A Method for Finding a 2D Bandeau on a 3D Skull Model

by

Vincent Luong

A paper
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computational Mathematics

Waterloo, Ontario, Canada, 2020

Author's Declaration

I hereby declare that I am the sole author of this paper. This is a true copy of the paper, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

This paper discusses a method to extract a 2D curve representing a bandeau from a 3D mesh of a skull. The bandeau is a strip of bone on the forehead of a skull, and is a critical component of craniosynostosis surgery. We use this bandeau representation for a measure of deformity for patients of craniosynostosis by comparing a patient's 2D bandeau curve with an ideal skull's bandeau curve. There is no current automatic detection of the bandeau, and it is not easily identifiable for those unfamiliar with craniosynostosis. Our method relies on identifying specific landmarks on the skull. We start with getting initial positions for the landmarks using point set registration techniques with a separate reference skull, then use the anatomical properties of each landmark to improve on these initial guesses. Typically, the method is effective if given a decent initial guess with the registration. Thus, initial transformations of the reference skull help improve the performance, even if they are simple. We test this method on infant skulls, both those diagnosed with craniosynostosis and those that are not. We find that the algorithm performs well when including several manual steps.

Acknowledgements

I would like to thank everyone in the SickKids research group for inviting me to work on such an interesting project. In alphabetical order, thank you to Chris, Dr.John Phillips, Dr.Jochen Koenemann, Justin, Marina, Mathieu, Dr.Ricardo Fukasawa (and Sheridan!). Thank you all for your guidance, you all made the transition into an area that I'm unfamiliar with easier.

Thanks to my colleagues in the Computational Math program, for all the long, but enjoyable working sessions at the CM lab.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Craniosynostosis is a birth defect that causes an infant's skull to shape abnormally. If left unattended, there will not be enough space for the brain to grow in the skull. During craniosynostosis surgery, parts of the skull are removed and reshaped, and one of the key parts of the skull that is reshaped is the bandeau [5][9]. The bandeau, or more specifically, the fronto-orbital bandeau, is a strip of bone that reaches slightly above the orbit, and has a height of about 2cm [3].

Currently, there are no known methods for automatically detecting the bandeau. Instead, the current methods to locate the bandeau are to manually find the bandeau by manually locating specific landmarks on the skull and hand-picking the location of the bandeau based on the landmarks. Thus, a lot of expertise is required to properly locate the bandeau.

In this paper, we propose a method to find a 2D slice of the bandeau. This 2D slice is in the middle of the bandeau strip. We find this 2D slice because it is a simplified version of the whole bandeau. Since the bandeau is not a full strip of bone due to the holes on the eye sockets, it can be difficult to compare the bottom portions of two different bandeaus (Figure 2.2). Instead, we compare the 2D slices of the bandeaus that can be plotted on a plane, and use metrics such as area between the curves to compare deformity. By comparing a 2D slice of the bandeau of a deformed skull and one of a normal skull, our comparisons can be used as a measure of deformity. Furthermore, there is work being done to optimally reconstruct the bandeau in the 2D and 3D setting. A method for finding a 2D slice of the bandeau will be used as a complementary aid for the bandeau reconstruction research.

The method proposed is centered around finding key landmarks on the skull. With a reference skull with known landmark locations, we use point set registration techniques to match the reference skull to an unknown target skull. These registration techniques transform a source point set to resemble the shape of a target point set. This gives initial guesses to the landmarks, which are then iteratively improved using techniques based on the properties of the location of each landmark. We use the landmarks to find the Frankfurt Plane [14], then move the Frankfurt plane to intersect another landmark. We then find a 2D slice of the bandeau by using the shifted Frankfurt plane.

From testing, a good registration is key, but can also be tricky due to the differences in shape between two different skulls. It is even more difficult because we are working with skulls with craniosynostosis. The registration results may be heavily affected by what skull is used as reference. Thus, a lot of work can still be done with improving general skull registration.

The paper is organized by the following. Chapter 2 covers preliminaries and background work. Chapter 3 explains the algorithm for identifying the 2D bandeau and each of its parts. Chapter 4 goes over tests results done on various skull meshes. Chapter 5 covers possible future work to improve the algorithm.

Chapter 2

Preliminaries and the Problem Description

Before we formally define parts of the skull, we first define the directions used to describe the skull. *Anterior* describes the direction from one's heels to one's toes — the front-facing direction. The opposite direction is *posterior*. *Superior* describes the direction from one's feet to one's head. The opposite direction is *inferior*. And finally, *left-lateral* describes the left, and *right-lateral* follows suit.

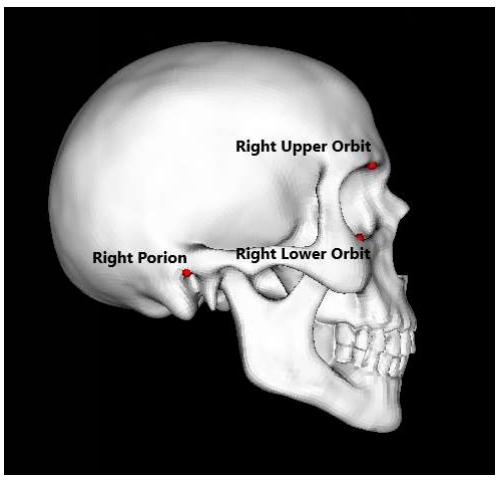
Notation: Throughout this paper, we will denote the unit vector in the right-lateral direction, the superior direction, and the anterior direction to be u_x, u_y, u_z respectively.

2.1 Skull Landmarks and the Bandeau

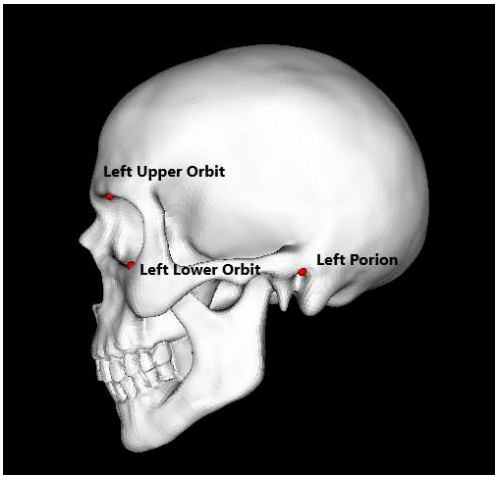
Before we define the bandeau, we must first define the landmarks on the skull that are incorporated in its definition.

We define the lower orbits of a skull to be the most inferior point on the rim of each eye socket. The left lower orbit is the most inferior point on the rim of the left eye socket, and the right lower orbit is the same for the right eye socket. Similarly to the lower orbits, the left and right upper orbit are the most superior point on the rim of the left and right eye socket respectively. Figure 2.1c shows examples of the upper and lower orbits of a skull.

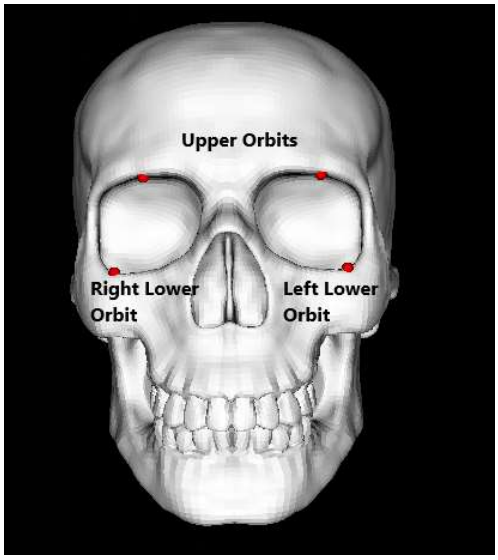
The left porion of a skull is the point in the most left-lateral direction that is on the roof of the left bony earhole. Similarly, the right porion is the point in the most right-lateral direction that is on the roof of right bony earhole. The definitions of the 6 landmarks are medical definitions, and are not well-defined mathematically. See figure 2.1b and 2.1a for examples of the porion locations of a skull.



(a) Patient-right view of a skull.



(b) Patient-left view of a skull.



(c) Front view of a skull

Figure 2.1: The orbits and porions of a skull mesh from TurboSquid. [12]

Definition 2.1.1 (Frankfurt Plane) *The Frankfurt plane of a skull is defined to be the plane that passes through the lower orbits and the porions.*

Although we define the plane with four points, due to the symmetry of skulls, we expect the plane defined by the porions and one lower orbit to also intersect the second lower orbit.

Definition 2.1.2 (Fronto-Orbital Bandeau) *First, we define the nasion as the most anterior point on the nasal bone. The fronto-orbital bandeau is defined to be the strip of bone that starts at 5mm above the nasion, and stretches upward until 15mm above the upper orbits. The bandeau is parallel to the Frankfurt plane, and stretches toward the back of the head until it reaches above the left/right porion.*

See figure 2.2 for an example of the location of the bandeau of a skull.

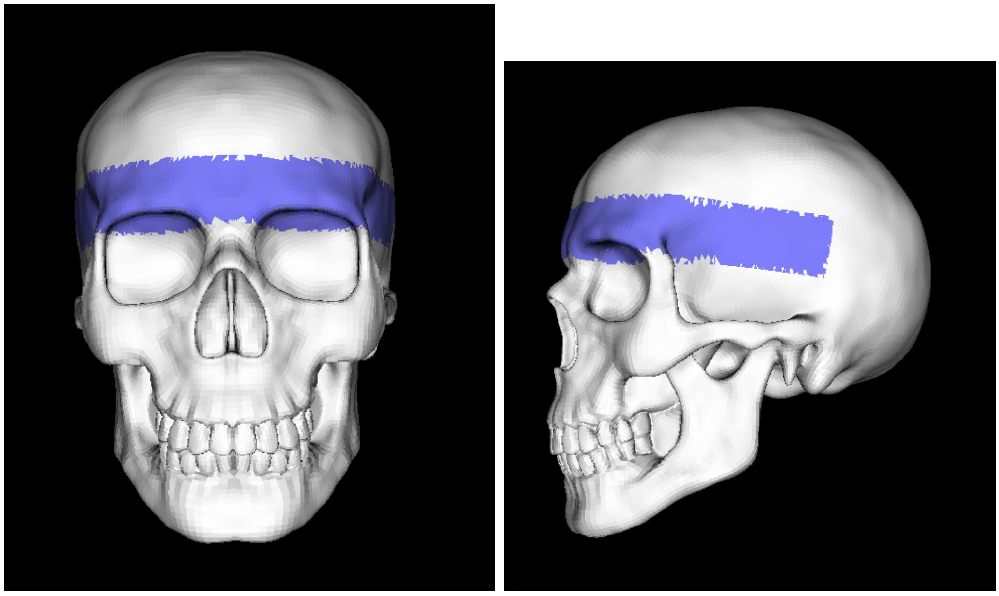


Figure 2.2: The fronto-orbital bandeau of a skull in blue. [12]

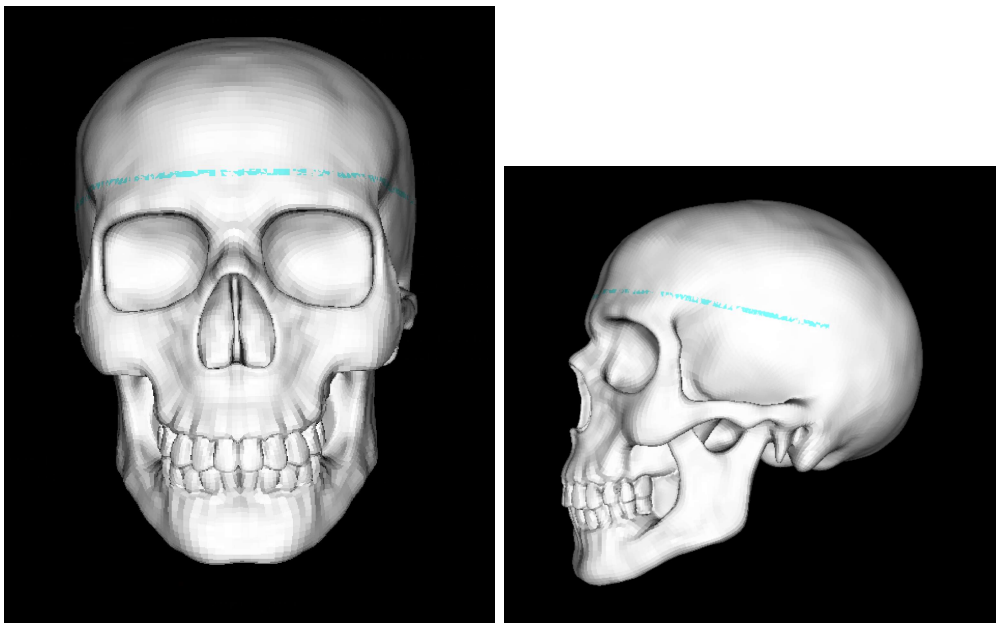
2.2 Goal of the Algorithm

We look to identify the bandeau curve of a skull, a 2D representation of the 3D bandeau. To get this 2D curve, our algorithm will look to find the curve on the exterior of the skull that is 5mm above the upper orbits, is parallel to the Frankfurt plane, and stretches toward the back of the head until it reaches above the left/right porion in the superior direction.

Definition 2.2.1 (Bandeau Plane) *The bandeau plane of a skull is the plane on which the bandeau curve lies. The bandeau plane is parallel to the Frankfurt plane and intersects the points 5mm above the upper orbits.*

Definition 2.2.2 (Bandeau Curve) *The bandeau curve of a skull is the intersection of the exterior skull with the bandeau plane. It stretches to the back of the skull until it reaches above the left/right porion.*

See figure 2.3 for an example of the location of the bandeau curve of a skull.



(a) Patient-right view of a skull. (b) Patient-left view of a skull.

Figure 2.3: The bandeau curve of a skull in cyan. [12]

Thus, the goal of the algorithm will be to first locate the upper/lower orbits and the porions. This is an important part of the algorithm where we try to locate landmarks that are not well-defined mathematically. Then, once we find those landmarks, we will find the Frankfurt plane, the bandeau plane, and then the bandeau curve.

2.3 Background Work

2.3.1 Point Set Registration

Registration is the process of matching two point sets together by transforming one point set to resemble the second point set as closely as possible. We will call the point set being transformed as the source point set, and the target point set is the point set we want to match. During this process, each point on the source is paired with a different point on the target. Next, the source point set is transformed (with linear transformations and non-linear transformations) to minimize the summed Euclidean distances between each pair of points. Different registration algorithms change the possible transformation types and how points are matched between the two point sets. The same transformation is applied to all source points, so that you do not end up transforming each source point individually to trivially match its paired target point.

Let $S, T \subset \mathbb{R}^3, T = \{t_1, t_2, ..., t_n\}, S = \{s_1, s_2, ..., s_n\}$. And let $d : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}$ be the Euclidean distance of two points. Then a registration technique looks like the following:

Algorithm 1 A cookie-cutter registration technique

Input: S, T, two sets of points, threshold $\in \mathbb{Z}$, max iterations $\in \mathbb{Z}$
Output: S', a transformed set of points created by transforming S

1: iterations = 0
2: **while** iterations < max iterations and $\sum d(s_i, t_i) <$ threshold **do**
3: Re-index the points in S to create a new matching
4: Transform S in some way (ie. rotation, translation).
5: iterations = iterations + 1
6: **end while**
7: $S' = S$

We will be using registration algorithms to match a reference skull with known landmarks to a target skull with unknown landmarks. On the transformed registered reference

skull, we will use the transformed landmark positions on the registered skull as the initial guess for the landmark positions on the target skull.

Example Registration Algorithms

Two registration algorithms that will be mentioned are Fractional Iterative Closest Point (FICP) [6] and Coherent Point Drift (CPD) [4]. FICP is variant of ICP [1] that matches only a subset of the points in each of the target and source point sets. The points in the source point set are matched to their closest neighbour in the target point set, and only rigid transformations (translation, reflection, rotation) are used in ICP. ICP is more well known classical algorithm for point set registration.

CPD is a registration algorithm that allows for non-rigid transformations on the source point set. CPD can also match point sets of different cardinality. We will not go into much detail due the complexity of CPD, but it treats aligning the two point sets as a probability density estimation problem. One point set represents the centroids of a Gaussian Mixture model, while the other represents the data points. In our bandeau identification algorithm, we will use CPD for registration because the non-rigid transformations better register the wide variety of shapes in deformed skulls. Different skull meshes typically contain a different number a points, and CPD can accommodate two skull meshes without taking a sample of the points. However, due to the computational complexity of CPD, our implementation of the bandeau curve algorithm will take a sample of skull meshes.

2.3.2 Frankfurt Plane Detection

To find the landmarks that define the Frankfurt plane, we use a slightly modified version of the algorithm by Cheung, Leow, and Lim [14]. In [14], the algorithm's input is two skull meshes — a reference skull with known landmark locations and a target skull with unknown landmark locations. The landmarks of interest are the lower orbits (referred to as orbitales in [14]), and the porions. We will give a basic overview of the algorithm in [14] below, and explain each step in more detailed afterwards.

1. Register the reference skull to match the target skull. On the registered reference skull, the transformed landmarks will be the initial guesses for the landmarks on the target skull.

2. For each landmark, repeat until convergence:

   (a) Select a region around the landmark as the set of next possible guesses

   (b) Select a new guess for the landmark based on the anatomical properties of the landmark

3. Use Principal Component Analysis (PCA) [13] to find the plane of best fit between the four landmark points. The plane of best fit is the plane that minimizes the distance from each point to the plane.

In step 1, the algorithm uses the registration algorithm Fractional Iterative Closest Point (FICP) [7] with a reference skull with known landmark locations to find initial guesses for the landmarks on a particular target skull.

In step 2(a), this algorithm considers an elliptical region around the landmark and its intersection with the skull. We call this intersection the landmark region. Then in step 2(b), we pick a specific point in the landmark region, depending on landmark we are trying to find.

Let p_r be the initial guess for the right porion, and let R_{p_r} represent the landmark region corresponding to the right porion. Then, we take S_{p_r}, a subset of the points in R_{p_r} where the surface normals are close to the inferior direction $-u_y$ to find the points on the roof of the bony earhole. Since the right porion is the point in the most right-lateral direction, the next landmark becomes the following:

$$p_r' = \arg\max_{p \in S_{p_r}} p \cdot u_x$$

We assume we know the orientation of the skulls, and so u_x, u_y, u_z are known.

Similarly, let p_l be the initial guess for the left porion, and let R_{p_l} denote the landmark region corresponding to the left porion, and define S_{p_l} as a subset of the points in R_{p_l} where the surface normals are close to $-u_y$. Then the point selected as the next guess for the left porion is the following:

$$p_l' = \arg\max_{p \in S_{p_l}} p \cdot -u_x$$

Let l_l, l_r be the initial guess for the left lower orbit and the right lower orbit. The lower orbits are defined as the lowest point on the rim of each eye socket. Let R_{l_l} be a landmark

region corresponding to the left lower orbit. We slice R_{l_l} in its sagittal sections, that is, we take slices of the landmark region in the vertical/superior direction. Formally, we define a sagittal section $S_{l_l}(x)$ as the points in R_{l_l} with value x in the u_x direction. Then, let $q_{l_l}(x)$ denote the highest point in the sagittal section $S_{l_l}(x)$.

$$S_{l_l}(x) = \{p \in R_{l_l}, p \cdot u_x = x\}$$

$$q_{l_l}(x) = \arg\max_{p \in S_{l_l}(x)} p \cdot u_y$$

Then, the next guess for the left lower orbit is the point in the most inferior direction $-u_y$. We can describe this point as the point that minimizes the following objective function $F_{l_l}(x)$:

$$F_{l_l}(x) = q_{l_l}(x) \cdot u_y$$

$$l'_l = q_{l_l}(p), \text{ s.t. } F_{l_l}(p) = \min F_{l_l}(x)$$

In practice, since the number of points on a skull is finite, there are a finite number values of x for which S_{l_l} is non-empty. Moreover, S_{l_l} will be a finite set. Thus, $\arg\max_{p \in S_{l_l}(x)} p \cdot u_y$ and $\min F_{l_l}(x)$ are both solved by searching through all feasible points.

Due to the similar definition of the right orbit, the same is done for the right orbit and its landmark region R_{l_r}.

$$S_{l_r}(x) = \{p \in R_{l_r}, p \cdot u_x = x\}$$

$$q_{l_r}(x) = \arg\max_{p \in S_{l_r}(x)} p \cdot u_y$$

$$F_{l_r}(x) = q_{l_r}(x) \cdot u_y$$

$$l'_r = q_r(p), \text{ s.t. } F_{l_r}(p) = \min F_{l_r}(x)$$

In step 3, we define the Frankfurt plane by applying PCA to the four landmarks. The first two principal components define the direction vectors of the plane, and the centroid of the four landmarks define the point through which the plane passes.

2.3.3 VTK's Deformed Point Set

To create landmark regions in our algorithm, we will be utilizing the Python library The Visualization Toolkit's (VTK) [10] *Deformed Point Set* algorithm. These landmark regions are sets of candidate points to improve a landmark l. As input for *Deformed Point Set*, we will use a skull mesh S, a transformed version of S, and the points of a sphere C. The output of *Deformed Point Set* will be a point set C' that is a transformed version of C. We translate C' to centre around l to create the landmark region corresponding to l. See section 3.3.1 for the full landmark region creation process.

Deformed Point Set alters a point set P based on a transformation done on a mesh $M = (P_M, F_M)$. *Deformed Point Set* creates a set of interpolation weights $W \in \mathbb{R}^n$ for each point in P. A weight $w_i \in W$ is used as the constant for $m_i \in P_M$ for a linear combination of the points in M. Then these weights are applied to the transformed version of the mesh to obtain a transformed version of the point set. These interpolation weights are computed using the mean value coordinates on a triangular mesh algorithm (MVC) by Ju et. al in [2]. For a single point in P, MVC computes a set of scalars that are used as the constants in a linear combination of the points on M. This can be illustrated by the following:

Let M be a mesh, and $\{m_1, m_2, ..., m_n\} \subset \mathbb{R}^3$ be the set of points of M. Suppose $p \in \mathbb{R}^3$ is a point.

$$MVC(M, p) = v \in \mathbb{R}^n, \sum_{i=1}^{n} v_i m_i = p$$

Now we can formally describe *Deformed Point Set*.

Algorithm 2 VTK's DeformedPointSet

Input: M, M', P, a mesh, a transformed mesh, and a point set
Let $\{m_i\}_{i=1}^n$ be the points of M, and $\{m'_i\}_{i=1}^n$ be points the M'
Output: P', a point set

1: **for** each point p_i in P **do**
2: $\quad v_i = MVC(M, p_i)$
3: $\quad p'_i = \sum_{i=1}^{n} v_i m'_i$
4: **end for**
5: $P' = \{p'_i\}$

Chapter 3

Finding the Bandeau Curve

Let $S = (P_S, F_S)$, $T = (P_T, F_T)$, be two skull meshes, where $P_S, P_T \subset \mathbb{R}^3$ are point sets in 3D space, and $F_S \subset P_T^3, F_T \subset P_T^3$ are the faces of each mesh. We assume we know the points $L_S \subset P_S$ that are the locations of the 6 important landmarks (Section 2.1) on the skull S. S will be often referred to as the reference or source skull, whereas T will often be referred to as the target skull. We assume we do not know the landmark locations of T.

Throughout this paper, we will denote the unit vector in the right-lateral direction, the superior direction, and the anterior direction to be u_x, u_y, u_z respectively. (See Section 2 for definition of theses directions)

Finding the 2D bandeau curve consists of three main parts: finding initial guesses of landmarks with registration, improving the the initial guess, and then using the landmarks to find the 2D bandeau curve. We will go over each part separately and discuss the intuition and reason behind it.

3.1 Bandeau Curve Identification Algorithm Overview

We design an algorithm for finding the bandeau curve of a skull with unknown landmark locations. We start by locating the landmarks of the unknown skull with the help of a skull with known landmark locations. We then compute the bandeau curve once all landmarks

have been found. Below is an overview of the algorithm, in its basic steps. For a more formal description of the algorithm, check section 3.5.

Input: Reference skull S with known landmark locations and direction vectors, unknown target skull T

1. (Optional) Re-orient the target skull so that it matches the orientation of a reference skull (Section 3.2.1)
2. Register a reference skull to the target skull (Section 3.2)
3. Use transformed landmarks on registered reference skull to get initial landmark guesses on the target skull 3.1
4. Create lattice of deformed spheres used to find landmark regions (Section 3.3.1)
5. Repeat until convergence:
   (a) Find the landmark region for each landmark (Section 3.3.1)
   (b) Improve each landmark by selecting a point in the landmark region (Section 3.3.2)
6. Use landmarks to find the bandeau curve and its midpoint (Section 3.4)

Output: Bandeau curve and its midpoint of the target skull

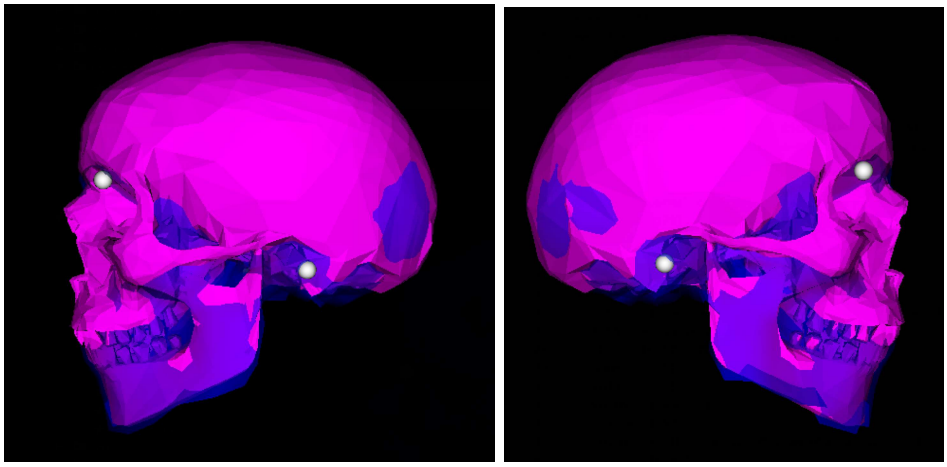


Figure 3.1: A registered skull (blue) [12], registered to a target skull (pink) [11]. Initial landmark locations for the target skull in white. Initial lower orbit locations not visible because it is located inside the pink mesh.

Our algorithm follows a similar structure to the Frankfurt plane identification algorithm in [14] (See Section 2.3.2). In both algorithms, initial landmark positions are found via registration, and the landmark positions are improved by searching through a landmark region. In our algorithm, we use a different registration algorithm that performs better on deformed skulls (Section 3.2). Furthermore, in our algorithm, the landmark regions are created differently. Instead of using a fixed elliptical region as the landmark region like in [14], our algorithm modifies the shape of the landmark region based on the location of the landmark (Section 3.3.1).

3.2 Registration

Our bandeau curve algorithm finds the points of the bandeau curve of a given input target skull. The algorithm also requires a reference skull with known landmark locations as input. We create a new skull mesh by registering (See Section 2.3.1 for point set registration) the points on the reference skull to the points on a target skull. This new skull mesh is created by transforming the points on the reference skull. We call this new mesh the registered reference skull. On our registered reference skull, we take the transformed points of the known landmarks to be the initial guess for the landmark locations on the target skull.

In [14], FICP is used for registration. Due to the nature of human skulls, one person's skull may vary in shape and in size to the next person's. This is especially true in our context, where we are dealing with infant skull with craniosynostosis — skulls which are deformed. We choose to use CPD as our registration algorithm because of its ability to allow non-rigid transformations, hoping to better fit skulls of various shapes.

Moreover, we employ one more technique to obtain a better registered reference skull. We register multiple source skulls to our target skull. Each source skull is a reference skull with known landmark locations. Due to the varying types of craniosynostosis and skull shapes, we collect several skulls hoping at least one will match any given target skull well during registration. We pick a set of reference skulls that attempts to fill the different categories of skull shapes. We define the registration error of a registered reference skull and a target skull as the sum of all Euclidean distances between a point in the registered reference skull and its paired point in the target skull. The "best" registration is the registration where the registration error is the least. We define this distance between the registered reference skull and the target skull to be the registration error. We choose the reference skull with the least registration error and use the transformed landmarks after registration to get our initial guesses for the landmark locations on the target skull. Alternatively, another method could be to have a list of reference skulls and to go down this list until your registration error meets a certain threshold value.

3.2.1 Initial Transformation

Prior to registration, there is an optional step that can improve the matching between registered skull and the target skull. This optional step is making initial transformations to the target skull so that its orientation matches the reference skull. We reflect the points on the target skull on certain axes so that the orientation of target skull is the same as the reference skull. In our test cases, we obtained skulls where the anterior direction on the target skull would be flipped from the anterior direction on the reference skull. To get a better result in the registration algorithm, we reflected the points on the target skulls along different axes so that the values of u_x, u_y, u_z with respect to the target skull are the same values of u_x, u_y, u_z with respect to the reference skull. We replace the target skull with this reflected version for our input to our registration algorithm.

Of course, this is not ideal, as this requires manual work to know how the target skull is oriented. Also, registration algorithms are designed to match orientation. However, at its core, CPD is an optimization problem that finds a local optimal solution with no guarantee

of global optimality. By giving CPD a better starting position, we tend to get a better solution to the optimization problem.

3.3 Landmark Improvements

After obtaining the initial guess for each landmark from registration, we refine the position of each landmark iteratively, like in [14]. (Section 2.3.2) Each landmark is improved separately, in an iterative fashion.

1. Select a set of points in a region around the landmark as candidate points. We call this set of candidate points the landmark region (Section 3.3.1)
2. Pick a point in the landmark region as the next guess for the landmark (Section 3.3.2)
3. Repeat 1-2 with new guess until convergence

In each iteration of improving a landmark l, we create a landmark region by taking the intersection of points in a neighbourhood around l and the target skull, and select a point within the landmark region as the next guess for l.

3.3.1 Landmark Region Creation

Let $l \in P_T$ be a landmark or a landmark guess on the skull mesh $T = (P_T, F_T)$. We define a landmark region $R_l \subset P_T$ of a landmark l to be a set of points on the skull T where each point in R_l is a candidate point for the next guess of l. We obtain R_l by taking points in P_T that are close to l. In our early tests, we obtained the landmark region by taking the intersection of P_T and a sphere of fixed radius r centered around the l. Like in the Frankfurt plane identification algorithm in [14] (See Section 2.3.2), we obtained the landmark region by taking the intersection of P_T and a neighbourhood of fixed size centered around l . In [14], an elliptical region of fixed size centered around l was used as this neighbourhood. In our early tests, we used a spherical region of fixed size centered around l. However, because the target skull landmark locations are unknown, the size of the neighbourhood

around l will be chosen based on the reference skull. The methods of picking a point in the landmark region will not accurately find the landmark if the landmark region contains too many or too little of the points on the skull (methods shown in section 3.3.2). Thus, we transform the points on a sphere around the landmark based on the transformation used to transform the reference skull to the registered reference skull. To do this, we use VTK's *Deformed Point Set* (See section 2.3.3).

We want to select a neighbourhood around a landmark based on how the transformation created during registration affected the neighbourhood around the landmark. We do not directly deform a sphere around l because that may move the centre of the sphere away from l, and may no longer include l. Instead we will create a grid of spheres in 3D space, and deform each of them using *Deformed Point Set*. We will choose the shape of the deformed sphere closest to l as the shape for the neighbourhood around l. Let S' be the registered reference skull mesh obtained by registering S to T. For each sphere C, we take the deformed sphere C' to be the output of $DeformedPointSet(S, S', C)$. The result of these deformations is many deformed spheres in 3D space. For a given landmark l, the shape of R_l will be the shape of the closest deformed sphere C'_l. By shifting the points in C'_l so that its centre is at l, we get a region around the landmark. We intersect this region with the skull to get the landmark region.

To obtain our deformed spheres, we first create a lattice of points of a bounding box for the original reference skull. Each point will be the centre of a sphere of fixed radius. For each sphere C in the lattice, the deformed sphere C' is the output of $Deforemd\text{-}PointSet(S, S', C)$.

Again, let S be the reference skull, S' be the transformed reference skull after registration, T be the target skull, and l be a guess for a landmark. The landmark region R_l is found by the following:

Input: $S = (P_S, F_S), S', T, l$

1. Create a lattice of points $\mathbb{L}$ of a bounding box for P_S
2. For each point $o \in \mathbb{L}$, create a sphere $C(o)$ with o as the centre
3. For each pair $(C(o), o)$, compute the pair $(C'(o), o')$, where
   $C'(o) = DeformedPointSet(S, S', C(o))$ and $o' = DeformedPointSet(S, S', o)$
4. Let $\mathbb{L}' = \{o' : o' = DeformedPointSet(S, S', o), o \in \mathbb{L}\}$.
   Select pair $(C'(o_l), o'_l)$ by computing $o'_l = \arg\min_{o' \in \mathbb{L}'} d(l, o')$

5. Compute $C'(o_l) + (l - o'_l)$ by shifting all points in $C'(o_l)$
6. Compute the landmark region $R_l = P_T \cap (C'(o'_l) + (l - o'_l))$

Output: R_l, the landmark region for l

Implementation note: We deform spheres with a radius of 5mm to find the landmark region for porions, and we deform spheres of radius 8mm to find the landmark regions for the orbits, with a lattice of 8000 points (divide each axis into 20 parts linearly). These numbers were found empirically to match the reference skull's size.

In the bandeau finding algorithm, the deformed spheres and their deformed centres $(C'(o), o')$ are computed and stored only once during the algorithm and not computed each time a landmark region is needed.

Algorithm 3 Finding the landmark region

Input: T target skull, l landmark point, all pairs $(C'(o), o')$, the pairs of deformed spheres and their centre

Output: R, the landmark region for l

1: Select $C'(o_l)$ such that $o'_l = \arg\min_{o' \in \mathbb{L}'} d(l, o')$
2: Compute $C'(o_l) + (l - o'_l) = \{c + (l - o'_l) : c \in C'(o_l)\}$
3: Compute $R = P_T \cap (C'(o'_l) + (l - o'_l))$, the landmark region

3.3.2 Picking from the Landmark Region

Once we have the landmark region for a landmark l, we pick a point in the landmark region the be the next value for l. How we pick the point in the landmark region depends on what type of landmark l is. Depending on if l is a porion, a lower orbit, or an upper orbit, a different method will be used to pick the next value of l. We use the methods from [7] to select the next value of l for the lower orbits and the porions. (See Section 2.3.2)

For the upper orbits, we follow a similar idea to the lower orbits, but we reverse the directions. Let u_l be the current guess for the left upper orbit. Let R_{u_l} be the landmark region for the left upper orbit. We define $S_{u_l}(x)$ to be a sagittal section of R_{u_l}, and we

take the lowest point in each sagittal section, denoted $q_{l_l}(x)$. Then we pick highest point out of all values of $q_{l_l}(x)$ to be the next guess for the left upper orbit.

$$S_{u_l}(x) = \{p \in R_{u_l}, p \cdot u_x = x\}$$

$$q_{u_l}(x) = \arg\min_{p \in S_{u_l}(x)} p \cdot u_y$$

$$F_{u_l}(x) = q_{u_l}(x) \cdot u_y$$

$$u'_l = q_{u_l}(p), \text{ s.t. } F_{u_l}(p) = \max F_{u_l}(x)$$

The exact same procedure is done with the right upper orbit, except with its own landmark region. $S_{u_r}, q_{l_r}, F_{u_r}$ are defined similarly to the left upper orbit.

$$u'_r = q_{u_r}(p), \text{ s.t. } F_{u_r}(p) = \max F_{u_r}(x)$$

Similar to lower orbits, we solve these optimization problems by searching through all feasible points, since in practice, a skull mesh has a finite number of points.

With the initial transformation step of section 3.2.1, we know u_x, u_y, u_z by using the same directions of the known reference skull. Otherwise, we obtain u_x, u_y, u_z by deforming the directional unit vectors with respect to the reference skull by using *Deformed Point Set* with the reference skull and the registered reference skull. The deformed unit vectors are then the directions for the target skull.

We improve each landmark until it converges to a single point or when we reach a maximum number of iterations. Since the landmarks are picked by solving an optimization problem, as long as the landmark region is not too large, a landmark region centered around a true landmark location will not change the landmark guess.

3.4 Using the Landmarks

Once all landmarks have converged, we use the landmark to locate the Frankfurt plane, the bandeau plane, and then the bandeau curve of the target skull. We find the Frankfurt plane, denoted P_F, using the lower orbits and the porions. Since there are four landmarks total, we use PCA to find the plane of best fit for the four points. Let n_{P_F} be the normal of the Frankfurt plane.

The bandeau plane lies 5mm above the upper orbits (Definition 2.2.1). We take the average of the upper orbits and move it up 5mm to find the point that intersects the bandeau curve. We define this point as the midpoint of the bandeau, denoted m_B. The bandeau plane is then defined by the point m_B and the normal vector n_{P_F}. Next, we take the intersection of P_B and the target skull, and this intersection will be denoted $I = P_B \cap P_T$.

$$m_B = \frac{(u_r + u_l)}{2} + 5u_y$$

However, the bandeau curve itself only reaches until above the porions (Definition 2.2.2). We start by defining $\Delta p = p_r - p_l$ be the direction from the left porion to the right porion. Then we define the anterior plane P_a with direction vectors u_y and Δp. We remove any points in I that are behind P_a in the anterior direction. Consider a point $p_I \in I$. Let $P_a(p_I)$ be the point on P_a that is closest to p_I. Now take the direction vector from $P_a(I)$ to p_I, $d_I = p_I - P_a(p_I)$. For points behind P_a, $d_I \cdot u_z$ is positive. Then the bandeau curve is defined to be the following:

$$d_I = p_I - P_a(p_I)$$
$$B = \{p_I \in I : d_I \cdot u_z < 0\}$$

The algorithm returns both the points on the bandeau curve B, and the midpoint m_B. The midpoint is especially helpful for aligning different bandeau curves together for comparison.

3.5 Bandeau Curve Identification Algorithm

In this section, we give a formal description of the bandeau curve algorithm.

Let $S_1, ..., S_n$ be a set of reference skulls. And let L_{S_i} be the landmark points in S_i. Let T be the target skull. We assume S_i to be oriented the same way in 3D space.

Algorithm 4 Finding the 2D Bandeau Curve

Input: $S_1, ...S_n, L_{S_1}, ...L_{S_n}, T, u_x, u_y, u_z$, skull meshes and the landmark locations, and direction vectors of the reference skulls, and constants ϵ, max_iter

Output: B, m_B, points on the bandeau curve, and the midpoint

Find the orientation of T
Transform T to match orientation of S_1
for $i \leftarrow 1 : n$ **do**
    Compute $S_i' = \text{CPD}(S_i, T)$
end for
Select $S = S_j$ s.t. S_j has the best registration $\forall S_i$
Compute lattice $\mathbb{L}$ of a bounding box for P_S
for point $o \in \mathbb{L}$ **do**
    Compute $C(o)$
    Compute $C'(o) = DeformedPointSet(R, R', C(o))$
    Compute $o' = DeformedPointSet(R, R', o)$
end for
Let $L_S^{'}$ be the set of points in $P_{S'}$ after transforming the points L_S
for landmark l' in L_S' **do**
    for $i \leftarrow 1 : max\_iter$ **do**
        Find the landmark region $R_{l'}$
        Select $l'_{new} \in R_{l'}$ based on type of landmark
        if $d(l', l'new) < \epsilon$ **then**
            $l' = l'_{new}$
            break
        else
            $l' = l'_{new}$
        end if
    end for
end for
Compute Frankfurt plane P_F
Compute bandeau plane P_B
Compute midpoint, bandeau curve m_B, B

Chapter 4

Results

In our tests, we use two different skull meshes as reference skulls with 39 different target skull meshes. One reference skull is a metopic skull (a type of deformed skull), and the other reference skull is a normal skull model generated using 103 normal skulls [8]. Of the 39 target skulls, 34 are metopic, 4 are normal, and 1 is sagittal (another type of deformed skull). It is important to note the mesh of all metopic skulls (and 1 normal skull) contained the entire skull with a small portion of the neck, while the mesh of 3 out of 4 normal skulls and the mesh sagittal skull only contained the top of the skull to the top of the mouth. Everything from the mouth down was not part of these meshes. These differences in the skull mesh affect how well the registered skull mesh matches the target skull, and ultimately the accuracy of finding the landmarks.

In our tests, during the registration step, for each skull mesh, we take a subset of the points in the mesh as our input for CPD. The size of the subset is 1200, and the points are chosen to maximize the average distance between each point. We do this for two reasons — one, to ensure that that the cardinality of the point sets used as input for CPD are the same, and two, to decrease the computation time for CPD. Other parameters for our tests are the following. Our max number of iterations for improving the landmarks is 10, and we use $\epsilon = 10^{-4}$. The lattice of the bounding box of the target skull has 8000 points, where each axis is divided into 20 parts linearly. To find the landmark regions for porions, we deformed spheres of radius 5mm, and we deformed spheres of radius 8mm to find the landmark regions of the upper and lower orbits.

Due to the reliance on landmarks in the algorithm to find the bandeau curve, we look at how well the landmarks are matched in the algorithm to the true landmark position.

The true landmark positions were all hand-picked, and so may contain some human error. To account for this, we will consider a *good* landmark guess to be one that is less than 1cm away from the hand-picked landmark location.

In our tests, in 25 of the 39 target skulls, at least 4 of the 6 landmarks were well identified. We notice that with the metopic target skulls, the landmarks were much better identified with the metopic reference skull than the normal reference skull. Likewise, the landmarks on the normal target skull were better identified with a normal reference skull than a metopic reference skull (See Tables 4.1, 4.2). We believe the algorithm performs better when the shape of the reference skull is similar to the shape of the target skull. This includes differences in shape due to skull deformities, but differences in shape due to the mesh representation of the skull. Since the normal reference skull was cut off before the mouth, we believe it becomes a better reference for target skull meshes which also are cut off before the mouth. You can see this with how well the algorithm performs with the sagittal target skull and the normal reference versus using the metopic reference skull with the sagittal target skull.

When the algorithm did not accurately find the landmarks, we noticed that the initial guesses for the landmark were too far away from the true landmark positions. This typically occurred because the registered reference skull did not match the shape of the target skull. Furthermore, the registration error — the average distance between matched points in the registered reference skull and the target skull was recorded in our tests. Of the 39 target skulls, 18 target skulls performed better with the reference skull with the lower registration error.

Results with ≥ 4 Good Landmarks Guesses		
	Metopic Reference	Normal Reference
Metopic Target (of 34)	22	5
Normal Target (of 4)	0	2
Sagittal Target (of 1)	0	1

Table 4.1: Algorithm results based on # *good* landmarks

Average Distance from True Landmark		
	Metopic Reference	Normal Reference
Metopic Target	1.6cm	3.1cm
Normal Target	2.2cm	1.6cm
Sagittal Target	1.8cm	0.4cm

Table 4.2: Algorithm results based on distance

The results of these tests were not perfect — in fact, only 1 target skull had all of its 6 landmarks guessed within 1cm with the algorithm. However, we believe that given at least 4 good landmark guesses, that the algorithm still performs well enough. By well enough, we mean that all 6 landmarks are guessed very close to the true landmark position. We can see this in table 4.3, where given at least 4 good landmark guesses, the average distance between the true landmarks and the landmarks produced in the algorithm is about 1cm. Our algorithm does not need to output the exact bandeau curve — it only needs to output a curve that well represents the bandeau in 2D.

Avg. Distance when $\geq$ 4 Good Landmarks		
	Metopic Reference	Normal Reference
Metopic Target	1.3cm	0.8cm
Normal Target	N/A*	0.9cm
Sagittal Target	N/A*	0.4cm
*Values are N/A when there are no skulls with $\geq$ 4 Good Landmarks		

Table 4.3: Algorithm results based on distance and # *good* landmarks

A full look at the results of the algorithm can be seen in Appendix A. The tables include the distance of all landmark guesses from their true landmark position for each skull tested.

Chapter 5

Conclusion and Further Work

Testing shows that the algorithm accurately outputs the bandeau curve for 25 of 39 of the target skulls when using the best performing reference skull. This shows the importance of finding a good reference skull when using the algorithm. Unfortunately, the results show that a reference skull with lower registration error does not always correlate to having more accurate results. Further work in finding the best reference skull for a target skull would improve the accuracy of the algorithm greatly. A simple idea is to manually classify each target skull and pick a reference skull based on the classification. Ideally, however, the ultimate goal is to have an automatic method of picking a reference skull.

The algorithm does not accurately find the bandeau curve for all skulls. Due to the importance of the initial registration of the reference skull to the target skull, it is worth exploring different registration algorithms. A good registration algorithm would also eliminate the optional manual step of re-orienting the target skulls before registration. This would lead to a more automatic process. Medical image registration is a difficult task, and bottlenecks the performance in the bandeau curve identification algorithm.

References

[1] Paul J. Besl and Neil D. McKay. Method for registration of 3-D shapes. In Paul S. Schenker, editor, *Sensor Fusion IV: Control Paradigms and Data Structures*, volume 1611, pages 586 – 606. International Society for Optics and Photonics, SPIE, 1992.

[2] Tao Ju, Scott Schaefer, and Joe Warren. Mean value coordinates for closed triangular meshes. In *ACM SIGGRAPH 2005 Papers*, SIGGRAPH '05, page 561–566, New York, NY, USA, 2005. Association for Computing Machinery.

[3] Christopher Lopez, Anand Kumar, Alexander Lin, Christopher Bonfield, Jeffrey Weinzweig, Thomas Naidich, Christopher Smith, and Peter Taub. Demystifying the "triple point: " technical nuances of the fronto-orbital advancement. *Journal of Craniofacial Surgery*, 29:1, 02 2018.

[4] A. Myronenko and X. Song. Point set registration: Coherent point drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12):2262–2275, 2010.

[5] Helena MD Pappa, David MD Richardson, Andy A. C. MD Webb, and Paul MD May. Individualized template-guided remodeling of the fronto-orbital bandeau in craniosynostosis corrective surgery. *Journal of Craniofacial Surgery*, 20:178–179, 2009.

[6] J. M. Phillips, R. Liu, and C. Tomasi. Outlier robust icp for minimizing fractional rmsd. In *Sixth International Conference on 3-D Digital Imaging and Modeling (3DIM 2007)*, pages 427–434, 2007.

[7] J. M. Phillips, R. Liu, and C. Tomasi. Outlier robust icp for minimizing fractional rmsd. In *Sixth International Conference on 3-D Digital Imaging and Modeling (3DIM 2007)*, pages 427–434, 2007.

[8] N. R. Saber, J. Phillips, T. Looi, Z. Usmani, J. Burge, Drake J., and P. C. Kim. Generation of normative pediatric skull models for use in cranial vault remodeling

procedures. *Child's nervous system : ChNS : official journal of the International Society for Pediatric Neurosurgery*, 28(3):405–410, 2012.

[9] KE Salyer and JD Hall. Bandeau–the focal point of frontocranial remodeling. *The Journal of craniofacial surgery*, 1(1):18—31, January 1990.

[10] W. Schroeder, K. Martin, and B. Lorensen. *The Visualization Toolkit (4th ed.)*. Kitware, 2006.

[11] TurboSquid. free c4d mode anatomy body, 2011. [Online; accessed December 14, 2020].

[12] TurboSquid. Skull 3d - turbosquid 1452999, 2019. [Online; accessed December 14, 2020].

[13] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1):37 – 52, 1987. Proceedings of the Multivariate Statistical Workshop for Geologists and Geochemists.

[14] Yuan Cheng, Wee Kheng Leow, and Thiam Chye Lim. Automatic identification of frankfurt plane and mid-sagittal plane of skull. In *2012 IEEE Workshop on the Applications of Computer Vision (WACV)*, pages 233–238, 2012.

APPENDICES

Appendix A

Full Algorithm Results

A.1 Algorithm Results with the Normal Reference Skull

Distance (cm) From True Landmark Locations						
Target Skull	Right Lower Orbit	Left Lower Orbit	Right Upper Orbit	Left Upper Orbit	Right Porion	Left Porion
Metopic Skull 1	0.135	0.586	11.09	4.649	1.597	2.297
Metopic Skull 2	3.799	3.620	5.287	2.456	3.144	2.119
Metopic Skull 3	3.929	4.051	4.709	0.151	1.515	1.741
Metopic Skull 4	0.347	1.757	0.275	0.130	0.397	1.225
Metopic Skull 5	0.506	1.376	0.138	0.209	0.640	2.017
Metopic Skull 6	2.193	0.158	0.068	0.123	1.660	1.714
Metopic Skull 7	2.778	1.982	1.726	7.903	1.849	4.929
Metopic Skull 8	0.116	0.584	0.108	0.495	1.586	1.843
Metopic Skull 9	7.043	8.450	8.945	6.575	5.003	5.344
Metopic Skull 10	7.071	7.471	6.926	6.905	4.016	4.544
Metopic Skull 11	0.144	0.044	3.220	0.214	0.866	2.103

Table A.1: Algorithm results using the normal reference skull

Distance (cm) From True Landmark Locations						
Target Skull	Right Lower Orbit	Left Lower Orbit	Right Upper Orbit	Left Upper Orbit	Right Porion	Left Porion
Metopic Skull 12	7.144	8.029	7.081	5.093	4.100	5.020
Metopic Skull 13	0.152	0.024	0.4880	0.328	0.580	1.955
Metopic Skull 14	8.305	8.005	12.50	7.698	3.977	4.981
Metopic Skull 15	0.493	2.743	0.121	5.776	0.644	2.613
Metopic Skull 16	15.66	15.70	15.48	4.384	13.75	2.716
Metopic Skull 17	0.647	0.410	4.333	3.814	0.709	1.769
Metopic Skull 18	2.327	2.331	1.090	0.132	0.701	2.220
Metopic Skull 19	0.328	1.790	5.165	5.395	0.690	2.961
Metopic Skull 20	0.095	3.165	2.181	3.889	1.885	4.244
Metopic Skull 21	0.156	2.207	5.779	0.616	1.904	1.347
Metopic Skull 22	2.510	4.309	7.991	0.243	3.002	4.430
Metopic Skull 23	0.074	2.834	0.093	4.743	0.346	2.108
Metopic Skull 24	9.183	6.930	8.377	7.592	7.637	5.866
Metopic Skull 25	0.074	3.478	0.150	4.257	1.041	3.344
Metopic Skull 26	2.935	2.689	2.800	5.930	1.673	2.491
Metopic Skull 27	0.121	2.731	0.170	8.049	1.306	2.438
Metopic Skull 28	2.678	3.714	7.531	0.070	1.777	4.510
Metopic Skull 29	7.090	5.233	0.011	0.129	1.248	2.552
Metopic Skull 30	0.149	2.844	7.200	6.287	1.936	3.672
Metopic Skull 31	0.084	0.115	0.093	1.056	0.759	1.244
Metopic Skull 32	6.840	6.068	9.743	5.724	4.689	4.114
Metopic Skull 33	0.167	2.152	0.022	0.057	1.058	1.595
Metopic Skull 34	0.134	2.484	2.064	0.234	1.336	0.842
Normal Skull 1	1.380	2.649	0.391	7.551	0.788	3.002
Normal Skull 2	0.082	0.060	3.592	0.066	0.428	1.472
Normal Skull 3	0.166	0.156	5.646	7.034	2.171	1.853
Normal Skull 4	0.765	0.172	0.417	0.304	2.567	1.428
Sagittal Skull 1	0.041	0.511	0.659	0.046	0.411	0.624

Table A.2: Algorithm results using the normal reference skull pt. 2

A.2 Algorithm Results with the Metopic Reference Skull

Distance (cm) From True Landmark Locations						
Target Skull	Right Lower Orbit	Left Lower Orbit	Right Upper Orbit	Left Upper Orbit	Right Porion	Left Porion
Metopic Skull 1	2.027	0.651	3.351	1.193	0.493	0.881
Metopic Skull 2	1.817	0.127	0.033	0.628	0.233	2.119
Metopic Skull 3	2.628	10.35	0.328	0.151	1.229	1.965
Metopic Skull 4	1.408	0.120	0.327	0.130	0.397	7.391
Metopic Skull 5	1.580	1.101	0.141	0.209	0.410	0.554
Metopic Skull 6	1.286	0.072	0.068	5.798	2.170	0.387
Metopic Skull 7	1.804	0.034	1.726	0.316	0.822	0.945
Metopic Skull 8	1.949	0.584	0.108	0.495	0.107	0.590
Metopic Skull 9	1.568	0.114	0.135	0.079	1.372	1.240
Metopic Skull 10	4.381	2.418	6.456	3.468	3.564	2.813
Metopic Skull 11	1.843	0.044	3.220	0.214	0.375	0.592

Table A.3: Algorithm results using the metopic reference skull

Distance (cm) From True Landmark Locations						
Target Skull	Right Lower Orbit	Left Lower Orbit	Right Upper Orbit	Left Upper Orbit	Right Porion	Left Porion
Metopic Skull 12	1.698	0.101	0.641	0.874	1.583	0.703
Metopic Skull 13	2.006	0.036	0.490	0.589	0.580	0.819
Metopic Skull 14	1.387	0.063	0.208	0.033	1.175	0.157
Metopic Skull 15	2.263	0.191	0.121	0.998	0.407	0.455
Metopic Skull 16	2.280	0.141	15.49	0.260	0.285	0.323
Metopic Skull 17	1.755	0.233	3.201	0.092	0.397	1.116
Metopic Skull 18	1.510	0.201	1.090	0.131	0.701	0.775
Metopic Skull 19	1.828	0.302	5.949	13.72	0.279	1.109
Metopic Skull 20	1.557	0.084	1.631	0.246	0.513	0.428
Metopic Skull 21	1.977	0.150	5.659	0.841	0.450	13.81
Metopic Skull 22	1.710	3.380	0.225	0.243	0.421	1.441
Metopic Skull 23	1.959	0.129	0.219	10.68	0.346	0.199
Metopic Skull 24	1.157	0.125	8.197	0.435	0.951	1.207
Metopic Skull 25	1.830	0.046	0.146	1.877	0.631	0.894
Metopic Skull 26	1.999	0.527	10.74	0.469	8.296	1.875
Metopic Skull 27	2.098	0.144	0.136	8.598	0.307	0.136
Metopic Skull 28	1.534	0.454	0.158	14.41	0.343	0.388
Metopic Skull 29	5.704	6.052	0.220	0.129	0.676	0.376
Metopic Skull 30	2.000	0.075	12.99	1.871	0.479	1.015
Metopic Skull 31	1.985	0.163	0.054	7.415	0.636	0.928
Metopic Skull 32	1.985	0.038	0.110	9.217	0.337	0.225
Metopic Skull 33	2.025	2.152	0.022	0.186	0.977	0.555
Metopic Skull 34	1.882	0.072	0.119	0.094	0.278	0.300
Normal Skull 1	2.242	1.191	0.391	4.406	3.799	1.102
Normal Skull 2	1.895	0.182	3.592	0.307	0.428	12.03
Normal Skull 3	2.013	0.070	5.749	7.634	0.821	0.766
Normal Skull 4	2.227	0.531	0.554	0.304	1.033	1.054
Sagittal Skull 1	3.123	2.800	0.659	2.264	0.829	1.846

Table A.4: Algorithm results using the metopic reference skull pt. 2