered when expanding an image's bounding box, values of -1000 HU are filled into the cropped image. Since each bounding box forces the patient's head to be located in the center of each cropped image, this process serves as a coarse translational alignment of the data. A graphical depiction of this cropping is shown in figure 5. The final cropped image dimensions used for our experiments was $106 \times 136 \times 154$.

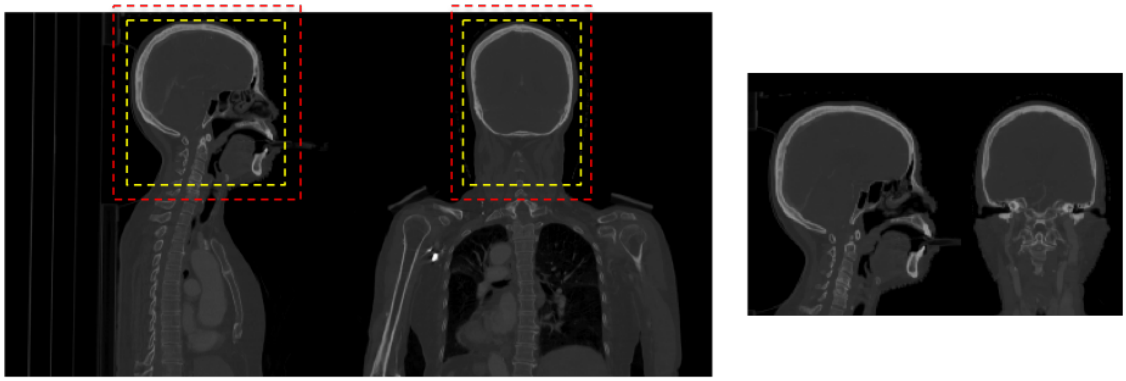


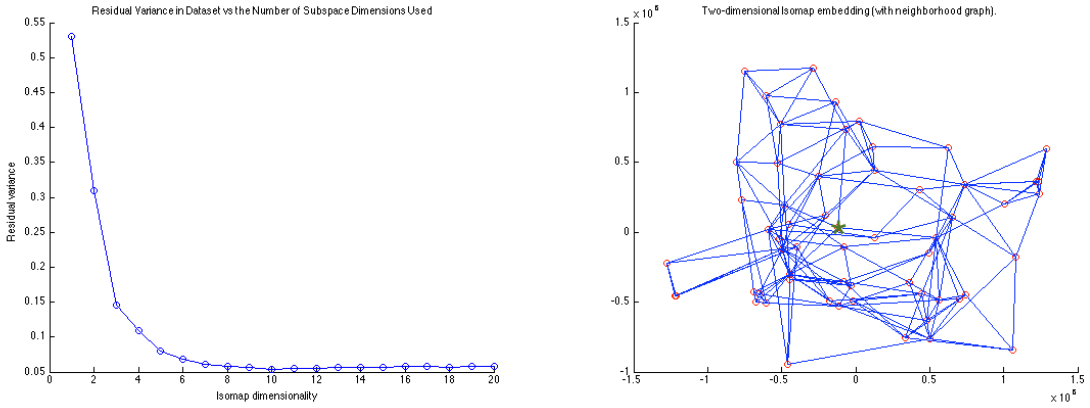
Figure 5: An example of cropping a resampled CT image. Coronal and sagittal slices of the uncropped, resampled image are shown in (a), with the original tight bounding box shown in yellow and the global bounding box shown in red. The regions of the global bounding box that are not contained within the image bounds will be filled with a constant value of -1000 HU. The cropped result is shown in (b). CT is from Patient 0522c0555_00 of the Cetuximab Dataset.

3.3. Template Selection

Due to the bias imposed by a template image on the deformable registration results, the selection of the template image is of some importance. We experimented with two methods of template selection: a random selection, and a selection performed via Isomap analysis [13]. Clearly, the random selection is far from ideal and most likely sub-optimal, however it will serve as a reference for comparison to the Isomap-based selection. The template selected at random was 0522c0845_00 and the template selected via Isomap analysis was 0522c0708_00, both from the Cetuximab Dataset.

Isomap is a nonlinear dimensionality reduction technique. By representing the CT image data in a subspace that best describes the relationship between each image, the image closest to the subspace origin is an estimate of the image that is similar to all images, and therefore a reasonable choice as a template. Each of the 59 resampled and cropped CT images is treated as a vector of $\mathbb{R}^{2,220,064}$ by "stacking" each image's intensity value into a large image column vector. Euclidean distances are computed between every combination

of image vector pairs; these distances are used as input to the Isomap algorithm. The residual variance is an indicator of Isomap performance, and indicates the amount data that is not representable by the Isomap subspace. The Isomap parameter K corresponds to the number of nearest neighbors to use when constructing the initial edges in a graph. After empirical analysis of values of $K \in \{1, 2, \ldots, 9\}$ and dimensions in $\{1, 2, \ldots, 20\}$, $K = 4$ with 10 dimensions was the chosen subspace as it resulted in the lowest residual variance. The CT image with subspace coordinates closest to the origin is then chosen as the template image. Isomap plots our data are shown in figure 6.



(a) (b)

Figure 6: Plots depicting the result of Isomap analysis on the resampled and cropped CT images. The residual variance for $K = 4$ is shown in (a). The two-dimensional representation of the Isomap graph is shown in (b), with the image closest to the origin represented by a green asterisk (Patient 0522c0708_00 of the Cetuximab Dataset).

3.4. Segmentation and Mesh Creation

Each template was manually segmented using the 3D Slicer application [14]. The 3D Slicer Editor module provides the functionality to perform basic label editing, threshold, connected components, and binary morphological operations. Throughout the segmentation process, a preview surface may be generated with the 3D Slicer Model Maker module; this helps to ensure that the labeling process yields a realistic surface. The bone label image is created via the following steps:

1. Perform a uniform thresholding of the image, labeling voxels with intensity in $[300, \infty)$ HU as 1, and all other voxels as 0
2. Manually remove imaging artifacts in the mouth (performed in the axial plane)

3. Manually fill in portions of the zygomatic bone that were discarded due to the thresholding (performed in the axial plane)

4. Manually erase C1, C2, vertebra from the label map (performed in the sagittal plane); this disconnects the skull from the vertebral column

5. Perform a connected-component analysis to relabel any new regions (keeping the skull as label 1)

6. Discard any labels not equal to 1 and save the label image to disk

An example of the bone segmentation process is shown in figure 7. The skin label image is created via the following steps:

1. Perform a uniform thresholding of the image, labeling voxels with intensity in $[-300, \infty)$ HU as 1, and all other voxels as 0

2. Manually fill in any holes (voxels labeled 0) in the interior of the body (performed in the sagittal and coronal planes)

3. Manually erase any imaging artifacts that are connected to the skin surface (performed in the sagittal and coronal planes)

4. Manually erase the labels in 3 contiguous axial slices starting at the bottom of C4 moving inferior; this disconnects the shoulders and base of the neck from the head

5. Perform a connected-component analysis to relabel any new regions (keeping the head as label 1)

6. Discard any labels not equal to 1 and save the label image to disk

The "hole filling" performed during skin segmentation was done so that the mesh created from the label map is "water tight" and represents a relatively simple surface when contrasted with the bone mesh. The "hole filling" was performed manually, due to the morphology operators in 3D Slicer ultimately result in the lose of detail on the skin surface. An example of the skin segmentation process is shown in figure 8.

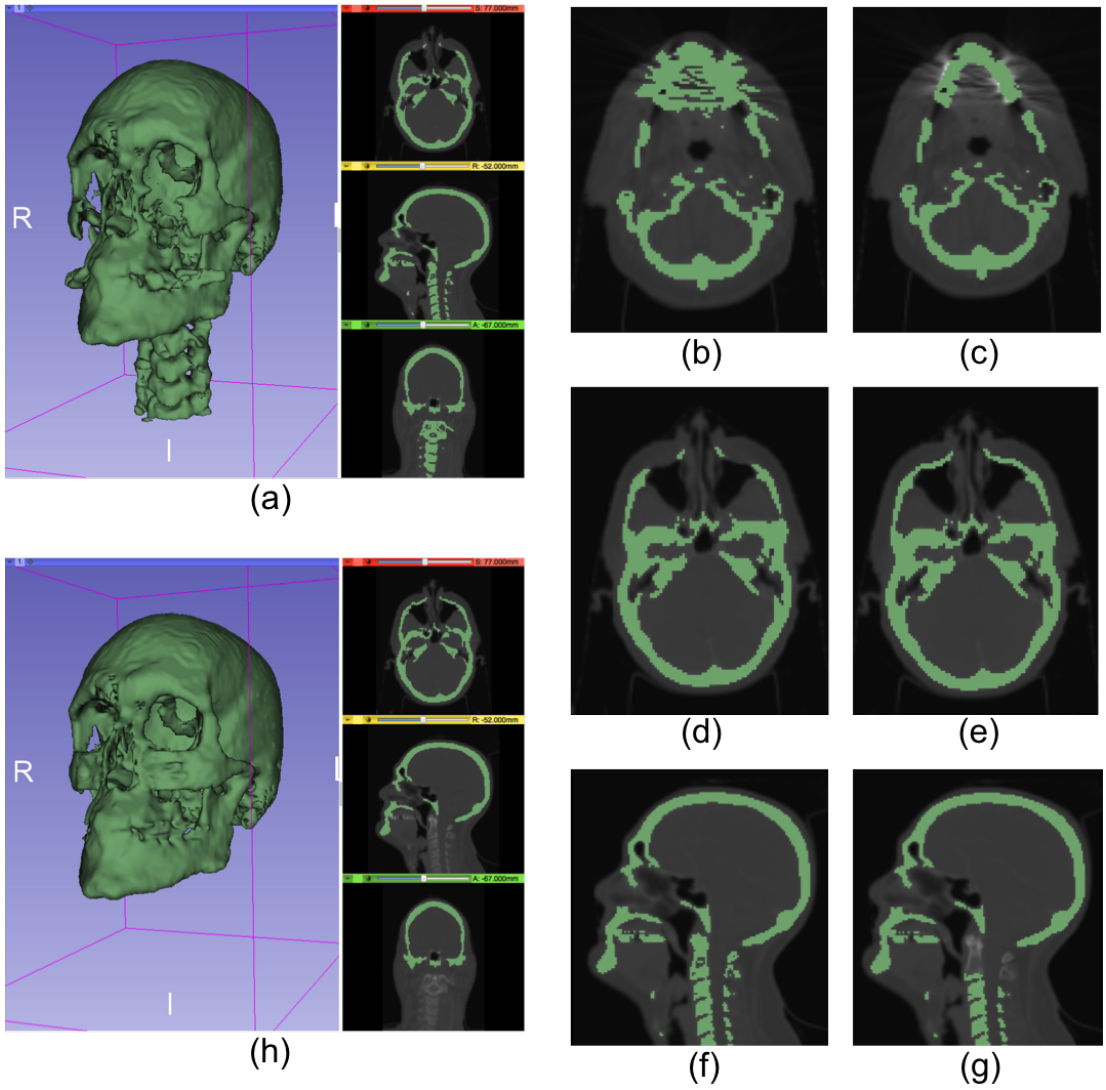


Figure 7: An example of manually segmenting the bone in a resampled CT image with 3D Slicer. The label map and preview mesh immediately after thresholding are shown in (a); note the artifacts about the mouth and missing zygomatic bone sections. Image artifacts in the mouth are shown in (b) and the result after manual removal is shown in (c). Discarded regions of the zygomatic bone are shown in (d) and the result after manual labeling is shown in (e). The label map prior to erasing C1 and C2 is shown in (f) and the result of their removal is shown in (g). The final label image and preview mesh are shown in (h). The base image is from the third bootstrapped template starting from Patient 0522c0845_00 of the Cetuximab Dataset.

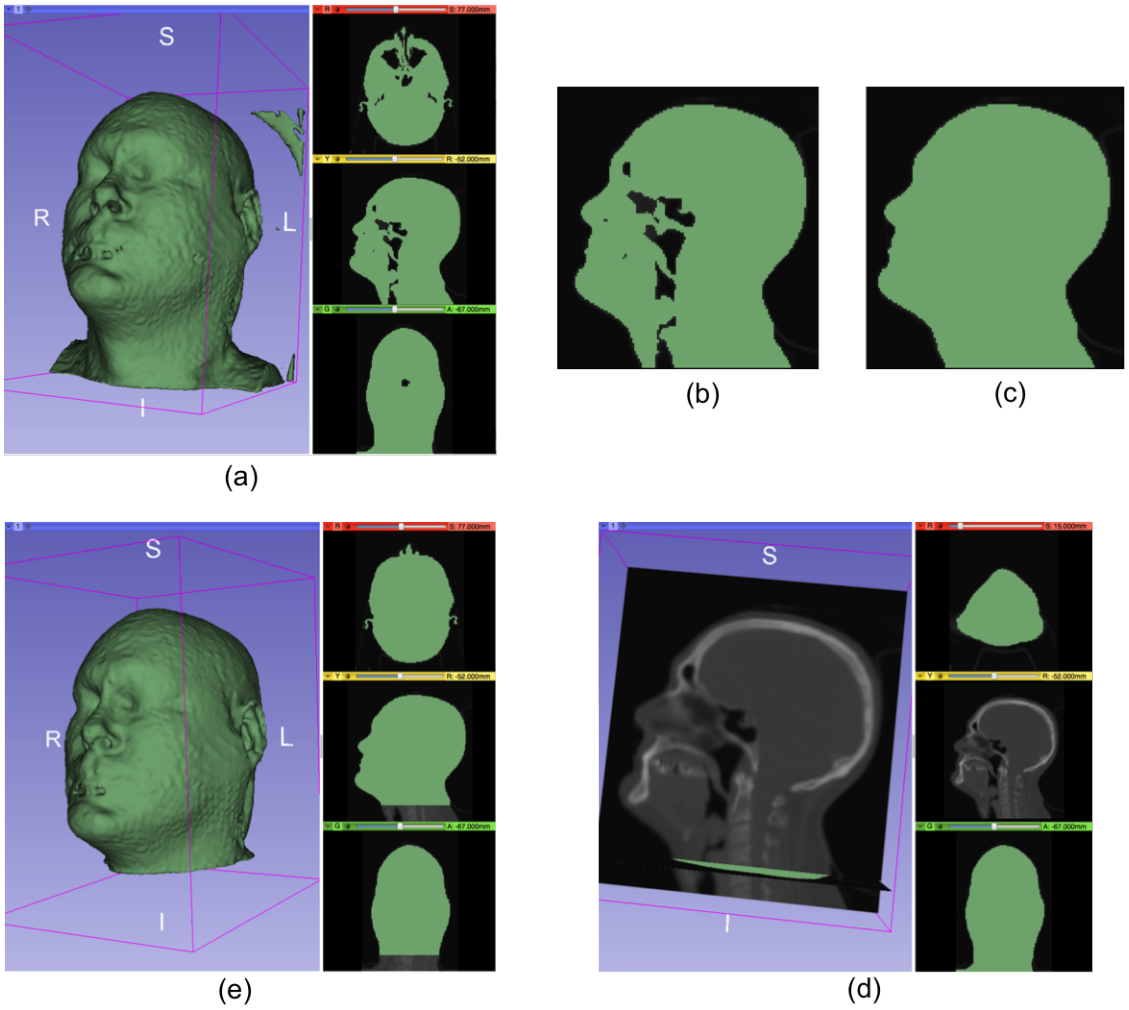


Figure 8: An example of manually segmenting the skin in a resampled CT image with 3D Slicer. The label map and preview mesh immediately after thresholding are shown in (a) and (b). Manual "hole filling" on a single sagittal slice is shown in (c). Erasing axial slices is shown in (d) and highlights the use of 3D Slicer's display capabilities. The final label image and preview mesh are shown in (e). The base image is from the third bootstrapped template starting from Patient 0522c0845_00 of the Cetuximab Dataset.

Once a label image is created, a triangular surface mesh is created via the following steps:

1. An isosurface of the label map is extracted via an edge-based algorithm (The MATLAB `isosurface` command)

2. The mesh is smoothed with an accurate curvature flow approach [15][16]; three iterations are performed

3. The mesh is decimated, keeping 75% of the faces and preserving shape (The MATLAB `reducepatch` command)

Figure 2 depicts a patient mesh created from this process. Once saved to disk, the meshes created by 3D Slicer were in coordinate system that did not correspond with our original image, therefore we chose the described method as it left the mesh coordinates in the image coordinate system.

3.5. Deformable Volumetric Registration

We use ANTs (Advanced Normalization Tools) to do the deformation volumetric registration for us. It is an open source code and provides lots of toolkits for biomedical analysis, such as registration and segmentation. What is more, it also provides binary files which can directly run in operating system. Before using ANTs, we have to set up ITK (Insight Segmentation and Registration Toolkit) first, since it is based on ITK.

For deformable volumetric registration, we use binary file, antsRegistration, to do it. It is a powerful function which provides couples of registration such as rigid registration and deformation registration. Moreover, there are several metrics that we can choose to optimize such as cross correlation and mutual information. Below is the parameter that we can set up for registration.

- -d: It is the dimension of image.
- -m: This metric depicts which metric we want to optimize, such as neighborhood cross correlation, CC, and mutual information, MI. It can also accept two metric for optimization. As we can see the last registration, we use the neighborhood cross correlation and mutual information for metric.
- -t: It is used to set up what kind of registration we are going to use, such as rigid and deformation registration.
- -c: Specify the convergence threshold.
- -s: Determine the variance for Gaussian smoothing.
- -f: Specify the shrink factor for the virtual machine.
- -l: Estimate the learning rate.
- -u: Do histogram-matching before registration.

We do the registration four times in the command. Each of them is different. The first registration just tries to pull the moving image to reference image. The second one aligns the moving image to reference image. The third one tries to do the affine registration to get the better alignment before last step. The last one is the deformation registration which is also the most important one.Here, we use the Symmetric Diffeomorphic Image Registration. It not only does the registration but also gives us the deformation maps. After getting deformation map, we can use them to get the meshes of training data set and do the bootstrapping method. Table 1 shows the parameters that we use.

	-t	-m	-c	-s	-f	-l	-u
1	Translate	CC	1.e-8	4x2x1 voxel	3x2x1	No	No
2	Rigid	CC	1.e-8	4x2x1 voxel	3x2x1	No	No
3	Affine	CC	1.e-8	4x2x1 voxel	3x2x1	No	No
4	SyN	CC and MI	0	1x0.5x0 voxel	4x2x1	Yes	Yes

Table 1: The parameters for the registration.

The key registration function is the Symmetric Normalization (SyN) component. The diffeomorphic nature of SyN ensures that a successful registration is both invertible and smooth, which will result in a higher likelihood of anatomical structure preservation.

In an attempt to automatically determine registration success we compute a Jaccard statistic using segmentations of the template and subject. The manual segmentation of the template's skull is warped to subject's space and compared against a simple segmentation derived from a labeling of all voxels in $[300, \infty)$ as bone. The Jaccard index between two coordinate sets is defined in (1):

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{1}$$

3.6. Mesh Deformation

After deformable volumetric registration is completed, a mesh of the template image may be deformed into a mesh of any of the other registered subjects. This is completed via the use of the displacement field that "pulls" a subject into the space of the template. At each voxel of the displacement field a vector is stored that indicates where this voxel maps to in the subject space. In the way that we constructed the template mesh, the vertex coordinates are with respect to the template image's, and therefore this displacement field's, coordinate system. Therefore, adding an appropriate displacement vector to each vertex will warp the template mesh into a mesh representing the subject's structure. If a vertex coordinate is located precisely on an integer voxel coordinate, then the displacement vector at that particular voxel is used. However, most vertex coordinates have subvoxel

(fractional) values; when this occurs interpolation is used to choose an appropriate displacement vector. We utilize tri-linear interpolation, so that the resulting displacement vector is a weighted linear combination of the displacement vectors at the nearest eight voxel coordinates.

3.7. Mesh Comparison Metrics

As part of the evaluation of each SSM, several distance and error measures are required. Several mesh comparison metrics, such as mean surface distance and vertex error, are defined in [8]. We shall utilize these in our processing and they are re-defined below for convenience. We also utilize the maximum surface distance measurement. Given two meshes, $M_1 = (V_1, F_1)$ and $M_2 = (V_2, F_2)$, we may define several comparison methods between the two. Define the operator returning the closest point from a specific triangle as $\mathcal{T}$. The operator, C_{M_1,M_2}, computing the closest point of a vertex, $v \in V_1$, of mesh M_1 to the surface of mesh M_2 may now be defined as:

$$C_{M_1,M_2}(v) = \min_{f \in F_2} \|\mathcal{T}(v, f) - v\|_2 \tag{2}$$

Via an efficient computation of $\mathcal{T}$ [17] and the use of a KD-Tree to represent the surface of a mesh [18], (2) is computed efficiently. The mean surface distance between the vertices of one mesh, M_1, to the surface of the other, M_2, may be defined as:

$$D_{\text{mean}}^{(\text{surface})}(M_1, M_2) = \frac{1}{N_{V_1,M_1}} \sum_{v \in V_1} C_{M_1,M_2}(v) \tag{3}$$

The maximum surface distance between any vertex of one mesh, M_1, to the surface of the other, M_2, may be defined as:

$$D_{\text{max}}^{(\text{surface})}(M_1, M_2) = \max_{v \in V_1} C_{M_1,M_2}(v) \tag{4}$$

(3) and (4) are biased by the choice of using the vertices in M_1 as sample points, therefore the same computations may be performed with the vertices of M_2 as sample points and using the surface of M_1. The new expressions of mean surface distance and maximum surface distance are shown in (5) and (6), respectively. (6) represents the Hausdorff distance between two meshes.

$$\widehat{D}_{\text{mean}}^{(\text{surface})}(M_1, M_2) = \max\left\{D_{\text{mean}}^{(\text{surface})}(M_1, M_2), D_{\text{mean}}^{(\text{surface})}(M_2, M_1)\right\} \tag{5}$$

$$\widehat{D}_{\text{max}}^{(\text{surface})}(M_1, M_2) = \max\left\{D_{\text{max}}^{(\text{surface})}(M_1, M_2), D_{\text{max}}^{(\text{surface})}(M_2, M_1)\right\} \tag{6}$$

When the meshes are topologically consistent (F_1 is equal to F_2 and $N_{V_1,M_1} = N_{V_2,M_2}$), we may define the mean vertex distance:

$$D_{\text{mean}}^{(\text{vertex})}(M_1, M_2) = \frac{1}{N_{V_1,M_1}} \sum_{i=1}^{N_{V_1,M_1}} \|v_{1,i} - v_{2,i}\|_2 \tag{7}$$

3.8. Mesh Alignment

Prior to performing statistical analysis on our set of topologically consistent training meshes, they must first be aligned so that they have a common pose, and optionally a common scale. Alignment to a common pose indicates an alignment via rigid transformations, and the addition of scale in the alignment process requires the use of a similarity transform. A generic algorithm for this alignment is shown in Algorithm 1. Since our training meshes are topologically consistent and derived from a single template, there is a direct correspondence between the vertices of two meshes. This correspondence allows the use of Horn's Quaternion Method [19] for the transformation computations in lines 10 and 12 of Algorithm 1. This saves significant computation time compared to computing the transform with another method, such as the Iterative Closest Point method [20].

Algorithm 1 Mesh Alignment

1: **function** AlignMeshes(Input Meshes: $\{M_1, M_2, \ldots, M_N\}$, Similarity Flag: S, Termination Threshold: ε)
2: **for** $i = 1, \ldots, N$ **do**
3: $M'_i \leftarrow M_i$ with centroid adjusted to be at the origin
4: **end for**
5: $\overline{M} \leftarrow M'_1$
6: $\varepsilon' \leftarrow 2\varepsilon$
7: **while** $\varepsilon' > \varepsilon$ **do**
8: **for** $i = 1, \ldots, N$ **do**
9: **if** S is True **then**
10: $T_i \leftarrow$ Compute the similarity transformation from M'_i to $\overline{M}$
11: **else**
12: $T_i \leftarrow$ Compute the rigid transformation from M'_i to $\overline{M}$
13: **end if**
14: $M'_i \leftarrow T_i(M'_i)$
15: $M' \leftarrow$ The mean mesh of $\{M'_1, M'_2, \ldots, M'_N\}$, by computing the mean value for each corresponding vertex
16: $\varepsilon' \leftarrow D^{(\text{vertex})}_{\text{mean}}(M', \overline{M})$ (see equation 7)
17: $\overline{M} \leftarrow M'$
18: **end for**
19: **end while**
20: **return** $\{M'_1, M'_2, \ldots, M'_N\}$
21: **end function**

3.9. Principal Component Analysis

Once training mesh alignment is complete, Principal Component Analysis (PCA) is performed on the vertices. As defined in Appendix A, $\mathcal{S}$ is the operator that transforms sets of vertices into a large column vector. Define $\mathbf{m}_i = \mathcal{S}(V_i)$ for $i \in \{1, 2, \ldots, N_M\}$. The mean vertex vector, $\overline{\mathbf{m}}$ is defined as:

$$\overline{\mathbf{m}} = \frac{1}{N_M} \sum_{i=1}^{N_M} \mathbf{m}_i \tag{8}$$

$M_S \in \mathbb{R}^{3N_V \times N_M}$

$$M_S = \begin{pmatrix} \mathbf{m}_1 - \overline{\mathbf{m}} & \mathbf{m}_2 - \overline{\mathbf{m}} & \cdots & \mathbf{m}_{N_M} - \overline{\mathbf{m}} \end{pmatrix} \tag{9}$$

After getting M_S, we need to compute the co-variance matrix and eigen-decomposition for variation modes. $M_S M_S^T \in \mathbb{R}^{3N_V \times 3N_V}$. Mesh with $60,000$ vertices, 32-bit floating point precision, $\approx$ 121 GB. It is impossible to compute the co-variance matrix directly, not to mention egein-decomposition.

$$\frac{1}{N_M - 1} M_S M_S^T = \frac{1}{N_M - 1} \widehat{U}_S D \widehat{U}_S^T \tag{10}$$

Actually, we don't have to compute the eigen-decomposition to get the eigen-values and eigen-vectors of co-variance matrix. We know that the relationship between SVD and eigen-decomposition. In Equation 11, it shows the standard form of SVD. Equation 12 depicts this relationship. Therefore, we can compute the SVD on M_S to get the eigen-values and eigen-vectors of co-variance matrix. U_S corresponds to the egein-vectors of co-variance matrix,$M_S M_S^T$.
$U_S \in \mathbb{R}^{3N_V \times 3N_V}$, $\Sigma_S \in \mathbb{R}^{3N_V \times N_M}$, and $V_S \in \mathbb{R}^{N_M \times N_M}$

$$\frac{1}{\sqrt{N_M - 1}} M_S = \frac{1}{\sqrt{N_M - 1}} U_S \Sigma_S V_S^T \tag{11}$$

$$M_S M_S^T = U_S \Sigma_S V_S^T V_S \Sigma_S^T U_S^T \tag{12}$$

However, computing the SVD is still not plausible. In Equation11, we can see that the size of U_S is same as the co-variance matrix. The trick is that the bottom $N_V - N_M$ rows of Σ_S are all zero, so we can remove them and only keep the first N_M rows of Σ_S and first N_M columns of U_S. This kind of trick is called "this SVD". In our program, we use C++ library, Eigen, to compute thin SVD. In Equation 13, the size of U_S' becomes $3N_V \times N_M$ which is much smaller than the size of U_S. Finally, it is possible to use SVD to get the eigen-vectors and eigen-values for variation modes. Equation 13 is the form of thin SVD.
$U_S' \in \mathbb{R}^{3N_V \times N_M}$, and $\Sigma_S' \in \mathbb{R}^{N_M \times N_M}$.

$$\frac{1}{\sqrt{N_M - 1}} M_S = \frac{1}{\sqrt{N_M - 1}} U_S' \Sigma_S' V_S^T \tag{13}$$

3.10. SSM to Patient Registration

Registration of a patient mesh to a Statistical Shape Model (SSM) may be formulated as optimization problem minimizing the mean surface distance between a rigid/similar transformed patient mesh and an instance of the SSM. The rigid/similar transform parameters, along with the mode weights, are the parameters which are optimized over. (14) depicts the mean surface distance objective function and (15) formulates the registration as an unconstrained optimization problem. In order to ensure an anatomically reasonable result, we impose the constraints that each mode weight must be within 3 standard deviation units of 0. The constrained optimization problem is shown in (16). We describe two methods for finding solutions to this problem, one when we have consistent mesh topologies and the other when the patient mesh topology differs.

$$R(s, \boldsymbol{\theta}, \mathbf{t}, \boldsymbol{\alpha}) = D_{\text{mean}}^{(\text{surface})}(T(M_P; s, \boldsymbol{\theta}, \mathbf{t}), (\boldsymbol{\mu}_A + \mathbf{U}_A\boldsymbol{\alpha}, F_A)) \tag{14}$$

$$\underset{s\in\mathbb{R},\boldsymbol{\theta},\mathbf{t}\in\mathbb{R}^3,\boldsymbol{\alpha}\in\mathbb{R}^{N_A}}{\operatorname{argmin}} \quad R(s, \boldsymbol{\theta}, \mathbf{t}, \boldsymbol{\alpha}) \tag{15}$$

$$\underset{s\in\mathbb{R},\boldsymbol{\theta},\mathbf{t}\in\mathbb{R}^3,\boldsymbol{\alpha}\in\mathbb{R}^{N_A}}{\operatorname{argmin}} \quad R(s, \boldsymbol{\theta}, \mathbf{t}, \boldsymbol{\alpha}) \text{ subject to } |\boldsymbol{\alpha}_i| \leq 3\sigma_{A,i} \text{ for } i \in \{1, 2, \ldots, N_A\} \tag{16}$$

3.10.1. Corresponding Vertices

Using the implicit vertex correspondences between the patient mesh, M_P, and the atlas mean, compute the optimal rigid/similarity transformation parameters s, $\boldsymbol{\theta}$ and $\mathbf{t}$ via Horn's Quaternion Method. Compute the rigidly/similarity transformed mesh: $M_P^* = T(M_P; s, \boldsymbol{\theta}, \mathbf{t}) = (V_P^*, F_P^*)$. As described in (17), the rigidly/similarity transformed patient mesh vertices are projected on the SSM mode vectors to obtain the mode weights ($\mathcal{S}$ represents the reshaping of a list of vertices into a column vector; see Appendix A).

$$\boldsymbol{\alpha} = \mathbf{U}_A^T(\mathcal{S}(V_P^*) - \boldsymbol{\mu}_A) \tag{17}$$

In order to reconstruct the SSM instance back into the space of the patient mesh, the mode weights are used to create a mesh $M_P^{(\text{est})\prime} = (\mathcal{S}^{-1}(V'), F_A)$, with V' computed as shown in (18).

$$V' = \mathbf{U}\boldsymbol{\alpha} + \boldsymbol{\mu}_A \tag{18}$$

The final estimate in the space of the original patient, $M_P^{(\text{est})}$, is computed using the inverse rigid/similarity transform:

$$M_P^{(\text{est})} = T^{-1}(M_P^{(\text{est})\prime}; s, \boldsymbol{\theta}, \mathbf{t}) \tag{19}$$

3.10.2. Non-Corresponding Vertices

We use a variant of the Active Shape Model (ASM) Search algorithm initially described by Cootes and Taylor [21]. This was initially implemented as part of Programming Assignment 5 from the Computer Integrated Surgery I class as Johns Hopkins [22]. The first stage of our ASM search performs a series of rigid/similarity alignments followed by a mean surface distance optimization over the mode weights until convergence of the change in mean surface distance. The next stage computes incremental rigid transformation parameter and mode weight updates. This is performed via a simultaneous linear least-squares optimization. This process is briefly described in Algorithm 2, however the reader is referred to [22] for specific mathematical formulations.

Algorithm 2 SSM to Patient Registration for Non-Corresponding Vertices

function SSM-REGISTRATION(Patient Mesh: M_P, SSM/Atlas: $\mathcal{A} = (\boldsymbol{\mu}_A, \mathbf{U}_A)$, Similarity Flag: S)
  $M_A \leftarrow \boldsymbol{\mu}_A$
  $s \leftarrow 1$, $\boldsymbol{\theta}, \mathbf{t} \leftarrow \mathbf{0}$
  while $D_{\text{mean}}^{(\text{surface})}(M_P, M_A)$ has not converged **do**
    $s, \boldsymbol{\theta}, \mathbf{t} \leftarrow \text{ICP}(M_P, M_A, S)$ (compute the rigid/similarity transformation parameters from M_P to M_A)
    while $\boldsymbol{\alpha}$ has not converged **do**
      Compute the $\boldsymbol{\alpha}$ that M_A adjusted by $\boldsymbol{\alpha}$ has minimum mean surface distance from $T(M_P; s, \boldsymbol{\theta}, \mathbf{t})$
      $M_A \leftarrow M_A$ adjusted by $\boldsymbol{\alpha}$
    end while
  end while
  while $\boldsymbol{\alpha}$ has not converged **do**
    Compute the incremental updates to $s, \boldsymbol{\theta}, \mathbf{t}, \boldsymbol{\alpha}$ so that M_A adjusted by $\boldsymbol{\alpha}$ has minimum mean surface distance from $T(M_P; s, \boldsymbol{\theta}, \mathbf{t})$
    $M_A \leftarrow M_A$ adjusted by $\boldsymbol{\alpha}$
  end while
  return $T^{-1}(M_A; s, \boldsymbol{\theta}, \mathbf{t})$
end function

In order to test the ASM with a valid SSM, random instantiations of the SSM were created and then input into our registration algorithm. Weights for each SSM mode were drawn from a normal distribution with zero mean and the standard deviation derived from the SSM, however if a value was chosen outside of 3 standard deviations, it was rejected and drawn again. Random scaling, rotation, and translation parameters were also chosen to simulate another coordinate system for the patient. Scale was drawn from a uniform distribution over $[\frac{2}{3}, \frac{3}{2}]$, rotation components were drawn from a uniform distribution over

$[-\frac{\pi}{4}, \frac{\pi}{4}]$ (radians), and translation components were drawn from a uniform distribution over $[-200, 200]$ (mm).

3.11. Synthetic Disfigurement

3.11.1. Introduction

Our goal is to extrapolate missing anatomical skeletal structure. To validate our method, we try to collect the CT scan of skull which contain severe craniofacial disfigurement. However, this kind of CT scan is rare. Although we have couple of them, it is still not enough for testing. For validation, how to synthetic severe craniofacial disfigurement becomes an important role in our project.

There are couples of method to synthesis disfigurement. The most easiest way is cutting part of skull off, but we want to try more realistic simulation. As we can see in Figure 9, the woman get gun shot in the face. In CT scan, there are several holes on the craniofacial skeletal structure due to the gun shot. In this section, we focus on how to simulate this kind of effect on the normal skeletal structure.

3.11.2. Synthetic Method

First of all, we choose which part of skull that we want to disfigure. The main idea of disfigurement is that we try add some random displacement on the vertex of this region. These random displacements are roughly in the same direction. For simplification, we make them move in the same direction. These displacements is used to simulate that the fraction of bullet pushes the skull and makes the holes on it. If we just add the random displacement on each vertex of mandible, we get the disfigurement as Figure 9.b. We can see that the holes are sharp. This is not the effect that we want.

To avoid the sharp effect, we consider another factor of displacement. The displacement of neighbor vertices should be similar. Therefore, we do the Gaussian smoothing between these displacement to make the more continuous and smoothing. After doing the Gaussian smoothing one the displacement, we can get the synthetic image Figure 9.c. As we can see, the sharp effect has been removed, and it becomes much more realistic.

To make the synthetic work easier, we also write the function which can just pass one point, the direction of displacement, the size of disfigured region. Then, this function will simulate the shooting in face. The direction of displacement and the point can determine which part of skull you want to shoot. The size of disfigured region is used to describe the firepower of shooting. In Figure 9.d, the blue circle is the point that we give, the blue line corresponds to the direction of displacement. After passing these argument, we can disfigure the mandible and get the synthetic image.

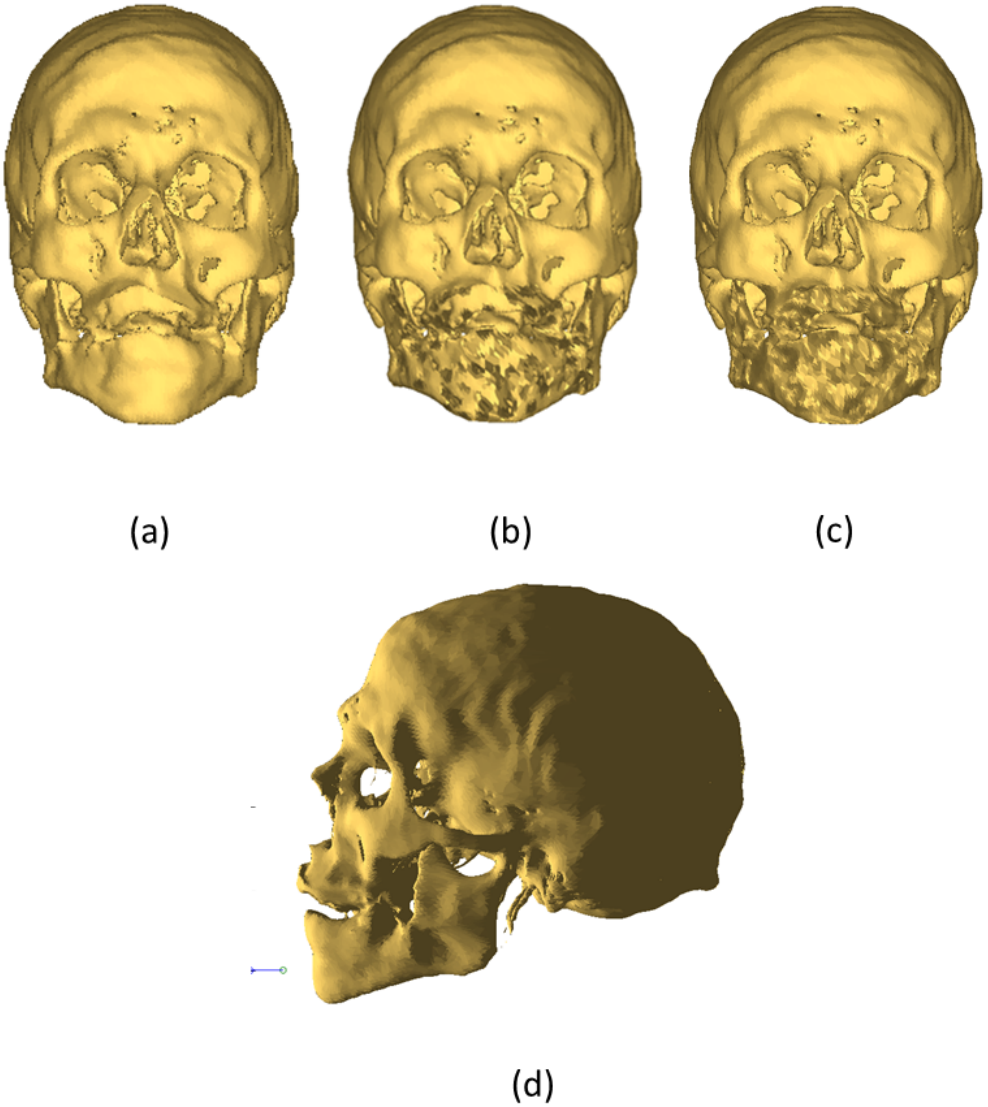


Figure 9: The example of how to synthesis severe craniofacial disfigurement. (a) is the normal craniofacial skeletal structure. (b) shows add the random displacement without Gaussian smoothing on mandible. (c) shows the sharp effect is alleviated by Gaussian smoothing. (d) shows how to simulate the gun shooting in our synthetic function.

3.12. Extrapolation

3.12.1. Introduction

In order to perform the extrapolation of missing mesh regions, SSM-to-Patient registration must first be performed. For patient meshes with differing topologies than the SSM, this initial registration will be used to perform a "topology transfer." The topology transfer functionality was not implemented as part of this project, therefore all extrapolation results are derived from meshes "left-out" from SSM generation.

3.12.2. Basic Atlas

The method described in this section closely follows that described by Chintalapani [8]. Let M_P be a triangular mesh representing the skull of a particular patient with some missing, or deformed, region $\mathcal{R}$. M_P may be represented by the sets of vertices and triangle faces, $\mathbf{V}_P$ and $\mathbf{F}_P$, respectively. Given a SSM of human skull surfaces, $\mathcal{A}$, patient-to-atlas registration may be computed to obtain an estimate, M_E, of the patient's ideal skull surface (e.g. without a deformity in $\mathcal{R}$). The estimate mesh is also represented by sets of vertices and triangles, $\mathbf{V}_E$ and $\mathbf{F}_E$, respectively. Furthermore, M_E may be partitioned into the "unknown" region, represented by $\mathcal{R}$, and the remaining "known" region as shown in (20).

$$M_E = [\mathbf{V}_E, \mathbf{F}_E] = \left[\begin{pmatrix} \mathbf{V}_E^{\text{known}} \\ \mathbf{V}_E^{\text{unknown}} \end{pmatrix}, \begin{pmatrix} \mathbf{F}_E^{\text{known}} \\ \mathbf{F}_E^{\text{unknown}} \end{pmatrix} \right] \tag{20}$$

Prior to performing the same partition on the original mesh, M_P, it is converted to match the mesh topology of M_E. M_P with updated topology is denoted $\widetilde{M}_P$, and is partitioned to match the "unknown" region $\mathcal{R}$ and remaining "known" region in (21).

$$\widetilde{M}_P = \left[\widetilde{\mathbf{V}}_P, \widetilde{\mathbf{F}}_P\right] = \left[\widetilde{\mathbf{V}}_P, \mathbf{F}_E\right] = \left[\begin{pmatrix} \widetilde{\mathbf{V}}_P^{\text{known}} \\ \widetilde{\mathbf{V}}_P^{\text{unknown}} \end{pmatrix}, \begin{pmatrix} \mathbf{F}_E^{\text{known}} \\ \mathbf{F}_E^{\text{unknown}} \end{pmatrix} \right] \tag{21}$$

A reconstruction is then possible by replacing the "unknown" vertices of the patient mesh with the "unknown" vertices of the atlas estimate, shown in (22)

$$\widetilde{M}'_P = \left[\begin{pmatrix} \widetilde{\mathbf{V}}_P^{\text{known}} \\ \mathbf{V}_E^{\text{unknown}} \end{pmatrix}, \mathbf{F}_E \right] \tag{22}$$

The major issue with $\widetilde{M}'_P$ is the possible presence of a transition that is not smooth, or "jaggy," [8] as shown in figure 10. In order to perform accurate surgical planning and biomechanical analysis, it is desired to minimize, or remove, these non-smooth transitions. It is hoped that the reviewed papers may be used for this purpose.

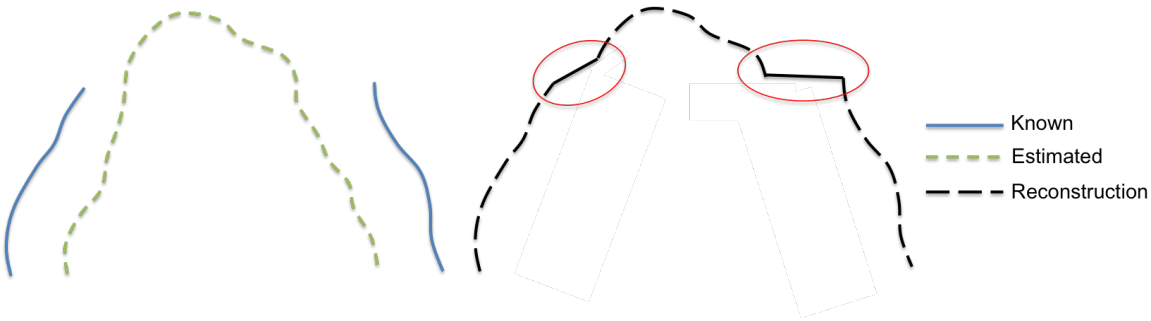


Figure 10: A 2D example of the atlas based extrapolation method. The left figure shows the original patient CT in solid blue and the atlas estimation in dashed green. The right figure depicts the extrapolated surface and the non-smooth transitions highlighted with red ellipses.

3.12.3. Multivariate Gaussian

Suppose we have a collection of N "support" meshes all with the same topology. We shall assume that these meshes have been aligned to yield: $\{\mathbf{m}^{(1)}, \mathbf{m}^{(2)}, \ldots, \mathbf{m}^{(N)}\}$. For the results shown, a similarity alignment was performed. Suppose also that we have a patient mesh. This mesh needs to be manipulated in some way to have the same topology as our N support meshes and aligned to the mean of these meshes. For this test case, the patient mesh was left-out of the meshes generated following volumetric deformable registration, and therefore had identical topology to the "support" meshes. Denote the patient mesh as $\mathbf{m}$. The patient mesh vertices may be partitioned into vertices whose values are valid, and vertices that are unknown, whose values we wish to estimate. Denote this partitioning as $\mathbf{m}_K$ and $\mathbf{m}_U$ for the known and unknown partitioned patient meshes, respectively. Since topologies are consistent, this partitioning may also be applied to the support meshes to yield $\{\mathbf{m}_K^{(1)}, \mathbf{m}_K^{(2)}, \ldots, \mathbf{m}_K^{(N)}\}$ and $\{\mathbf{m}_U^{(1)}, \mathbf{m}_U^{(2)}, \ldots, \mathbf{m}_U^{(N)}\}$, representing the partitioned known and unknown partitioned sets of support meshes, respectively. Let $\mathbf{M}_K$ be the shape matrix formed from the vertices the known partition of each support mesh. Likewise, let $\mathbf{M}_U$ be the shape matrix formed from the vertices of the unknown partition of each support mesh. We have that $\mathbf{M}_K \in \mathbb{R}^{L_K \times N}$ and $\mathbf{M}_U \in \mathbb{R}^{L_U \times N}$, where L_K is the number of known vertices and L_U is the number of unknown vertices. Concatenating $\mathbf{M}_K$ and $\mathbf{M}_U$ yields, $\mathbf{M}$, a permuted version of the original shape matrix. Let $\boldsymbol{\mu}_K$ denote the mean shape vector derived from the known partitions of each support mesh, and let $\boldsymbol{\mu}_U$ denote the mean shape vector derived form the unknown partitions of each support mesh. $\widehat{\mathbf{m}}_K^{(i)} = \mathbf{m}_K^i - \boldsymbol{\mu}_K$ and $\widehat{\mathbf{m}}_U^{(i)} = \mathbf{m}_U^i - \boldsymbol{\mu}_U$ for $i \in \{1, 2, \ldots, N\}$

$$\mathbf{M}_K = \begin{pmatrix} \widehat{\mathbf{m}}_K^{(1)} & \widehat{\mathbf{m}}_K^{(2)} & \cdots & \widehat{\mathbf{m}}_K^{(N)} \end{pmatrix} \tag{23}$$

$$\mathbf{M}_U = \begin{pmatrix} \widehat{\mathbf{m}}_U^{(1)} & \widehat{\mathbf{m}}_U^{(2)} & \cdots & \widehat{\mathbf{m}}_U^{(N)} \end{pmatrix} \tag{24}$$

$$\mathbf{M} = \begin{pmatrix} \mathbf{M}_U \\ \mathbf{M}_K \end{pmatrix} \tag{25}$$

It is required that the patient mesh be registered to the "support" mean, denote this registration transformation as T. Since all mesh topologies are consistent, this may be computed via the Quaternion Method. Assuming that the data follows some multivariate Gaussian distribution, the expected value of the unknown region of the patient mesh is given by (26), where $\mathbf{C}_{UK}$ and $\mathbf{C}_{KK}$ represent the appropriate covariance matrices between U and K.

$$\widehat{\mathbf{m}}_U = T^{-1}\left(\boldsymbol{\mu}_U + \mathbf{C}_{UK}\mathbf{C}_{KK}^{-1}\left(T(\mathbf{m}_K) - \boldsymbol{\mu}_K\right)\right) \tag{26}$$

From (28), we can see that $\mathbf{C}_{UK} = \frac{1}{N-1}\mathbf{M}_U\mathbf{M}_K^T$ and $\mathbf{C}_{KK} = \frac{1}{N-1}\mathbf{M}_K\mathbf{M}_K^T$, and (26) becomes (31).

$$\mathbf{M}\mathbf{M}^T = \begin{pmatrix} \mathbf{M}_U \\ \mathbf{M}_K \end{pmatrix}\begin{pmatrix} \mathbf{M}_U^T & \mathbf{M}_K^T \end{pmatrix} \tag{27}$$

$$= \begin{pmatrix} \mathbf{M}_U\mathbf{M}_U^T & \mathbf{M}_U\mathbf{M}_K^T \\ \mathbf{M}_K\mathbf{M}_U^T & \mathbf{M}_K\mathbf{M}_K^T \end{pmatrix} \tag{28}$$

$$\tag{29}$$

$$\widehat{\mathbf{m}}_U = T^{-1}\left(\boldsymbol{\mu}_U + \frac{1}{N-1}\mathbf{M}_U\mathbf{M}_K^T\left(\frac{1}{N-1}\mathbf{M}_K\mathbf{M}_K^T\right)^{-1}\left(T(\mathbf{m}_K) - \boldsymbol{\mu}_K\right)\right) \tag{30}$$

$$= T^{-1}\left(\boldsymbol{\mu}_U + \mathbf{M}_U\mathbf{M}_K^T\left(\mathbf{M}_K\mathbf{M}_K^T\right)^{-1}\left(T(\mathbf{m}_K) - \boldsymbol{\mu}_K\right)\right) \tag{31}$$

Similar to the computational intractability of the spectral decomposition of the covariance matrix for PCA (see section 3.9), it is not feasible to compute the inverse of $\mathbf{C}_{KK}$, and will only be feasible to store these covariance matrices via computers with a very large amount of system memory (greater than 50 GB). We shall therefore attempt a SVD based factorization trick to obtain this result. Consider the "thin" SVD of $\mathbf{M}_K$ and $\mathbf{M}_U$ shown in (32) and (33).

$$\mathbf{M}_K = \mathbf{U}_K\boldsymbol{\Sigma}_K\mathbf{V}_K^T \tag{32}$$

$$\mathbf{M}_U = \mathbf{U}_U\boldsymbol{\Sigma}_U\mathbf{V}_U^T \tag{33}$$

Substituting (32) into the expression of $\mathbf{C}_{KK}^{-1}$ yields (37).

$$\left(\mathbf{M}_K\mathbf{M}_K^T\right)^{-1} = \left(\mathbf{U}_K\boldsymbol{\Sigma}_K\mathbf{V}_K^T\left(\mathbf{U}_K\boldsymbol{\Sigma}_K\mathbf{V}_K^T\right)^T\right)^{-1} \tag{34}$$

$$= \left(\mathbf{U}_K\boldsymbol{\Sigma}_K\mathbf{V}_K^T\mathbf{V}_K\boldsymbol{\Sigma}_K\mathbf{U}_K^T\right)^{-1} \tag{35}$$

$$= \left(\mathbf{U}_K\boldsymbol{\Sigma}_K^2\mathbf{U}_K^T\right)^{-1} \tag{36}$$

$$\approx \mathbf{U}_K\boldsymbol{\Sigma}_K^{-2}\mathbf{U}_K^T \quad \text{This is } \approx \text{, since it may be that } \mathbf{U}_K\mathbf{U}_K^T \neq \mathbf{I} \tag{37}$$

Substituting (33) into the expression of $\mathbf{C}_{UK}$ yields (39).

$$\mathbf{M}_U\mathbf{M}_K^T = \mathbf{U}_U\mathbf{\Sigma}_U\mathbf{V}_U^T\left(\mathbf{U}_K\mathbf{\Sigma}_K\mathbf{V}_K^T\right)^T \tag{38}$$

$$= \mathbf{U}_U\mathbf{\Sigma}_U\mathbf{V}_U^T\mathbf{V}_K\mathbf{\Sigma}_K\mathbf{U}_K^T \tag{39}$$

Combining (31), (37), and (39) results in (40). (40) may be evaluated as a series of successive matrix-vector products to avoid storing an extremely large matrix in memory.

$$\widehat{\mathbf{m}}_U \approx T^{-1}\left(\boldsymbol{\mu}_U + \mathbf{U}_U\mathbf{\Sigma}_U\boldsymbol{V}_U^T\boldsymbol{V}_K\mathbf{\Sigma}_K^{-1}\mathbf{U}_K^T T\left(\mathbf{m}_K - \boldsymbol{\mu}_K\right)\right) \tag{40}$$

It is worthwhile to examine the intuition behind each matrix multiplication; we shall proceed right to left in (40).

1. $\mathbf{U}_K^T$: Projects the patient mesh onto the SSM of the known region support data; mode weights are obtained
2. $\mathbf{\Sigma}_K^{-1}$: Converts the mode weights to units of standard deviation
3. $\boldsymbol{V}_K$: Transform into the space of covariances between entire instances of the known region
4. $\boldsymbol{V}_U^T$: Project the covariances between entire instances of the known region into covariances between entire instances of the unknown region
5. $\mathbf{\Sigma}_U$: Convert to mode weights of the SSM of the unknown region support data
6. $\mathbf{U}_U$: Combine modes into a valid shape instance for the unknown region

3.13. Bootstrapping

3.13.1. Introduction

In section 3.2, we talk about how to select the template. After selecting template, all the training data are going to aligned to this template by volumetric gray scale deformable registration method. Therefore, the template selection is a important step for whole process. If we use the abnormal template in the beginning, the whole data set are going to register to it. The Atlas created by this template may not be able to describe the instance accurately.

In [8], they use the iterative framework to remove the bias of template and stabilize the Atlas creation. After each registration, they try to create the new template according the mean shape and mean density polynomial. Each training data will align to this new template again. This process will be done several times until error metric converge. They also called this kind of method "BootStrapping".

In [8], they use the tetrahedron mesh to describe the whole volume. Moreover, they also interpolate the density for each tetrahedron. Therefore, the mean density can be

computed by these tetrahedron. In our case, we just use the triangular mesh to describe the appearance of face or skull. There is no density information in these meshes, so we can't modify the step of Bootstrapping method in []. We will discuss more detail in next section.

3.13.2. Bootstrapping method

Let $\mathbf{M}_1, \mathbf{M}_2, \mathbf{M}_3 ... \mathbf{M}_N$ be the mesh of training data set and $\mathbf{M}_T$ be the template that we choose. We can describe the relationship between $\mathbf{M}_i$ and $\mathbf{M}_T$ by the equation below:

$$\mathbf{M}_T = \mathbf{D}_i \circ \mathbf{A}_i \mathbf{M}_i \tag{41}$$

$\mathbf{D}_i$ is the deformation transformation, and $\mathbf{A}_i$ is the affine transformation.In $A \circ B$, $\circ$ is the operation which warps B according the deformation map A.

We assume that the summation of all transformation from each training data to true mean template image will be zero. According this assumption, we warp the old template image according the inverse of average transformation. After warping, the summation of all transformation from each training data to new template image suppose to be zero. The inverse of average transformation can be computed by

$$\mathbf{T}_{avg}^{-1} = \frac{\sum_{i=1}^{N} \mathbf{A}_i^{-1} * \mathbf{D}_i^{-1}}{N} \tag{42}$$

where A_i^{-1} is the inverse of affine transform, and D_i^{-1} is the inverse of deformation map. If A is the affine transformation and B is the deformation transformation, $A * B$ means that transform each displacement vector in B according A.

For our bootstrapping method, we align all the training dataset to template image, compute the average transformation, warp the old template image according the inverse of average transformation, and repeat these steps until the error metric converge. We show the psuedocode of this method in Algorithm 3.

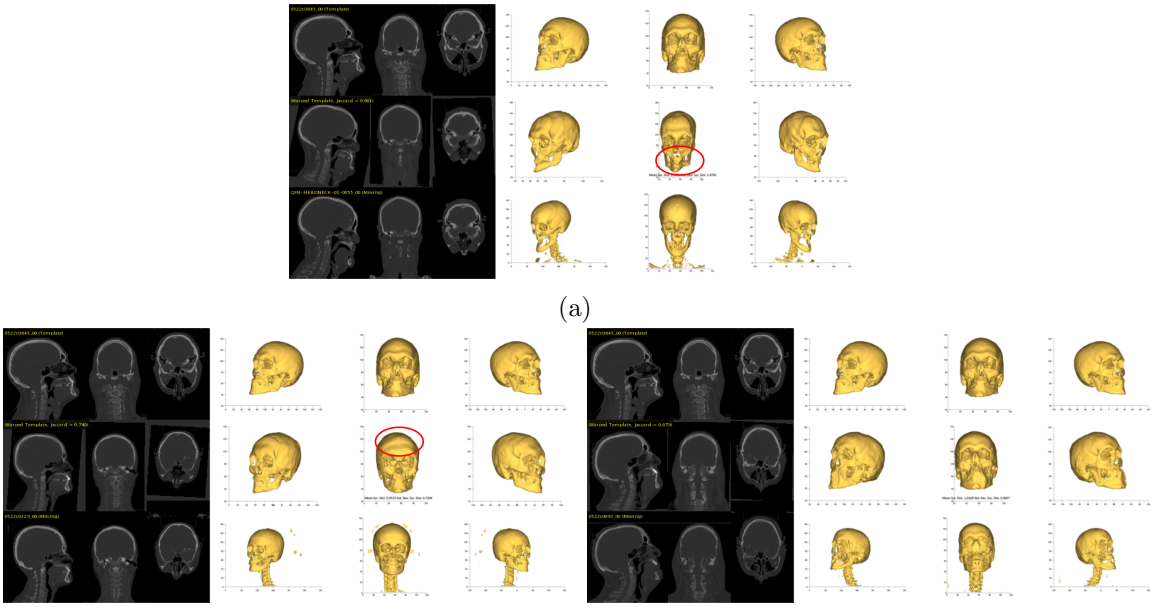
Algorithm 3 BootStrapping

function BootStrapping(Template $\mathbf{M}_T$, Training Data Set $\mathbf{M}_{1...N}$, Number of Iteration n)
  $e \leftarrow \infty$
  for $i = 1, 2, \ldots, n$ **do**
    for $i = 1, 2, \ldots, N$ **do**
      $D_i, A_i \leftarrow$Deformation Registration($\mathbf{M}_T$,$\mathbf{M}_i$)
    end for
    $\mathbf{T}_{avg}^{-1} \leftarrow \frac{\sum_{i=1}^{N} \mathbf{A}_i^{-1} * \mathbf{D}_i^{-1}}{N}$
    $\mathbf{M}_T \leftarrow \mathbf{T}_{avg}^{-1} \circ \mathbf{M}_T$
  end for
  return $\mathbf{M}_T$
end function

4. Results and Discussion

4.1. Deformable Volumetric Registration and Mesh Deformation

The manual evaluation of each registration result is depicted in figure 11. These examples highlight the need for manual inspection of the results, as the Jaccard index failed to perfectly discriminate between successful registrations and failed registrations. Due to the limited amount of training data, it was important to utilize as many successful registrations as possible. 24 successful registrations were generated with the use of the randomized template, and 26 successful registrations were generated with the use of the template chosen via Isomap analysis.



(a)

(b) (c)

Figure 11: Deformable registration and mesh deformation manual inspection. In each figure the left half shows a slice preview in the CT imagery, and the right half shows a mesh preview. The top row shows the template image and mesh, the center row shows the template deformed to the subject (both image and mesh), and the bottom row shows the subject image and mesh generated via an automatic basic thresholding. (a) shows the a registration result that is an obvious failure, due to an extremely deformed mandible. (b) shows an example of a registration failure despite a reasonably good Jaccard statistic. (c) shows an example of success, despite a lower than average Jaccard statistic. Registration errors are highlighted with red ellipses on the warped template mesh. These results are taken from the initial registration using the random template.

4.2. Mesh Alignment

Comparisons of our unaligned, rigidly aligned, and similarly aligned bone and skin training meshes are provided in figures 12 and 13, respectively. The removal of pose is clear when comparing the aligned sets of meshes to the unaligned meshes, however the removal of the scaling component is difficult to discern without "flickering" between full-resolution views of the rigid data set and similarity data set.

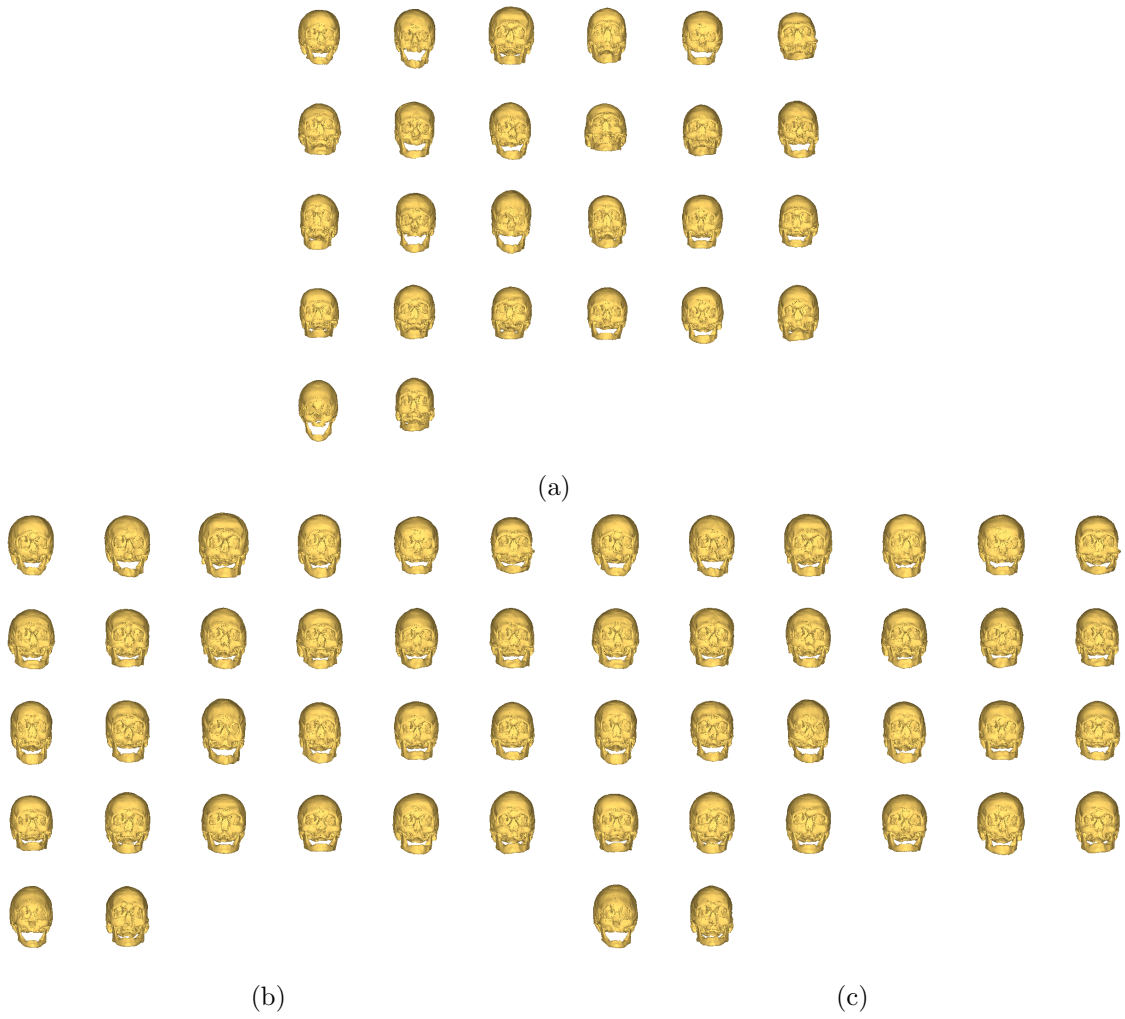


Figure 12: Bone alignment. (a) shows the training data prior to any alignment, (b) shows the training data after a rigid alignment, and (c) shows the training data after a similarity alignment.

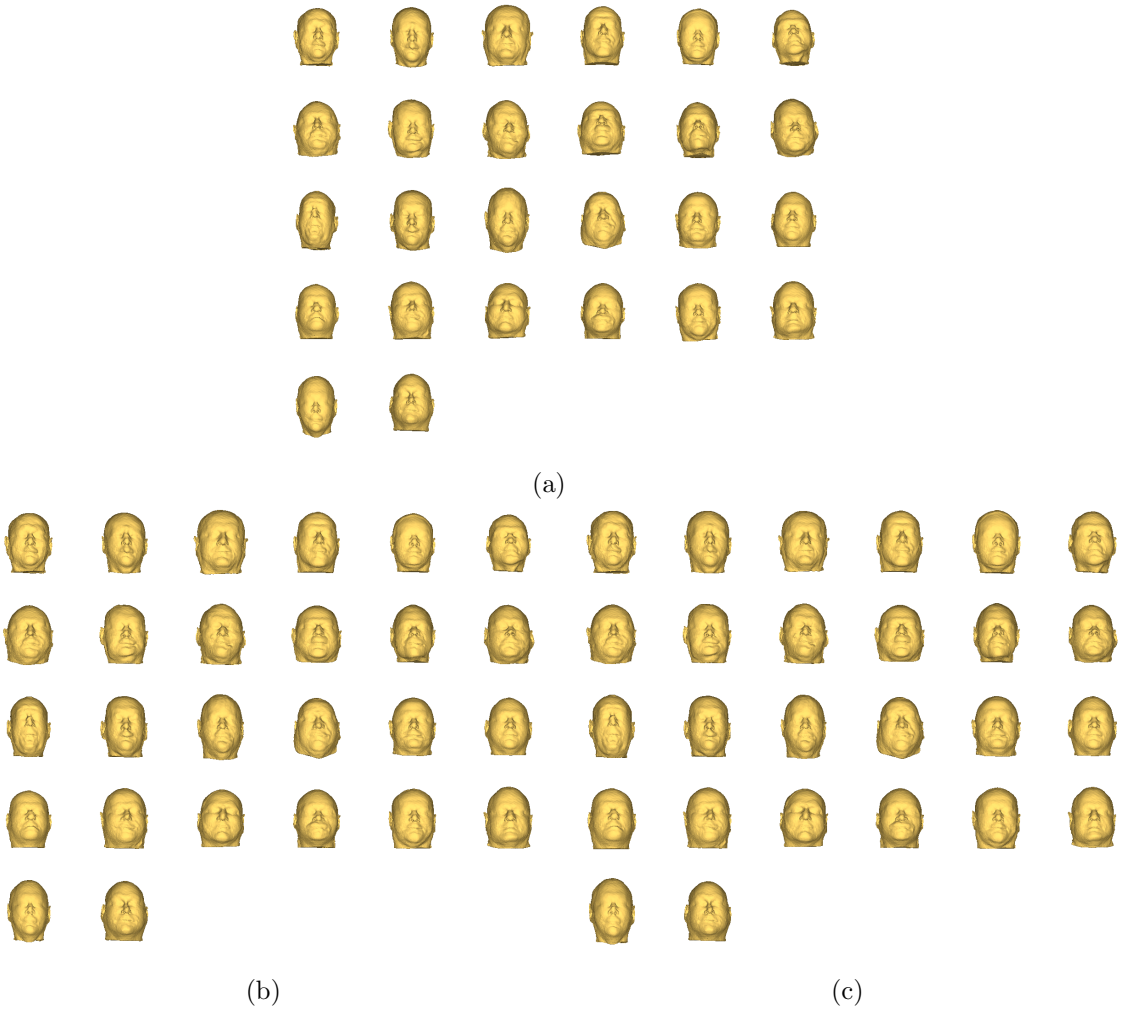


Figure 13: Skin alignment. (a) shows the training data prior to any alignment, (b) shows the training data after a rigid alignment, and (c) shows the training data after a similarity alignment.

4.3. SSM Evaluation

We conducted a series of leave-one-out performance evaluations on each Statistical Shape Model (SSM) created. For each leave-one-out validation, we use three error metrics to compute the error distance: mean surface distance, maximum surface distance and mean vertex error (as defined in Section 3.7). Moreover, the cumulative variance is also shown to express amount of possible shapes represented with a selected number of modes.

We created both bone and skin SSMs for registration results derived from the randomly chosen template and the template chosen with Isomap. Both SSMs were created with rigidly aligned training data. Figure 14 shows a leave-one-out comparison between the two bone SSMs, with the Isomap template derived SSM exhibiting better error performance in surface distance metrics with greater than, or equal to, 7 modes used. Figure 15 shows a leave-one-out comparison between the two skin SSMs. Contrary to the bone comparison, the random template exhibits better error performance than the Isomap template. This could be due to the random template having a skin surface that is closer to the true skin surface mean. Since we are primarily concerned with skeletal structure, any extrapolation will be performed with an SSM derived from the Isomap template.

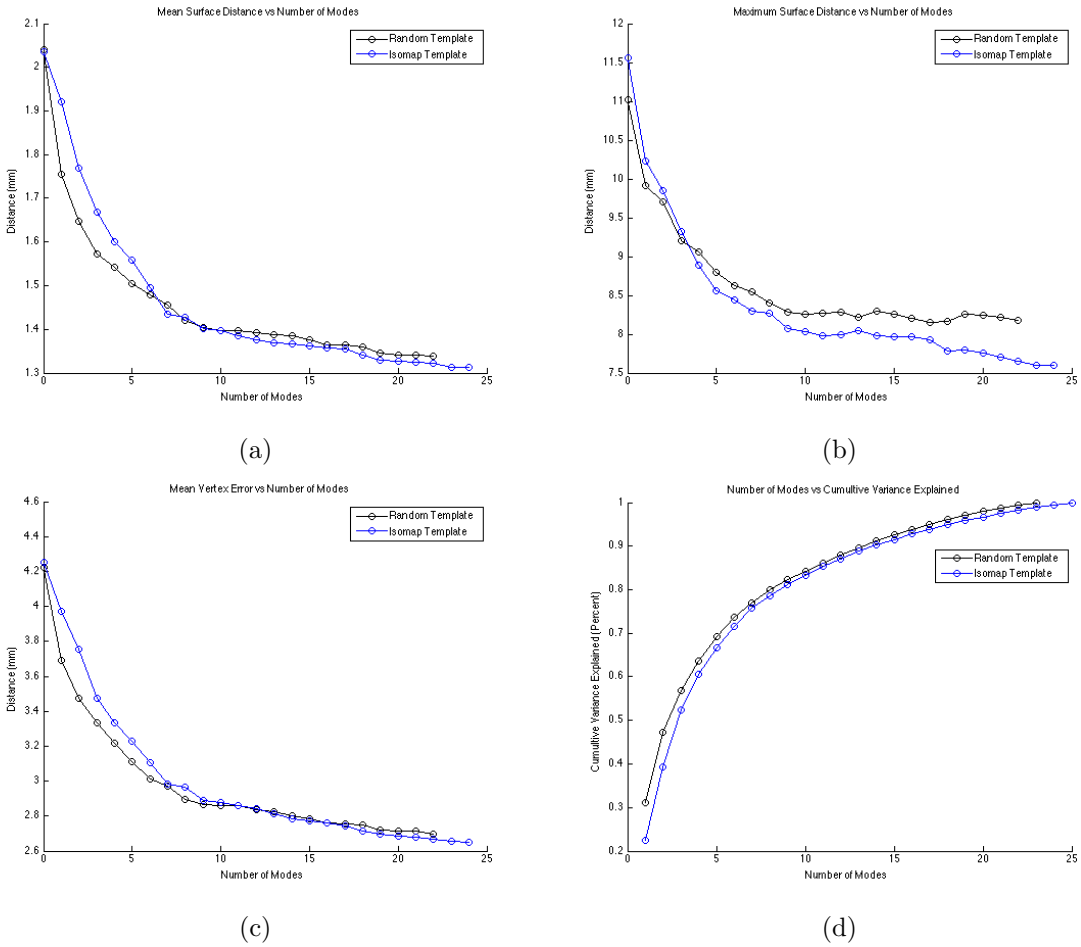


Figure 14: Bone atlas leave-one-out statistics comparing an atlas created with initial random template and an atlas created with an initial template chosen by Isomap. Rigid transformations were used for data alignment and Atlas-to-Patient registration. The left top uses mean surface distance as error metric. The right top uses maximum surface distance. The left bottom uses the mean vertex error. The cumulative variance is shown in the bottom right.

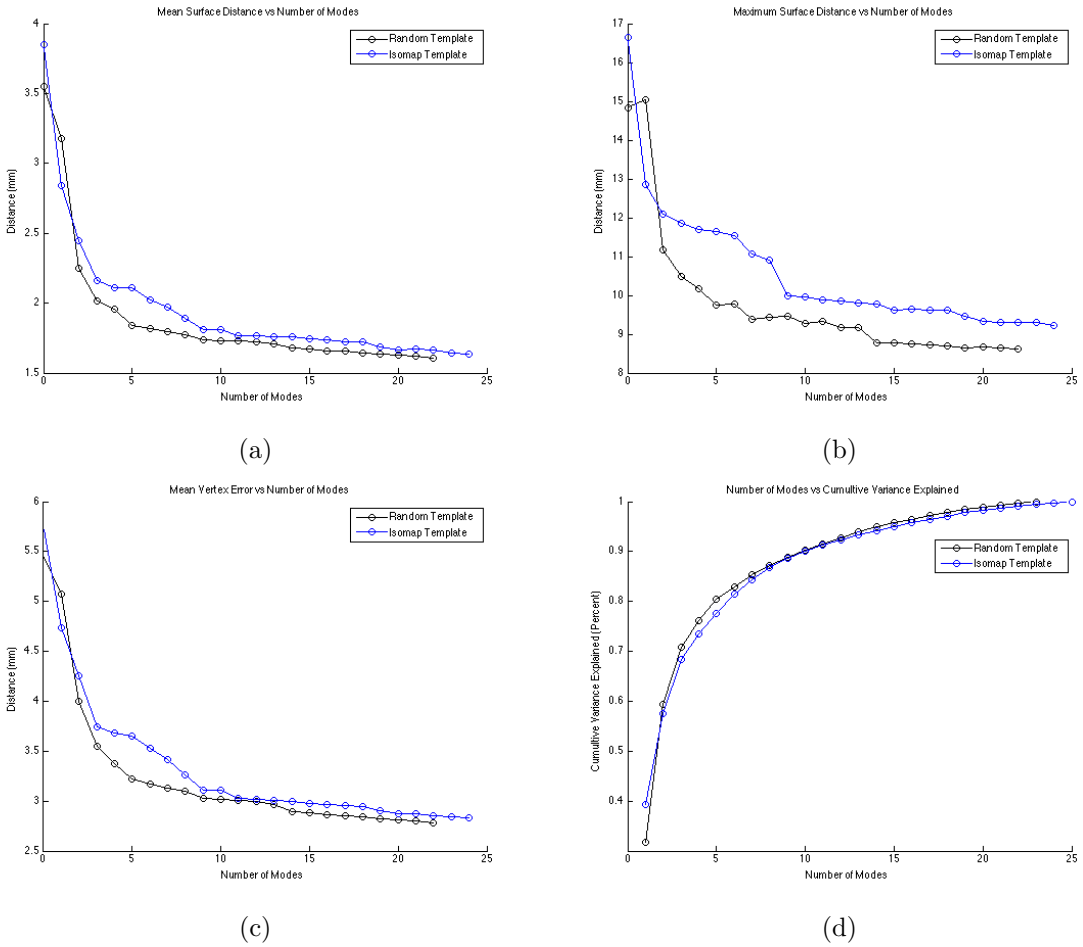


Figure 15: Skin atlas leave-one-out statistics comparing an atlas created with initial random template and an atlas created with an initial template chosen by Isomap. Rigid transformations were used for data alignment and Atlas-to-Patient registration. The left top uses mean surface distance as error metric. The right top uses maximum surface distance. The left bottom uses the mean vertex error. The cumulative variance is shown in the bottom right

We then compared performance between SSMs generated with rigidly aligned training data with SSMs generated with similarly aligned training data. The SSMs for this experiment were derived from the Isomap template. Figure 16 shows leave-one-out performance plots of the bone SSMs and figure 17 shows the leave-one-out performance plots of the skin SSMs. The SSMs generated with a similarly aligned data outperform those generated with rigidly aligned data. This is intuitive as it forces the similarly aligned SSM forces

SSM-to-Patient registration to account for any scale factors, whereas the rigidly aligned SSMs must utilize a mode to account for scale. We shall perform all extrapolation via data that has been similarly aligned.

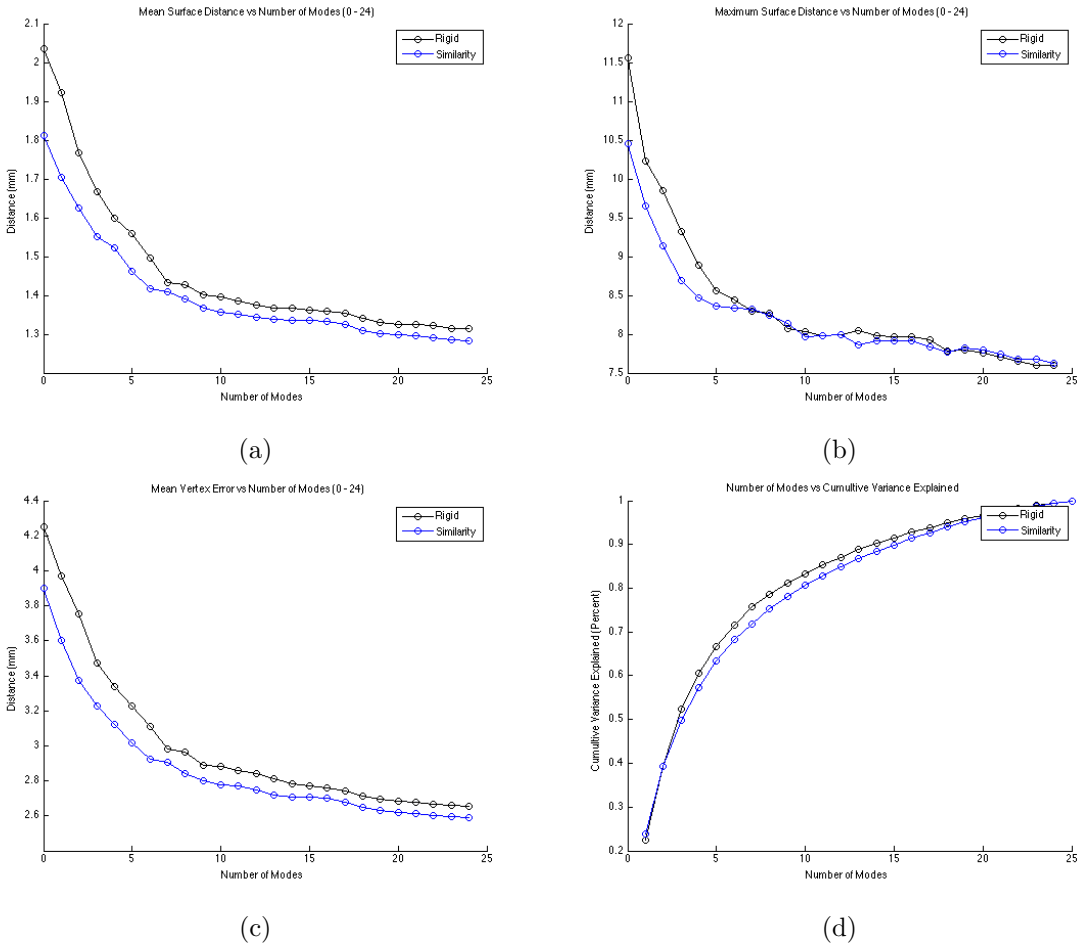


Figure 16: Bone atlas leave-one-out statistics comparing an atlas created with rigid transformations and an atlas created with similarity transformations. The Isomap template was used. The left top uses mean surface distance as error metric. The right top uses maximum surface distance. The left bottom uses the mean vertex error. The cumulative variance is shown in the bottom right

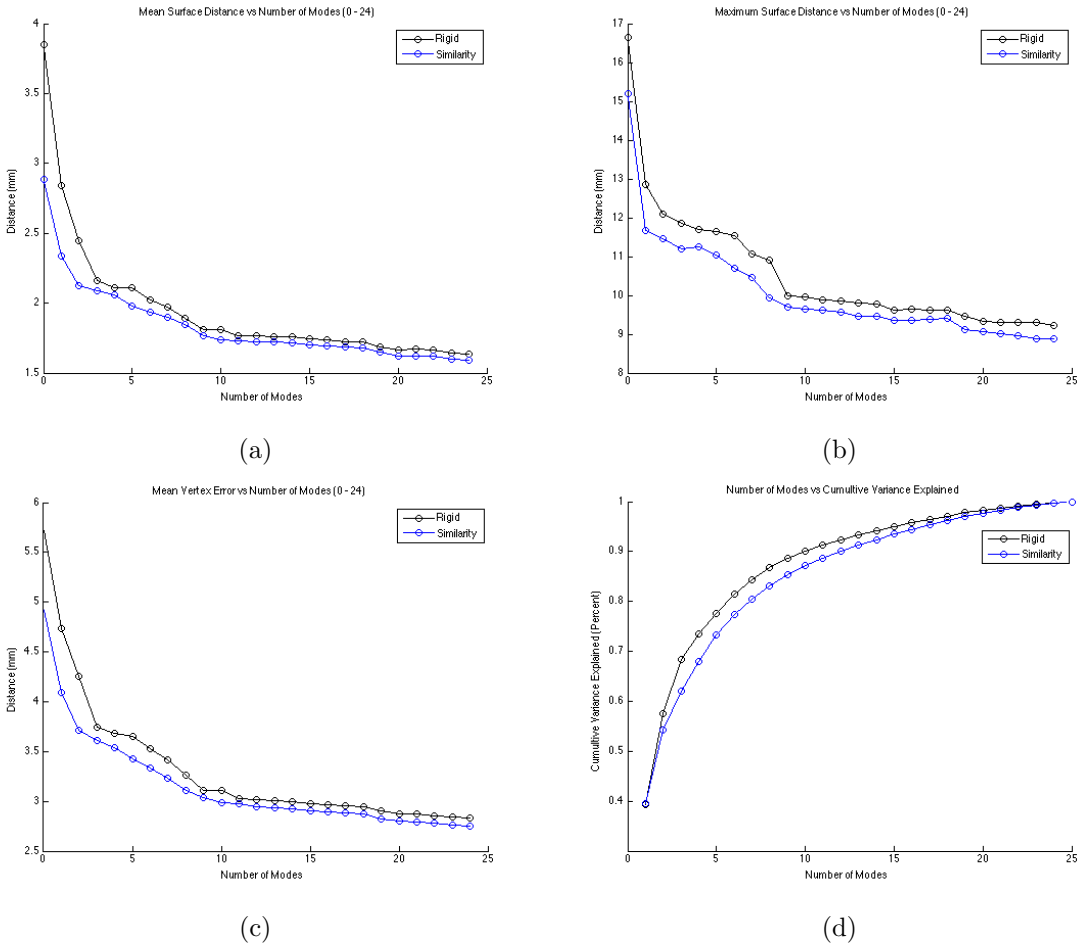


Figure 17: Skin atlas leave-one-out statistics comparing an atlas created with rigid transformations and an atlas created with similarity transformations. The Isomap template was used. The left top uses mean surface distance as error metric. The right top uses maximum surface distance. The left bottom uses the mean vertex error. The cumulative variance is shown in the bottom right

For the bootstrapping, the data sets for validating bootstrapping includes mesh of skin and bone. In the very beginning, we randomly pick the image from training data set as our template. Then, we use the leave-one-out test to validate the result of Bootstrapping in each iteration.

As Figure 18, it shows the validation of bootstrapping for bone. In first iteration, our bootstrapping method does not work very well. We can see in the plot the error of first iteration is similar to error of non-bootstrapping in maximum surface distance and mean

vertex error. However, the mean vertex error and maximum surface distance become smaller in second and third iteration. This is what we expect after we do the bootstrapping.

For maximum surface error and mean vertex error, the distance error is reduced significant when we use less modes for Atlas. When we use the more modes for Atlas, these modes start plays an more important roles in error reduction than bootstrapping.

As Figure 19, it depicts the validation of bootstrapping for skin. For skin data, our bootstrapping method also doesn't work very well in first iteration. The error does not reduce very much in all three error metrics. However, the distance error becomes smaller in second and third iteration, What is more, these plots show the same property that we can see in Figure 18. After using more modes for Atlas, the error reduction of bootstrapping is not significant.

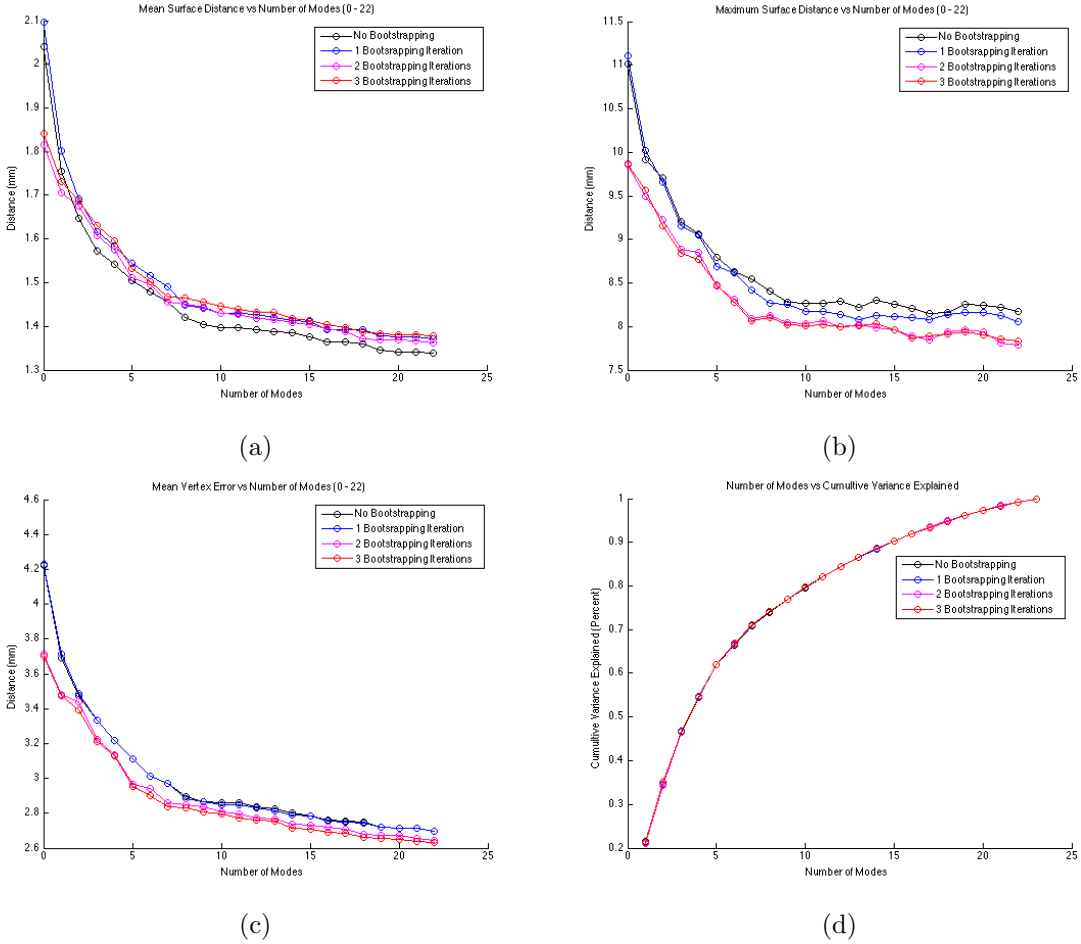


Figure 18: Bone atlas leave-one-out statistics using an initial random template and bootstrapping. The left top uses mean surface distance as error metric. The right top uses maximum surface distance. The left bottom uses the mean vertex error. The cumulative variance is shown in the bottom right

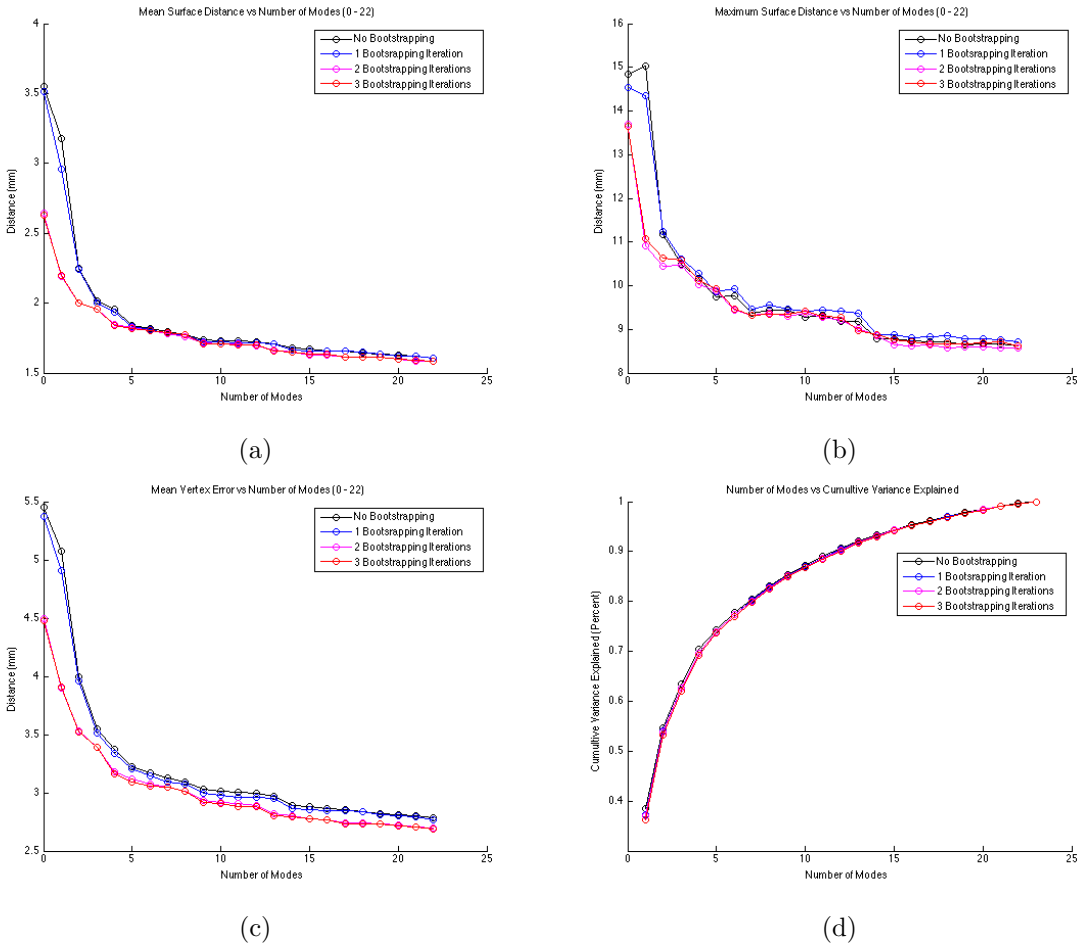


Figure 19: Skin atlas leave-one-out statistics using an initial random template and bootstrapping. The left top uses mean surface distance as error metric. The right top uses maximum surface distance. The left bottom uses the mean vertex error. The cumulative variance is shown in the bottom right

To verify our method, we also try to select the a template that is most similar to every other image as a template via Isomap analysis. In Figure 20, we can see that the maximum surface distance and mean surface distance performance degrades after the first iteration. However, the mean vertex error improves slightly with each bootstrapping iteration, indicating that the registration is more stable. This is most likely a worthwhile tradeoff with the increases in mean surface distance, since those increases are on the order of ≈ 0.1 mm.

We use meshes of skin data set for the same validation as shown in figure 20. The results correlate directly with those seen with the Isomap bootstrapped bone SSM.

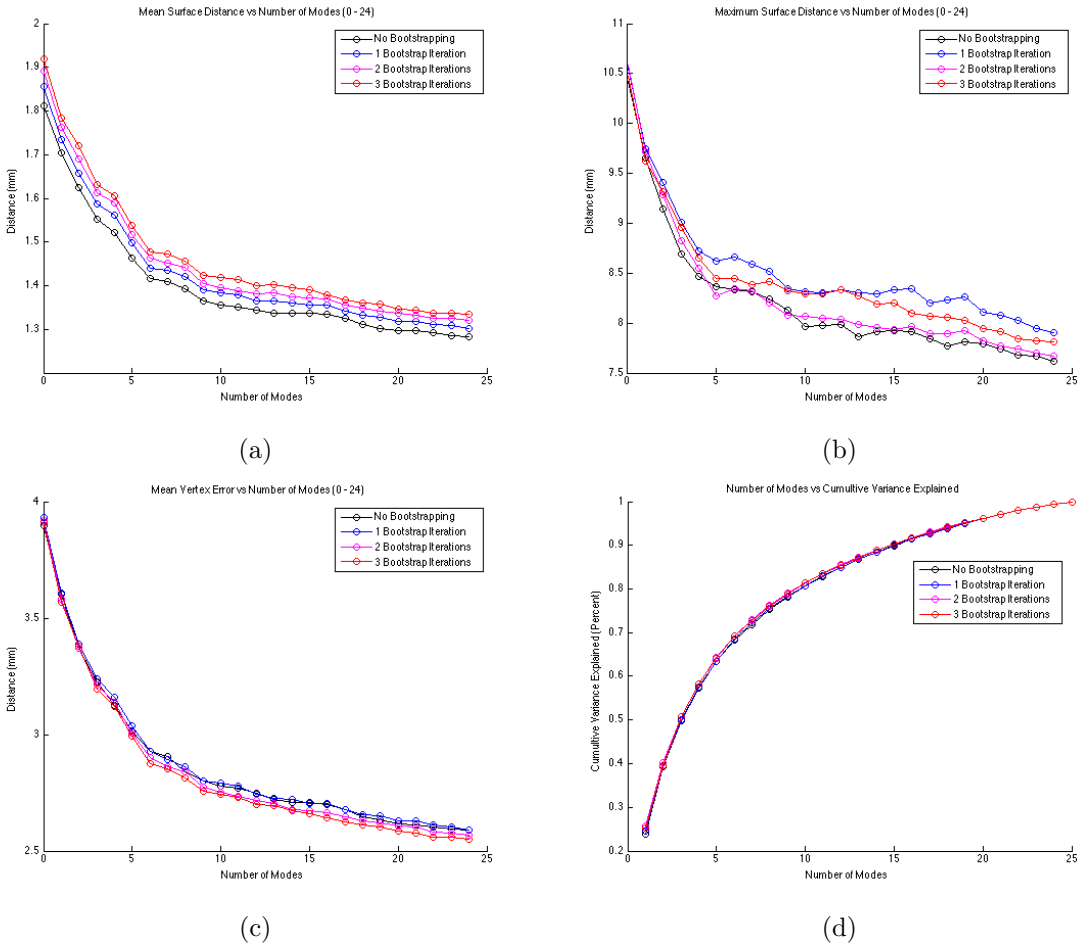


Figure 20: Bone atlas leave-one-out statistics using an initial Isomap template and bootstrapping. The left top uses mean surface distance as error metric. The right top uses maximum surface distance. The left bottom uses the mean vertex error. The cumulative variance is shown in the bottom right.

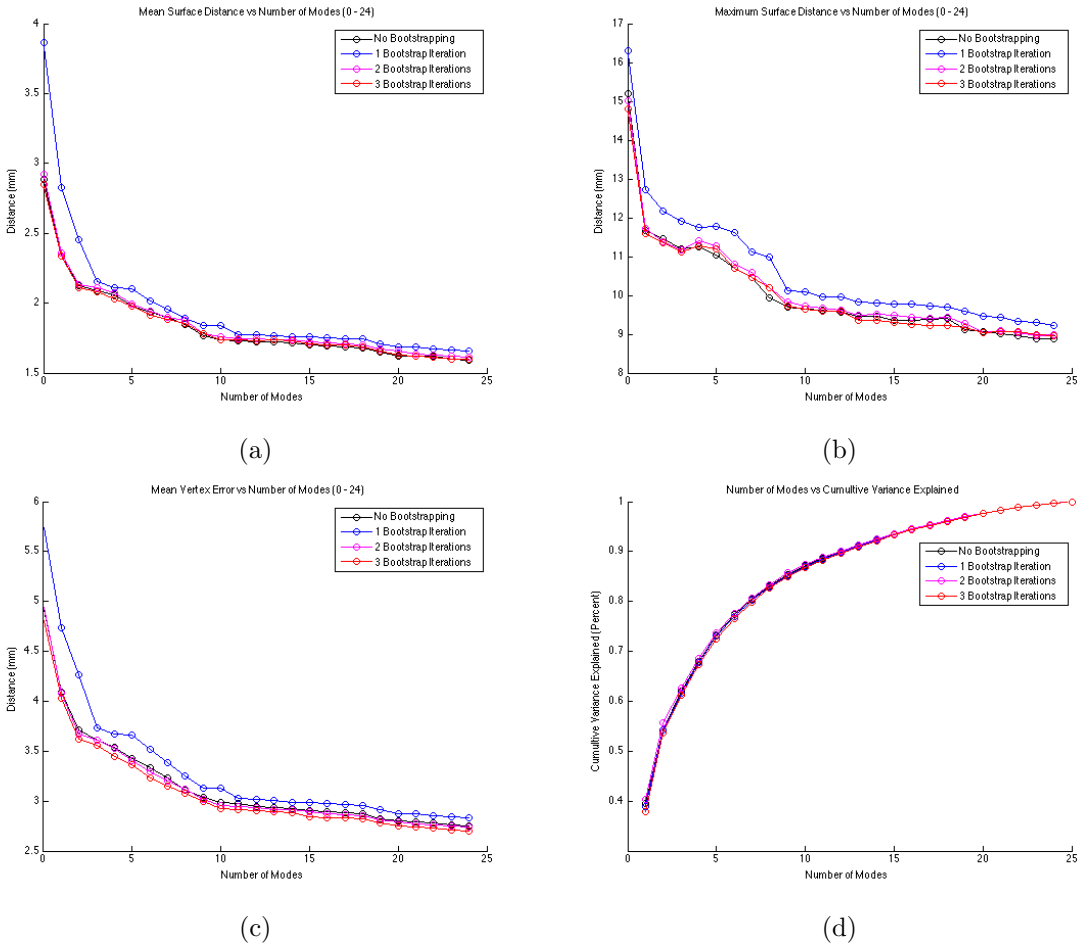


Figure 21: Skin atlas leave-one-out statistics using an initial Isomap template and bootstrapping. The left top uses mean surface distance as error metric. The right top uses maximum surface distance. The left bottom uses the mean vertex error. The cumulative variance is shown in the bottom right

It is interesting to note that the mean surface distances for each bone SSM start at approximately 2 mm with zero modes and then decrease to about 1.3 mm with all modes used. This is unexpected as it indicates sub-voxel error with zero modes used. We believe that this is due to misalignments of the neurocranium having little affect on surface distance. This can be thought of by considering two concentric circles; if one is rotated it has no affect on the minimum surface distance between the two. Figure 22 aids in visualizing the distribution of surface errors as the number of modes used for matching increases. The use of zero modes indicates plentiful regions of large residual error, but also many regions

about the neurocranium with small surface residuals.

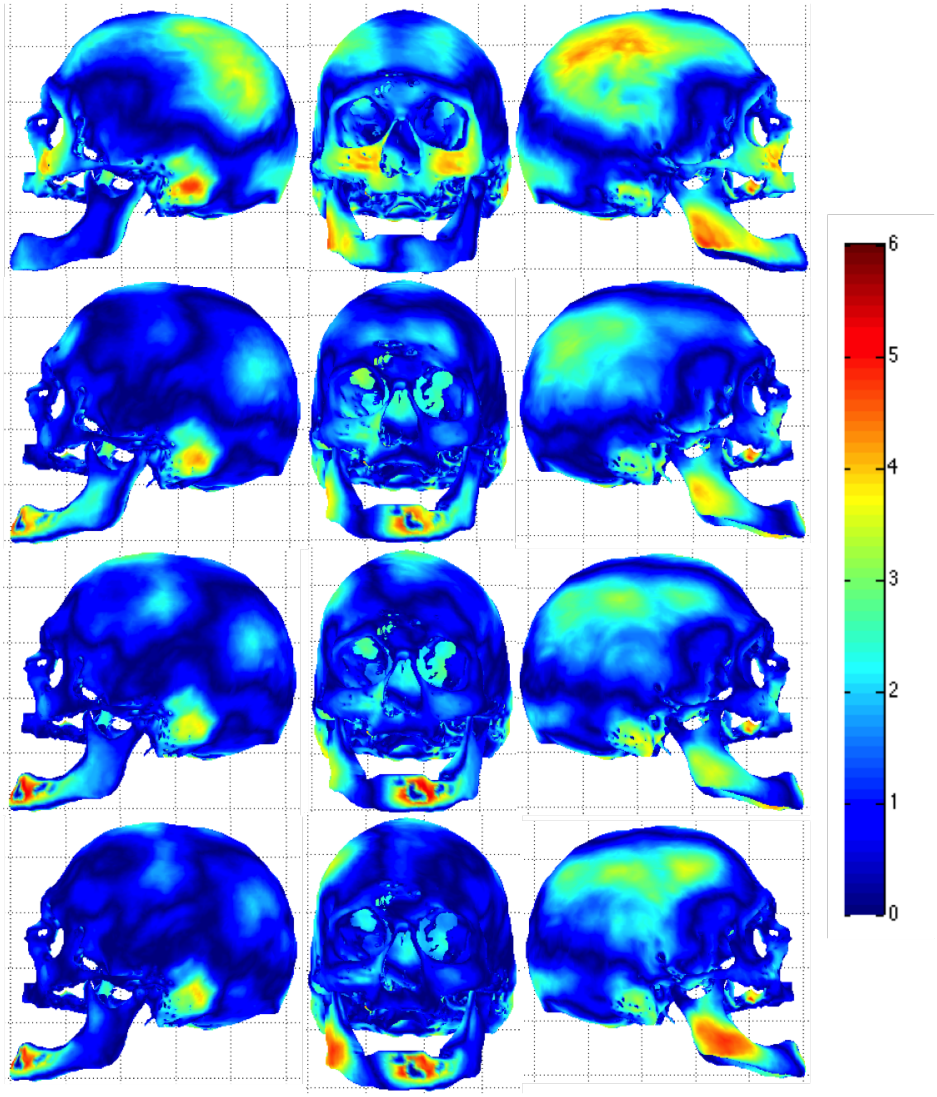


Figure 22: Surface distance error heat maps for a specific leave-one-out SSM. Top Row: Using 0 modes. Second Row: Using 5 modes. Third Row: Using 10 modes. Bottom Row: Using all 24 modes. The left out subject was 0522c0708_00. Distances are in mm.

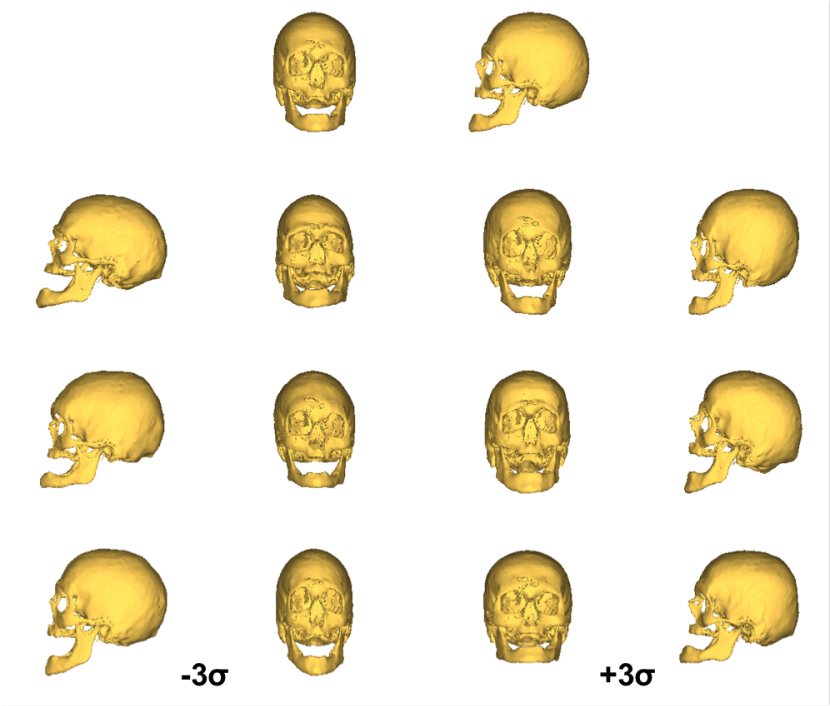


Figure 23: Mean shape, and first three modes of a SSM created from similarly aligned training meshes with initial template chosen via Isomap analysis. No bootstrapping performed.

4.4. Extrapolation

First a simulation of missing data is performed with one of the training meshes, henceforth referred to as the patient. This is analogous the leave-one-out SSM evaluation, in that the SSM is created from the remaining training meshes. Figure 23 shows a symmetric region, about the mid-sagittal plane, that was removed from the patient. The patient, after masking out the removed regions, is registered to the SSM via the ASM based method described in section 3.10.2 and the reconstruction is shown in figure 24. The transition between the known region and unknown region is smooth, however there are non-zero surface errors in the known region which is undesirable for our application. We perform a "copy-and-paste" of the unknown region estimate by the SSM into the original patient mesh to obtain the result in figure 25. This extrapolation will exhibit zero surface error in the known region, however there is a non-smooth transition between the known and unknown regions. In an attempt to overcome this, we apply the multivariate Gaussian regression approach detailed in section 3.12.3; the result is shown in figure 26. Unfortunately, this extrapolation results in a more significant non-smooth transition than the SSM based

"copy-and-paste." One plausible explanation for this is the approximation as a result of equation 37. This could also be due to the greater amount of asymmetry in the mandible region opposed to the known region; if the known region is approximately symmetrical, than any asymmetries will not be transferred to the mode weights of the unknown region. Figure 27 shows a comparison of vertex errors from the true shape between the two reconstruction methods. The SSM based "copy-and-paste" method has lower errors in the transition regions, but larger errors in the extremities.

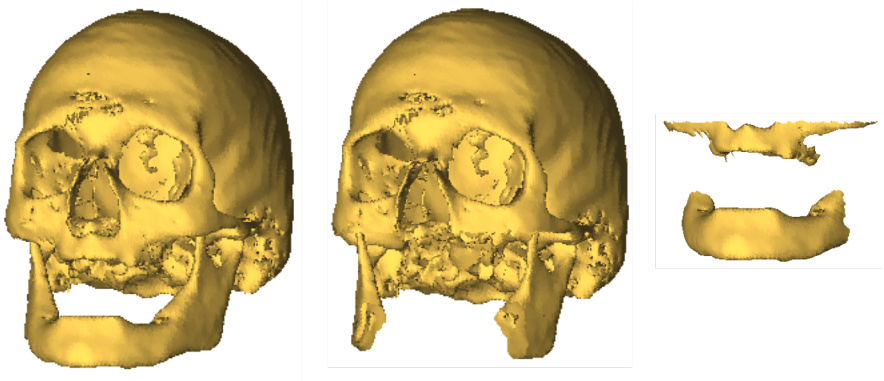


Figure 24: Left: Test subject mesh used for extrapolation testing. Center and Right: a portion of the mandible, maxilla, and zygomatic bones are masked out to simulate an "unknown" region.

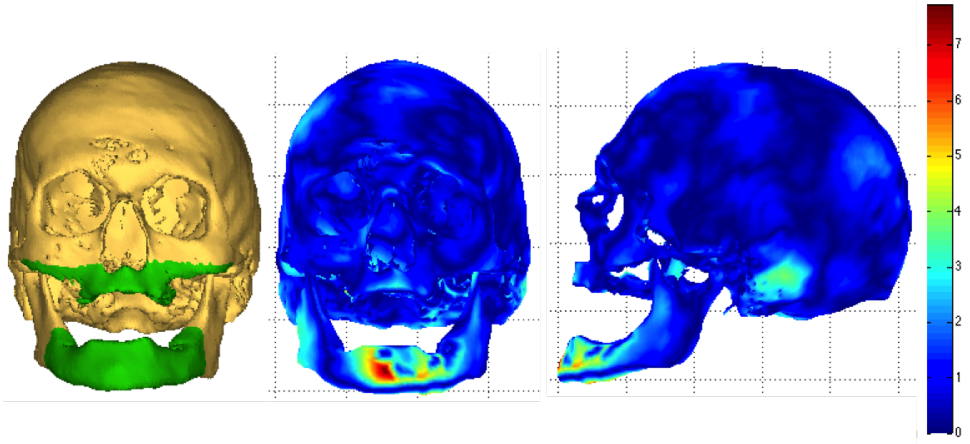


Figure 25: Depiction of the SSM instance created from the test subject. On the left, the extrapolated region is highlighted in green. The two heat maps on the right depict surface distance from the true test subject mesh.

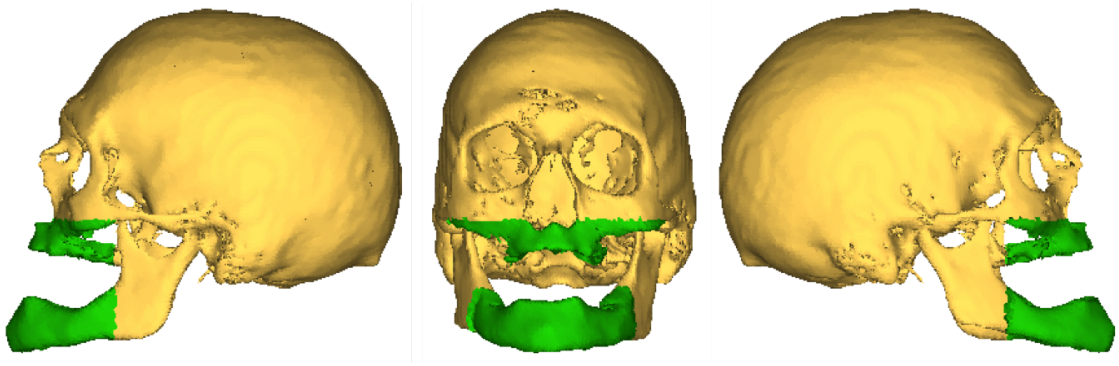


Figure 26: Depiction of the SSM "cut-and-paste" extrapolation created from the test subject; the extrapolated region is highlighted in green. Not the non-smooth transition on the patient's right mandible.

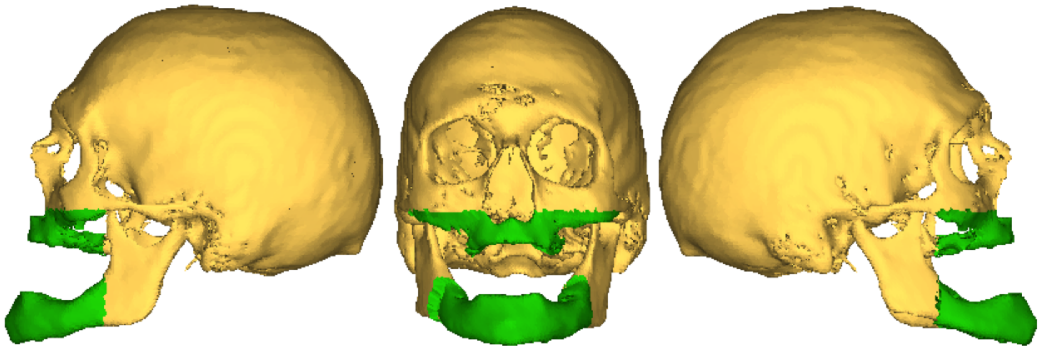


Figure 27: Depiction of the Multivariate Gaussian Regression based extrapolation created from the test subject; the extrapolated region is highlighted in green. Not the non-smooth transition on the patient's right mandible.

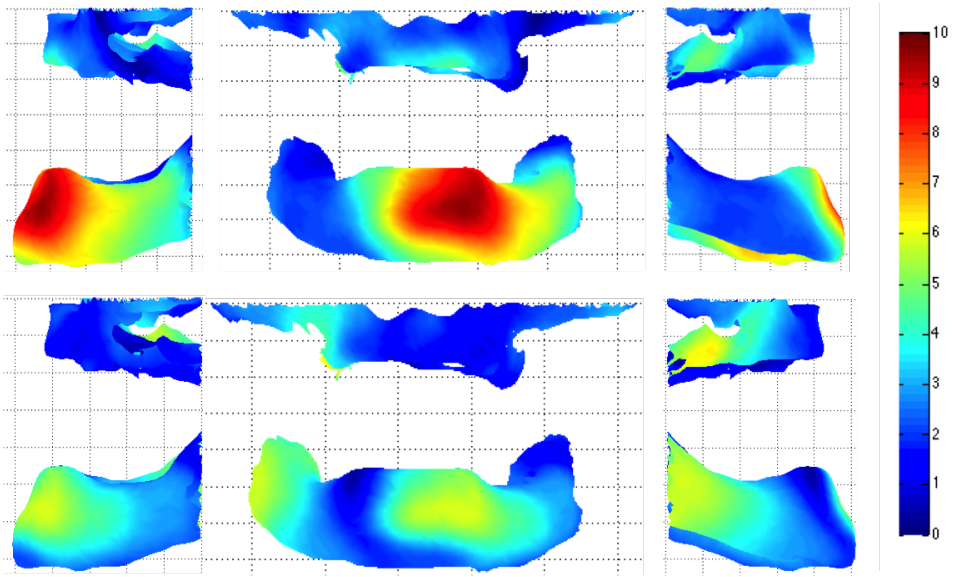


Figure 28: Vertex displacement heat maps of extrapolations from the true patient vertices. Top: SSM based extrapolation, Bottom: Multivariate Gaussian Regression based extrapolation.

We also performed SSM-to-Patient registration to obtain an estimate of what the patient's entire skull should look like. The replacement mandible, nasal bone, and maxilla region were masked prior to registration. The registration was performed via the Active Shape Model method, since this patient has a different topology than our SSM. The SSM was created from training meshes derived from an Isomap template that were similarly aligned; all modes were used. The result is shown in figure 28. The surface error heat map indicates some error about the orbits. The patient had significant reconstruction performed, and these regions should have been masked prior to registration. There is also some error in the "dent" region on the right side of the neurocranium. This region should also have been masked prior to registration, as the patient's nasal bone graft was harvested from this region.

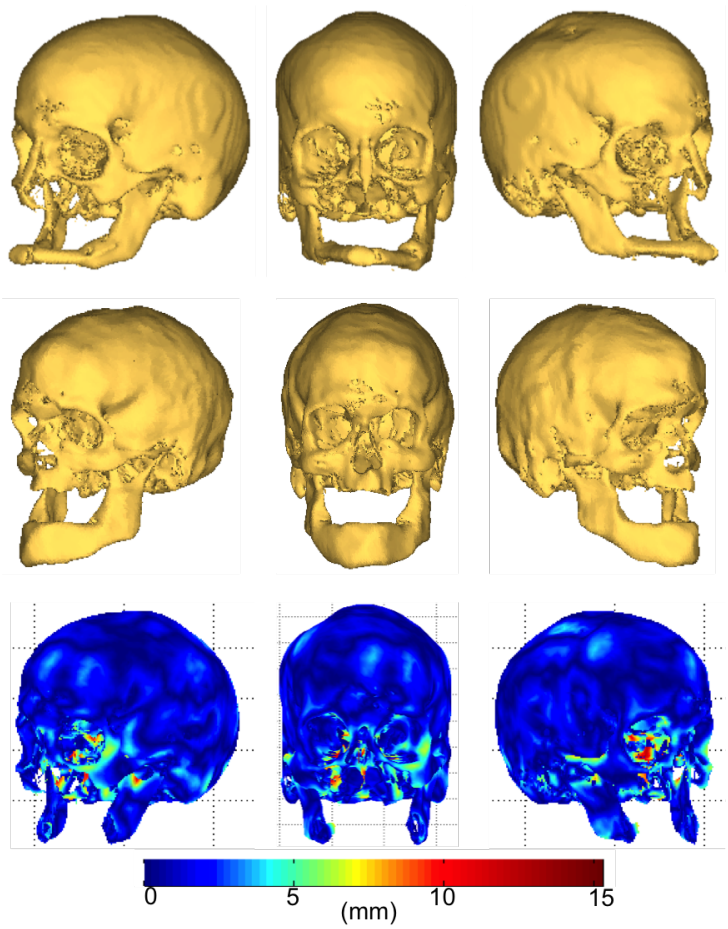


Figure 29: Depiction of a patient with severe facial trauma registered to a skull SSM. Top Row: The current patient skull, regions of transplants and other deformities were masked prior registration. Middle Row: SSM instance to the patient. Bottom Row: Surface distances from the unmasked regions of the patient's skull to the SSM instance.

5. Program Management

5.1. Dependencies

- **Obtaining the Cranial CT Data**
  Dependency met via the use of the TCIA dataset.

- **Access to Mentors**
  Dependency met via recurring weekly meeting with Dr.Otake. Additionally, the re-

curring weekly BIGSS Lab meeting can be used to address issue with other mentors.

- **Access to Fast Computer**
  Dependency met via the use of the new BIGSS Lab server.

5.2. Deliverables

- **Minimum**:(Expected by April 4, 2014)
  - Manual segmentation of the skeletal regions in the cranial CT images. **(Achieved March 16, 2014)**
  - A deformable registration of each CT image, or the surface mesh represented by each CT image, to a chosen template.**(Achieved March 10, 2014)**
  - Creation and evaluation of the cranial CT atlas using the segmented CT images and deformable registration outputs. **(Achieved April 28, 2014)**
  - Creation and evaluation of the method to extrapolate missing skeletal data utilizing the atlas. **(Partially Achieved May 4, 2014)**
- **Expected**:(Expected by April 20, 2014)
  - Creation and evaluation of an atlas via a bootstrapping technique. **(Achieved May 6, 2014)**
  - Development of a realistic patient disfigurement. **(Achieved April 13, 2014)**
- **Maximum**:(Expected by May 5, 2014)
  - Design of a method to use the estimated surface of the patient to assist in surgical planning.
  - Create a system architecture for the future use of this system.

We have achieved the most of deliverables the minimum and expected category except the evaluation an creation of extrapolation. Although we have developed the extrapolation method, it doesn't perform well. There is still "jaggy problem " after extrapolation. We still look for other methods to solve this problem. Maximum deliverables are abandoned for this semester, since Atlas evaluation and updates have taken significant time.

5.3. Schedule

Figure 29 depicts the schedule of our project.

- Green task bars indicate a task completed on, or before, schedule.
- Blue task bars indicate a task that is in not yet started, but not affecting the schedule, or a task that is in progress and on schedule.

- Red task bars indicate a task that is over schedule.
- The red arrows indicate schedule slips.
- A purple diamond indicates an achieved milestone and a yellow diamond indicates a planned milestone.

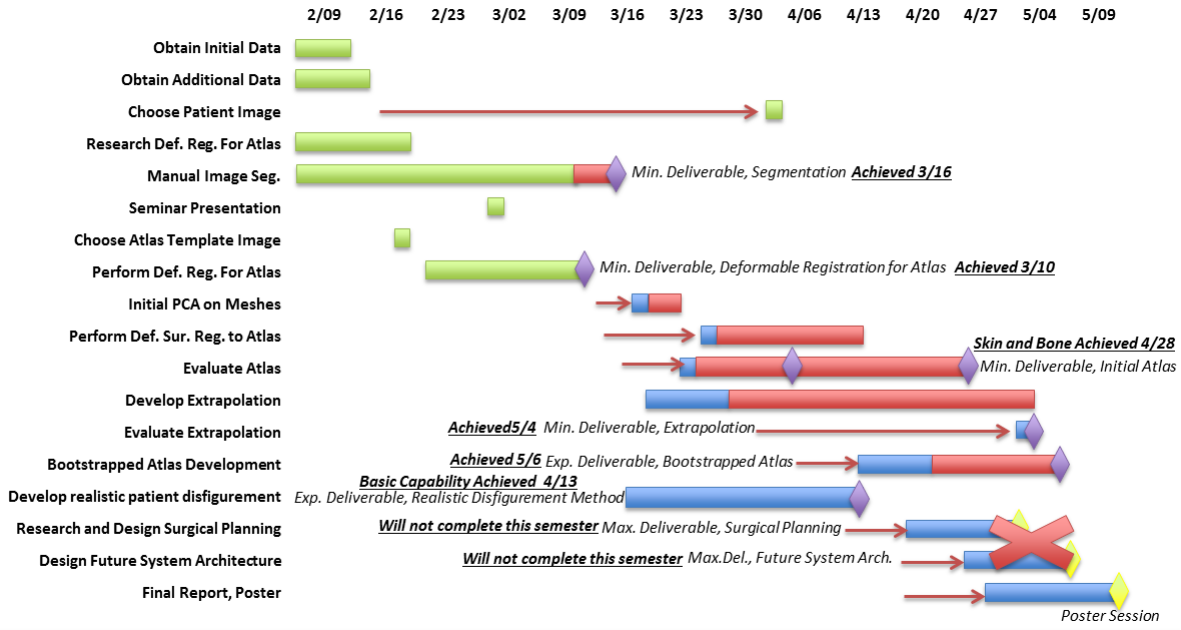


Figure 30: The Gant Chart of Poject Schedule

6. Conclusion

Although we did not complete all of our original requirements, this project served as a valuable introduction to the field of Statistical Shape Models (SSM); specifically their creation and use as an extrapolation mechanism. SSMs for the skull surface and skin surface were successfully created, evaluated, and used for extrapolation. Further analysis and coordination with surgeons will be necessary to determine the performance criteria for an acceptable SSM and extrapolation method. We plan to work with Dr. Gordon to in order to meet our maximum deliverables of surgical planning integration and a total system architecture. We also plan on investigating the following topics:

- Development of a mesh topology transfer routine, so that meshes of arbitrary topology may be extrapolated

- Development of a leave-one-out extrapolation accuracy experiment
- Methods to create a smooth transition into the extrapolated region (such as Thin Plate Splines)
- Find a metric, perhaps Mutual Information, to automatically determine registration quality more reliably than the Jaccard Statistic.
- Evaluation alternative bootstrapping strategies
- Process finer CT resolution (1 mm)
- Better segmentation; separately segmented mandible
- Comparison of several Patient-to-Atlas registration techniques (such using an evolutionary optimization routine; the Covariance Matrix Adaptation Evolutionary Strategy [23])
- Evaluation of incomplete disfigurement knowledge on SSM estimation
- PCA applied to deformation fields

As an additional software task, we hope to further automate the SSM processing pipeline and also design a data model that automatically correlates image and surface data, the statistical models, the performance analysis, and the extrapolation results. The application of computer assistance to face transplant surgery is an exciting and expanding topic [24] [25], and we believe that this project will contribute to the field and positively impact surgical planning.

7. Acknowledgements

We are extremely grateful for the support of our mentors whose zeal for this topic was of great inspiration to us. In particular, the guidance of Dr. Yoshito Otake resulted in infinitely better methods and outcomes than those had we not had access to his expertise. The use of computing resources provided by the Biomechanical and Image Guided Surgical Systems Laboratory, the Laboratory for Computational Sensing and Robotics, the Johns Hopkins Applied Physics Laboratory, and the Johns Hopkins Computer Science department were invaluable when processing the large datasets and computationally intensive processing involved with this study.

Appendices

A. Notation

Let $\mathcal{M}$ denote the set of all valid triangular meshes with vertices in $\mathbb{R}^3$. This appendix will define the mathematical representation of $\mathcal{M}$ used throughout this report. Define the vertex index set for K indices as:

$$\mathcal{I}_K = \{1, 2, \ldots, K\} \subset \mathbb{Z} \tag{43}$$

Define the set of valid faces over $\mathcal{I}_K$ as:

$$\mathcal{F}_K = \{(k_1, k_2, k_3) \in \mathcal{I}_K \times \mathcal{I}_K \times \mathcal{I}_K \mid k_1 \neq k_2, k_2 \neq k_3, k_1 \neq k_3\} \tag{44}$$

A triangular mesh consists of vertices and faces. Given a mesh, $M \in \mathcal{M}$, we may express it as a tuple of vertices and faces: $M = (V, F)$. The first element is a set of vertices:

$$V = \{v_1, v_2, \ldots, v_{N_{V,M}}\} \subset \mathbb{R}^3 \tag{45}$$

The second element is a set of triangle faces:

$$F = \{f_1, f_2, \ldots, f_{N_{F,M}}\} \subseteq \mathcal{F}_{N_{V,M}} \tag{46}$$

$N_{V,M}$ and $N_{F,M}$ will be used to represent the number of vertices and faces, respectively, in the mesh M. For certain mathematical computations it is useful to treat the vertices of a mesh as a single column vector. For $M = (V, F) \in \mathcal{M}$, we shall use the operator $\mathcal{S}(V)$ to define this operation:

$$\mathcal{S}(V) = \left(v_1^{(x)}, v_1^{(y)}, v_1^{(z)}, v_2^{(x)}, v_2^{(y)}, v_2^{(z)}, \ldots, v_{N_{V,M}}^{(x)}, v_{N_{V,M}}^{(y)}, v_{N_{V,M}}^{(z)}\right)^T \tag{47}$$

References

[1] C. R. Gordon, M. Siemionow, F. Papay, L. Pryor, J. Gatherwright, E. Kodish, C. Paradis, K. Coffman, D. Mathes, S. Schneeberger, *et al.*, "The world's experience with facial transplantation: what have we learned thus far?," *Annals of plastic surgery*, vol. 63, no. 5, pp. 572–578, 2009.

[2] M. Siemionow, F. Papay, D. Alam, S. Bernard, R. Djohan, C. Gordon, M. Hendrickson, R. Lohman, B. Eghtesad, K. Coffman, *et al.*, "Near-total human face transplantation for a severely disfigured patient in the usa," *The Lancet*, vol. 374, no. 9685, pp. 203–209, 2009.

[3] C. R. Gordon, S. M. Susarla, Z. S. Peacock, L. B. Kaban, and M. J. Yaremchuk, "Le fort–based maxillofacial transplantation: Current state of the art and a refined technique using orthognathic applications," *Journal of Craniofacial Surgery*, vol. 23, no. 1, pp. 81–87, 2012.

[4] P. Claes, D. Vandermeulen, S. De Greef, G. Willems, J. G. Clement, and P. Suetens, "Computerized craniofacial reconstruction: conceptual framework and review," *Forensic science international*, vol. 201, no. 1, pp. 138–145, 2010.

[5] T. F. Cootes, A. Hill, C. J. Taylor, and J. Haslam, "Use of active shape models for locating structures in medical images," *Image and vision computing*, vol. 12, no. 6, pp. 355–365, 1994.

[6] T. F. Cootes, C. Beeston, G. J. Edwards, and C. J. Taylor, "A unified framework for atlas matching using active appearance models," in *Information Processing in Medical Imaging*, pp. 322–333, Springer, 1999.

[7] M. Fleute and S. Lavallée, "Building a complete surface model from sparse data using statistical shape models: Application to computer assisted knee surgery," in *Medical Image Computing and Computer-Assisted InterventationMICCAI98*, pp. 879–887, Springer, 1998.

[8] G. Chintalapani, *Statistical atlases of bone anatomy and their applications*. Johns Hopkins University, 2010.

[9] University of Iowa, "Quantitative imaging network head-neck." `https://wiki.cancerimagingarchive.net/display/Public/QIN-HeadNeck` [Accessed: 2014-02-14]. The Cancer Imaging Archive.

[10] Radiation Therapy Oncology Group and American College of Radiology Imaging Network, "Head-neck cetuximab." `https://wiki.cancerimagingarchive.net/display/Public/Head-Neck+Cetuximab` [Accessed: 2014-02-14]. The Cancer Imaging Archive.

[11] National Electrical Manufacturers Association and American College of Radiology and others, *Digital imaging and communications in medicine (DICOM)*. National Electrical Manufacturers Association, 1998.

[12] "Neuroimaging informatics technology initiative." `http://nifti.nimh.nih.gov`.

[13] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[14] S. Pieper, M. Halle, and R. Kikinis, "3d slicer," in *Biomedical Imaging: Nano to Macro, 2004. IEEE International Symposium on*, pp. 632–635, IEEE, 2004.

[15] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr, "Implicit fairing of irregular meshes using diffusion and curvature flow," in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pp. 317–324, ACM Press/Addison-Wesley Publishing Co., 1999.

[16] D.-J. Kroon, "Smooth triangulated mesh." `http://www.mathworks.com/matlabcentral/fileexchange/26710-smooth-triangulated-mesh` [Accessed: 2014-02-12]. MATLAB File Exchange.

[17] D. Eberly, "Distance between point and triangle in 3d," *Magic Software, http://www.magic-software. com/Documentation/pt3tri3. pdf*, 1999.

[18] H. Samet, *The Design and Analysis of Spatial Data Structures*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1990.

[19] B. K. Horn, "Closed-form solution of absolute orientation using unit quaternions," *JOSA A*, vol. 4, no. 4, pp. 629–642, 1987.

[20] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Robotics-DL tentative*, pp. 586–606, International Society for Optics and Photonics, 1992.

[21] T. Cootes, C. Taylor, D. Cooper, and J. Graham, "Active shape models-their training and application," *Computer Vision and Image Understanding*, vol. 61, no. 1, pp. 38 – 59, 1995.

[22] R. Grupp and H.-H. Chiang, "Programming assignment 5," December 2013. EN.600.445 Coursework.

[23] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation," in *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*, pp. 312–317, IEEE, 1996.

[24] E. N. Brown, A. H. Dorafshar, B. Bojovic, M. R. Christy, D. E. Borsuk, T. N. Kelley, C. K. Shaffer, and E. D. Rodriguez, "Total face, double jaw, and tongue transplant simulation: a cadaveric study using computer-assisted techniques," *Plastic and reconstructive surgery*, vol. 130, no. 4, pp. 815–823, 2012.

[25] C. R. Gordon, R. J. Murphy, D. Coon, E. Basafa, Y. Otake, M. Al Rakan, E. Rada, S. Susarla, E. Swanson, E. Fishman, *et al.*, "Preliminary development of a workstation for craniomaxillofacial surgical procedures: Introducing a computer-assisted planning and execution system," *Journal of Craniofacial Surgery*, vol. 25, no. 1, pp. 273–283, 2014.