5-2023

Non-Rigid Registration with Deep Learning and Conformal Harmonic Maps

Daniel Billmann
billmannd@gmail.com

Non-Rigid Registration with Deep Learning and Conformal Harmonic Maps

A Thesis presented

by

Daniel Billmann

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Master of Science

in

Computer Science

Stony Brook University

May 2023

Stony Brook University

The Graduate School

Daniel Billmann

We, the thesis committee for the above candidate for the

Master of Science degree, hereby recommend

acceptance of this thesis

Dr. Xianfeng David Gu
SUNY Empire Innovation Professor - Department of Computer Science

Dr. Haibin Ling
SUNY Empire Innovation Professor - Department of Computer Science

Dr. Xiaojun Bi
Associate Professor - Department of Computer Science

This thesis is accepted by the Graduate School

Celia Marshik

Interim Dean of the Graduate School

Abstract of the Thesis

Non-Rigid Registration with Deep Learning and Conformal Harmonic Maps

by

Daniel Billmann

Master of Science

in

Computer Science

Stony Brook University

2023

We present a novel fully-automated approach to non-rigid registration for high-resolution facial scans using conformal harmonic maps. The novelty of this paper is its use of applied deep learning models to prepare data for geometric algorithms to compute non-rigid registration. We use facial detection to both constrain the boundary of the face and provide a mechanism to manipulate the input mesh. We use conformal harmonic maps[7] to map a dense 3D point cloud to the closed unit disc $D_1(0)$ and optimize the weights of each edge. Our experiments show the effectiveness of this approach.

Dedication Page

For my grandfather, who exclaimed "I know you can do it!" in response to me telling him I was pursuing my Master's of Computer Science.

Contents

Acknowledgements

Thank you to Dr. David Xianfeng Gu for his poignant insights and to Zhou Zhao for being my sounding board along the way.

This would not have been possible without the continued love and support from my parents - thanks Mom and Dad.

Finally, a special thank you to Heidi Warning and Jacob Dray for helping me get to the finish line.

1 Introduction and Prior Research

Advances in 3D digital scanning technology and computer vision in recent years, particularly with the capabilities of deep neural networks, have produced many impressive models of the human face. These models are able to quickly recognize facial features and track their movements in real time. We wanted to harness the usefulness and efficiency of these models to denoise facial scan data and return a consistent boundary of the face for 3D image registration. In this work, we make two contributions: First, we use a deep learning model to capture the consistent boundary of a 3D facial scan for non-rigid registration. Second, we show the importance of a consistent boundary in the context of non-rigid registration by comparing it to some example inconsistent boundaries we constructed. When we use the term consistent boundary, we are referring to a consistent physical boundary between two surfaces. For example, two circles with radii 1 and 1.01 have a consistent boundary, while a circle with radius 1 and a square with area 2 do not. We constructed inconsistent boundaries using the deep learning library by removing "chunks" of the face, being careful to preserve important regions of the face for computation and measurement. Our research uses conformal harmonic maps to understand surface-level effects of orthopedic maxillary expansion (OME) in pediatric patients. OME is a common expansion technique that widens the upper jaw in order to fit the upper and lower teeth together better. It primarily performed on children aged 8-18 using devices such as a palatal expander. We chose conformal harmonic maps because of their angle-preserving properties and because of the Uniformization Theorem, which proves a Riemannian surface is conformally equivalent to a unit disc. Using a unit disc is important for computational efficiency given the high density of mesh inputs in $\mathbb{R}^3$. Conformal maps were used in prior research for both tracking facial surfaces in 3D [15] and matching surfaces in 2D and 3D [2, 6, 8, 13, 14]. We use a deep learning model and quadric edge collapse decimation to preprocess the mesh. Then we conformally map each mesh to a unit disc and pass key landmarks to a Möbius transformation to rotate the discs. Finally we construct a non-rigid mapping from one disc to the other using a K Nearest Neighbors (KNN) classifier with $k = 1$. We also provided a mechanism to analyze metric landmarks to understand how the surface has changed. The rest of the paper is organized in the following sections: theoretical background on the operations themselves (2), followed by methodology and step-by-step implementation (3), then the analysis and results (4), before the conclusion and potential next steps (5).

2 Theoretical Background

2.1 Quadric Edge Collapse Decimation[4]

2.1.1 Description

Quadric Edge Collapse Decimation (QECD) is a surface simplification algorithm that removes unimportant edges $(v_1, v_2) \mapsto \bar{v}$ and unifies them to a single point [4] (see figure 1).

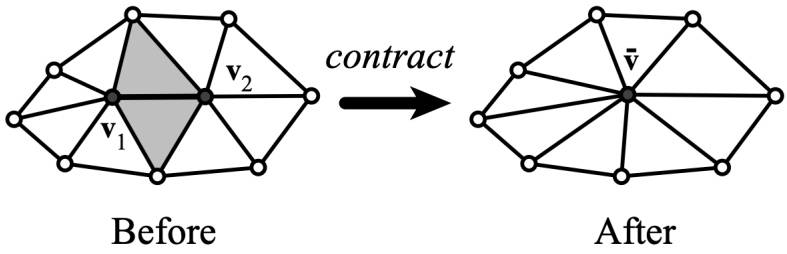


Figure 1: Contraction of the edge (v_1, v_2) into a single point. The shaded triangles become degenerate and are removed during the contraction.

2.1.2 Algorithm

The fundamental operation of the algorithm is edge contraction, written as $(v_1, v_2) \mapsto v$. The algorithm has three steps:

1. Move vertices v_1 and v_2 to $\bar{v}$.
2. Replace all occurrences of v_2 with v_1.
3. Delete v_2 and any degenerate faces.

2.2 Discrete Conformal Harmonic Map[5]

2.2.1 Description

The facial surface is approximated by simplicial complex, namely triangle mesh M. A vertex is denoted as v_i; an oriented edge (halfedge) $[v_i, v_j]$, where v_i is the source, v_j is the target; an oriented face is $[v_i, v_j, v_k]$, where v_i, v_j, and v_k are sorted counter-clock-wisely. function $f : M \mapsto R$ is defined on vertices. Suppose p is a point inside a triangle $[v_i, v_j, v_k]$, then p can be represented as the linear combination of three vertices, $p = \lambda_i v_i + \lambda_j v_j + \lambda_k v_k$, where $(\lambda_i, \lambda_j, \lambda_k)$ are *barycentric coordinates* of p, where $\lambda_i + \lambda_j + \lambda_k = 1$. Barycentric coordinates can be computed as follows: $\lambda_i = \frac{|[p,v_j,v_k]|}{|[v_i,v_j,v_k]|}$ where $|\cdot|$ represents the signed area of the oriented triangle. Then the function f becomes a piecewise linear function:

$$f(p) = \lambda_i f(v_i) + \lambda_j f(v_j) + \lambda_k f(v_k) \tag{1}$$

By direct computation, it is possible to directly compute the harmonic energy

$$E(f) = \sum_{v_i, v_j \in M} w_{ij}(f(v_i) - f(v_j))^2 \tag{2}$$

2.2.2 Useful equations

Image Boundary:

$$IB(v_i) = (cos(\theta_i), sin(\theta_i)), \theta_i = 2\pi \frac{\sum_{i=1}^{l} |v_l - v_{l-1}|}{\sum_{k=1}^{n} |v_k - v_{k-1}|} \tag{3}$$

Cotangent Edge Weight:

$$w_{ij} = \begin{cases} \frac{1}{2}(cot(\theta_k) + cot(\theta_l)) & \text{if } [v_i, v_j] \text{ is adjacent to 2 faces} \\ \frac{1}{2}(cot(\theta_k) & \text{if } [v_i, v_j] \text{ is on the boundary.} \end{cases} \tag{4}$$

Harmonic Energy:

$$E(f) = \sum_{v_i, v_j \in M} w_{ij}(f(v_i) - f(v_j))^2 \tag{5}$$

2.2.3 Algorithm

Algorithm 1 Conformal Harmonic Map - Discrete $\mathbb{R}^3 \mapsto \mathbb{R}^2$ [5]

1. Trace the mesh boundary and get a sequence of vertices $[v_0, ..., v_{n-1}]$.
2. Set the image boundary vertex v_i with the function $IB(v_i)$.
3. For all edges $[v_i, v_j]$ compute the cotangent edge weight w_{ij} using the above formula.
4. For all inner vertices v_k go through all the surrounding vertices v_j and construct the discrete Laplace equation, $E(f) = 0$.
5. Solve the Linear System.

2.3 Möbius Transformation

2.3.1 Description

There is a unique Möbius transformation taking any three distinct points of $\mathbb{C}+$, z_1, z_2, z_3, to any three distinct points of $\mathbb{C}+$, w_1, w_2, w_3 such that $z_i \mapsto w_i$. We use z_1 and z_2, designated as the left and right eye corner points, respectively, to construct the angle θ using the arctan, then use θ to compute the transformation of each point.

2.3.2 Algorithm

Let $z, origin \in \mathbb{C}+$, $\theta \in [0, 360)$ then the transformation $T(z)$ can be described by the equation

$$T(z, origin, \theta) = e^{(\theta i)} \cdot \frac{z - origin}{1 - (conj(origin) \cdot z)} \tag{6}$$

3 Methodology

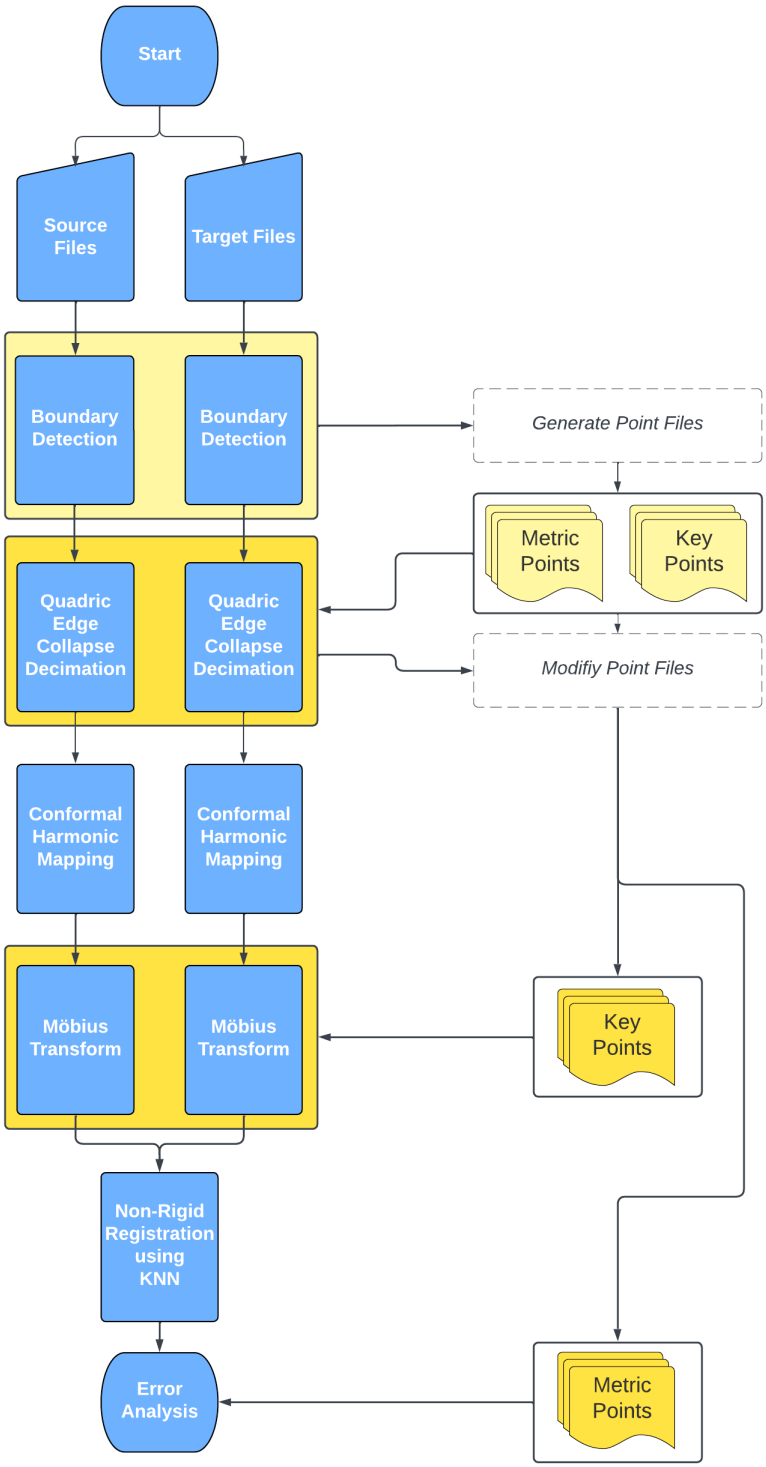
Methodology

To calculate the non-rigid changes to the face we use a conformal harmonic map to transform the input mesh $M \in \mathbb{R}^3$ into a surface on the unit disc $D_0(0) \in \mathbb{R}^2$ $x, y \in [-1, 1]$ and evaluate changes in $\mathbb{R}^2$. On the next page we illustrate this process with a flow chart.

Algorithm 2 Non-Rigid Registration Pipeline

1. Face detection with MediaPipe
2. Quadric Edge Collapse Decimation
3. Conformal Harmonic Mapping
4. Möbius Transformation
5. K-Nearest Neighbors

Figure 2: Non-Rigid Registration with Deep Learning and Conformal Harmonic Maps



3.1 Data Acquisition

3.1.1 Subject Description

Subjects are pediatric patients being treated with OME which has a side effect of changing how the surface of the face appears. To protect their confidentiality, we are only using my scan in the presentation of results.

3.1.2 3D Camera Description

The 3D scanning system is based on multiple wavelength phase shifting structured light methods. The sinusoidal lighting patterns are precisely controlled by the Digital Mirror Device (DMD) and projected onto the physical objects. The high speed digital cameras capture the raw images with high frame rate. For each pixel on the image, the relative phase information can be decoded directly by solving equation groups with trigonometric functions. The relative phase equals the absolute phase modulo 2π. In order to recover the absolute phase, an integer optimization process is required to find the phase periods. This can be achieved by using multiple wavelength fringe patterns, such that the synthesized wavelength is long enough to cover the whole scanning volume. The depth information is proportional to the absolute phase. Furthermore, a color camera captures the rich texture of the human skin. By camera calibration, we can determine the relative relation between the color camera and the gray camera, then we can compute the texture coordinates

3.1.3 3D Image Capture Process

Each subject was scanned twice. Both scans are captured with the subjects seated, facing the camera against a black background. The subjects are scanned a few weeks apart. The shortest difference between scan times was 14 days and the longest was 87 days, see Appendix A for more details.

3.1.4 Technical Specifications

Each 3D camera scan returns a pair of files - a WaveFront object file (.obj) and a Portable Network Graphic image file (.png). The WaveFront objects are formatted as follows:

```
v x y z r g b
v ...
...
vt x y
vt ...
...
f v/vt v/vt v/vt
f ...
...
```

For each vertex point, denoted by **v** there is a corresponding texture point **vt** such that v_i and vt_i refer to the same point. This relationship allows us to run computations on the 2D image and infer computation on the 3D object.

3.2 Boundary Detection with MediaPipe [11]

3.2.1 Boundary Detection

We evaluated the boundary of the simplicial complex mesh using an approximate solution with a facial detection model called MediaPipe (MP), an open-source machine learning pipeline for tracking landmarks in either still or streamed images. MP uses a 3D canonical face model comprised of 468 landmarks that capture a face's key points. The landmark locations on the canonical image as well as their semantic grouping are both published on GitHub under the Apache License. In our experiments, we constructed two sets of these landmarks to form a facial boundary. We ran boundary detection which returned an image with a red line, BGR $(0, 0, 255)$ that outlined the face. A version of this boundary is displayed in Figure 3 below, with the boundary thickness augmented to improve visibility.

Figure 3: Detected Facial Boundary

We used NumPy[9] to identify the respective height and width indices where the pixel color equaled the boundary color. We then passed the input image and the indices to OpenCV[1] where we used the **.fillPoly()** method to construct a binary mask of the image (see Figure 4).

Figure 4: Binary Mask of Face

The binary mask is represented as a height by width by depth matrix, with the depth being the BGR colors such that the matrix dimensions are always $(H, W, 3)$. We know from our data inputs that each pixel in the image corresponds to a texture vt line in the **.obj** file. We read in the texture lines from the **.obj** file and scale them up to the dimensions of the matrix. The resulting texture image gives a binary impression of the shape of the face, as depicted in Figure 5(a) below. When we combine the binary mask and the texture image we return a constrained texture image as in Figure 5(b).

(a) Unconstrained (b) Constrained

Figure 5: Texture Image

From the data input section, for each texture point vt_i there is a corresponding vertex v_i. We use the constrained face to construct a list of indices. For each set of texture coordinates (y, x), if the pixel at location constrained texture y, x is red, BGR $(0, 0, 255)$, the texture index is added to the list. Below is a quick code snippet illustrating the idea.

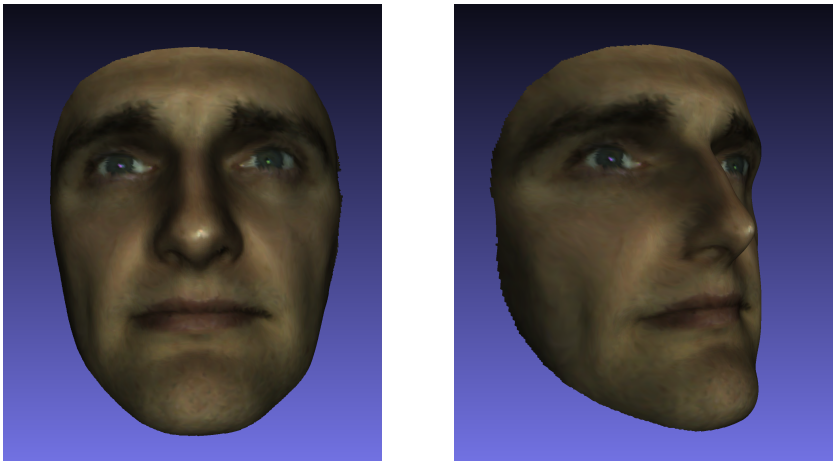
```
indices = []
for idx, (y, x) in textures:
    if constrained_face[y,x] == (0, 0, 255)  # Red
        indices.append(idx)
```

We then use the indices list to filter the vertices in the object. However, this adds complexity when writing back the object for the faces. Faces in the input object are of the form `f v1/vt1 v2/vt2 v3/vt3` which means they are highly dependent on vertex order. To account for this, we constructed a hash map that mapped the index of the unfiltered object to the index of the filtered object. For example, if there are 10 vertices in the unfiltered object but we filtered the object down to use only vertices 4, 5, and 6, the mapping would be as follows:

$$\begin{Bmatrix} 4 & \mapsto & 1 \\ 5 & \mapsto & 2 \\ 6 & \mapsto & 3 \end{Bmatrix} \tag{7}$$

We then use this mapping when writing faces back to the object. First we translate the unfiltered vertex indices to a filtered vertex. Then, if all filtered vertices are in the new mesh,

we write the face to the **.obj** file.
In summary, we use MP to detect the boundary image indices and build a mask. We translate these filtered indices to texture indices from the **.obj** file. We then map the unfiltered texture indices to filtered vertex indices. Once we obtain all this information, we can construct a filtered **.obj** file. Below we display the filtered file:



(a) Front View (b) Side View

Figure 6: Collapsed Source Object

3.2.2 Inconsistent Boundary Construction

To establish the importance of a consistent boundary in 3D facial scan registration, we decided to build inconsistent boundaries, which we also call "boundary methods" to test against the consistent boundary. We elected to build six inconsistent boundaries per consistent boundary for a total of twenty four inconsistent boundaries. We removed chunks from the left part of the forehead, right part of the forehead, center of the forehead, left cheek, right cheek, and chin. The chunks varied in size and shape because our main focus was to remove some part of the face to make it inconsistent. We built the boundaries the same way we built the consistent boundaries with a few added constraints. First, we made sure not to use any keypoints or metric points when removing chunks. Second, we made sure at least one edge of the chunk was also an edge in the corresponding boundary. Finally, we kept the number of vertices below 15 in all cases since we built these manually. Below are the resulting masks of each chunk and its corresponding inconsistent boundary mask. We will discuss how these inconsistent boundaries affected the final results in the next full section (4).

Figure 7: Initial Mask

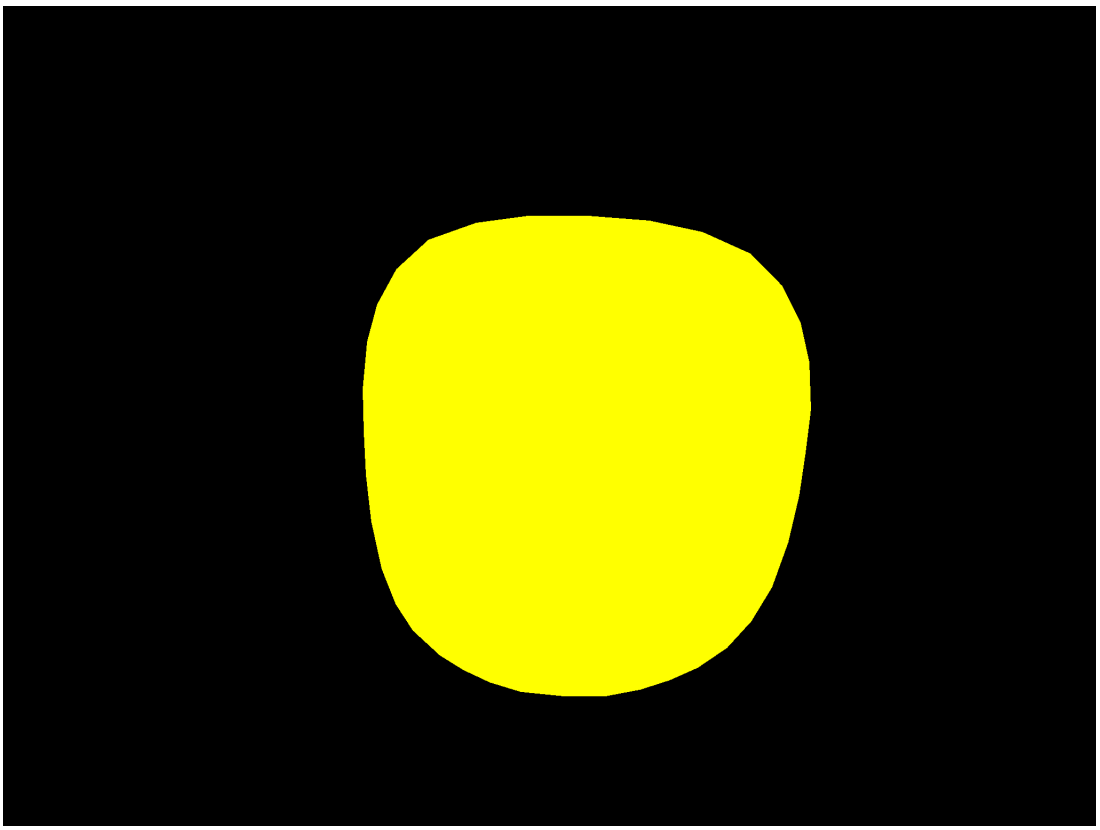
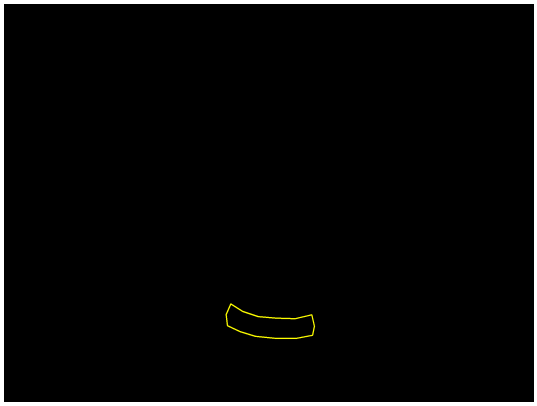


Figure 8: Chin

(a) Chunk



(b) Inconsistent Boundary

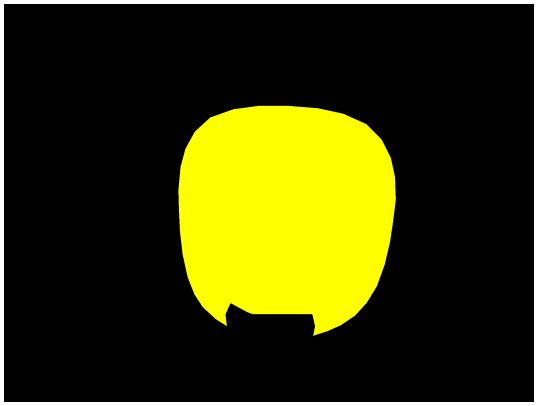


Figure 9: Left Cheek

(a) Chunk

(b) Inconsistent Boundary

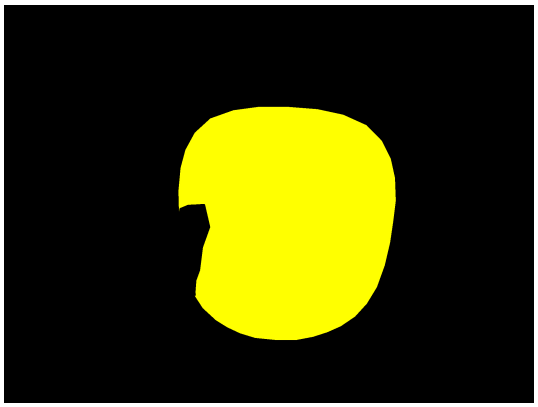


Figure 10: Right Cheek

(a) Chunk

(b) Inconsistent Boundary

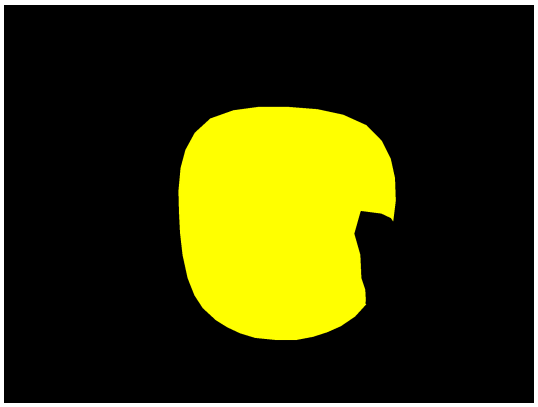
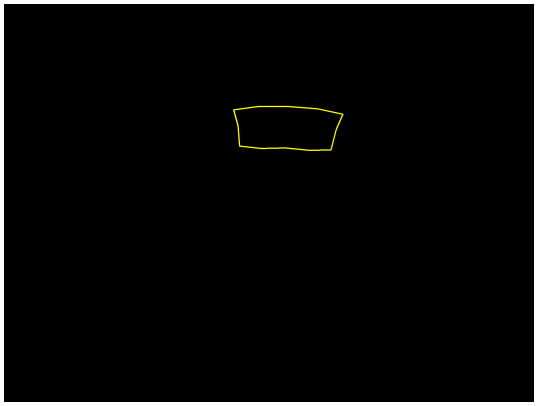


Figure 11: Center Forehead

(a) Chunk



(b) Inconsistent Boundary

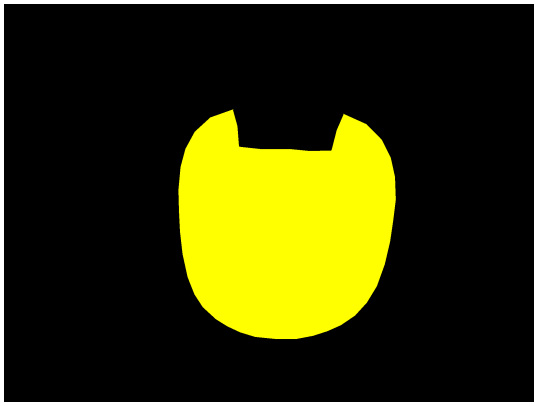
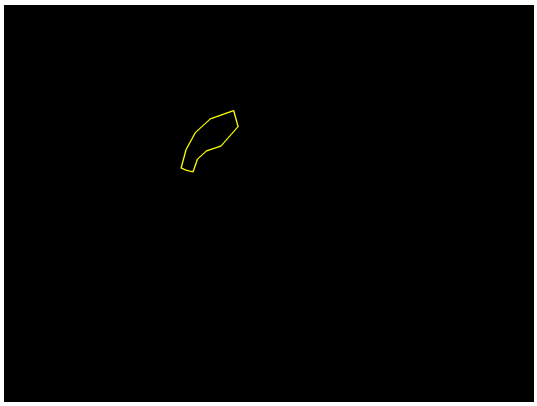


Figure 12: Left Forehead

(a) Chunk



(b) Inconsistent Boundary

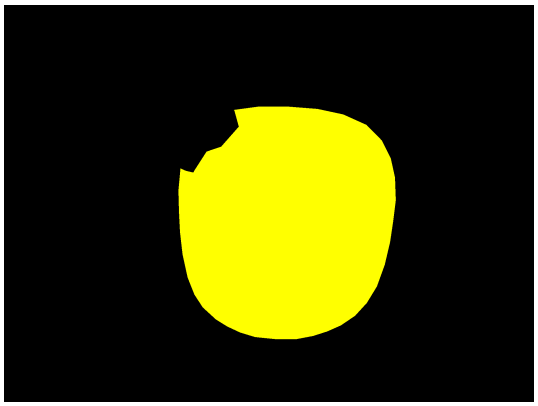
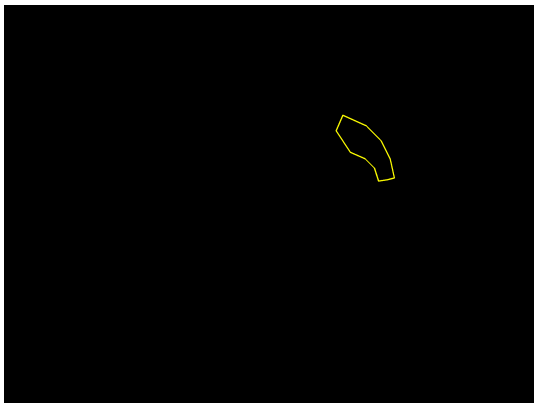
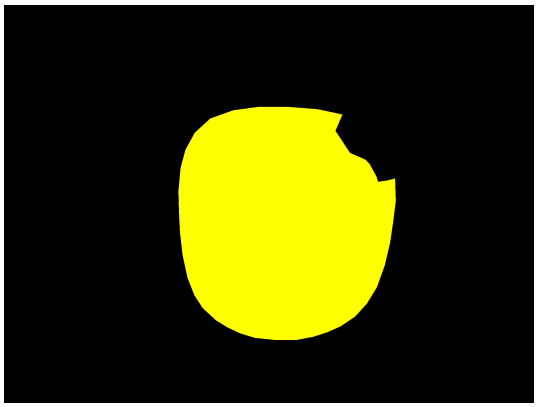


Figure 13: Right Forehead

(a) Chunk



(b) Inconsistent Boundary



3.2.3 Keypoint Detection

MediaPipe builds this face boundary with 468 3D face landmarks [11]. We use these landmarks to compute the Möbius Transformation in step 4 of the pipeline as well as the analysis post-registration. For the Möbius transformation, we capture three points: the left eye corner, right eye corner, and nosetip, as pictured in Figure 14. We will go into more detail why those points were chosen in section 3.5, Möbius Transformation.

Figure 14: Detected Möbius Keypoints

This method is repeated for any points used for evaluating metrics downstream. We talk more about these points' usage in sections 3.3, 3.5, and 4.

3.3 Quadric Edge Collapse Decimation [4]

We introduced QECD because the simplicial complex in our use case is too dense to run computation algorithms in a reasonable time. This was an operation we previously executed using MeshLab[3] to collapse the vertices. We used the software's associated Python API, PyMeshLab[12], to automate the operation. It is important to note that we convert the key points calculated during boundary detection from a vertex value to an integer index value. First we load the mesh and the key points, then we compute QECD. After completing the operation, we use the key point vertex coordinates to select the closest vertex and return the vertex's index.

Figure 15: QECD-Collapsed Source Data

(a) Front View

(b) Side View

3.4 Conformal Harmonic Mapping

We used the algorithm described in section 2.2.3 to convert the collapsed mesh into a unit disc and then compute a conformal harmonic map on the disc.

Figure 16: Surface After Conformal Harmonic Mapping

3.5 Möbius Transformation

In order to compute non-rigid registration more accurately, we naïvely align the two unit discs D_0 using a Möbius transformation. We use the calculated key points from MediaPipe as our initial z_1, z_2, and z_3, which represent the left eye corner, right eye corner, and nose tip, respectively. Our assumption is that by aligning these points we will align the faces and minimize registration error.

This step is important for checking the predicted boundary from the first step in the algorithm is correct. If the transformation leads to a face that looks more warped than the input, it is likely this boundary is not consistent. This helped us identify issues with capturing shadows on the face and choose different boundary landmarks to outline the face. In figure 17 below, we can see that both the source (a) and target (b) images are aligned similarly. In the case of (a), the face is more centered, with the nosetip closer to the origin than in figure 16.

Figure 17: Möbius Transformed Surface

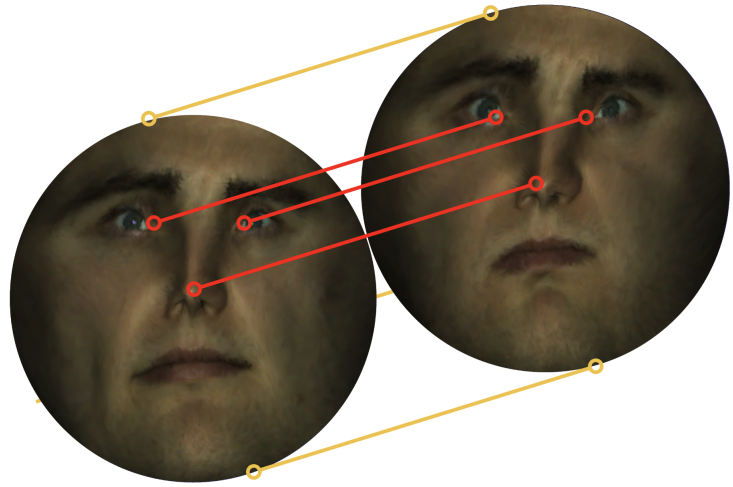
(a) Source

(b) Target

3.6 Non-Rigid Registration with KNN

We approached registration two different ways. The first is to map each vertex on the source disc to the nearest corresponding point on the target disc. The second is to choose specific metric points (i.e. corners of the mouth, cheeks, chin) on each disc and measure how the points changed. We use the Euclidean distance metric in $\mathbb{R}^2$ to compute both these metrics. We write out a map of the vertex indices to a text file of the form $source_i \mapsto target_j$, with $i \in Vertices_{source}$ and $j \in Vertices_{target}$. This allows us the option to understand the movements of corresponding points in $\mathbb{R}^3$ We also write out the distance between metric points to a file so they may be analyzed for future use.

Figure 18: Non-Rigid Registration



4 Analysis and Results

4.1 Challenges with Boundary Detection

In this section we discuss some of the challenges and obstacles we encountered on the way to achieving our results.

4.1.1 Faces behind Manifold

Another side effect of the boundary issues is related to re-mapping the filtered face. In some cases, we had an issue with mapping the input vertices to the output vertices. This issue led to triangle faces appearing behind the mesh, spanning the width of the face. We wrote a patch to alleviate the issue, only writing out faces if their areas are less than a constant. As expected, this check significantly increased run-time, nearly doubling it. However, the issue typically resolved itself when we constrained the boundary closer to the center of the face. Our use case involves changes closer to the center of the face which allowed us to adjust the boundary accordingly.

4.1.2 Shadows

We initially used the boundary silhouette as defined by MediaPipe to constrain faces before we ran into issues with shadows on patients' faces. These shadows led to warped faces on the unit discs, boundary shadows being weighted heavier than facial features, and issues with aligning the unit discs with a Möbius transformation. All of these are symptoms of an inconsistent boundary.

Another thing to consider is the subjects' skin color. There was some variety between subjects' tone and base that may make some subjects harder to identify with boundary detection. This should be front of mind when selecting deep learning models for facial detection.

4.1.3 Illumination

Another issue is that some of the subjects wore jewelry, specifically earrings. This affected the mask that we used to filter the boundary of the face and led to computational issues. We changed the filter to use a solely red filter, BGR $(0, 0, 255)$, which solved the problem.

4.2 Results

4.2.1 Boundary Detection Efficacy

We built four different facial boundaries using MP's landmarks, which we named *outer*, *middle*, *inner*, and *custom* and refer to collectively as "boundary types". we elaborate on their boundary construction in Appendix B. In the table below, we display the results of running our pipeline on each patient with each boundary we constructed. We also briefly

describe the different outcomes in the key below. The extended descriptions for this key are in Appendix C.

Key:	
Acronym	Description
A - Best	Unit Disc aligned
FL	Unit Disc flipped 180°
MA	Misaligned
MS	Unit Disc Majority Shadow
WF	Warped Unit Disc
FBF	Face Behind Manifold
SB - Worst	Unit Disc straight boundary

Patient No.	**Outer**	**Middle**	**Inner**	**Custom**
4	FBF, SB	A	A	A
5	FBF	SB, MS	A	FL
9	FBF	MA	A	A
11	FBF	MS	A	FL
12	FBF	FBF, WF	WF	MA
13	FBF	MA	A	A
14	A	A	MA	A
16	A	ma*	ma*	ma*
18	FBF	MS, WF	FL	FL
30	FBF	FBF, WF	A	A

It is worth noting that patient 16 was very closely misaligned, rotated by about 10°.
This table shows that different boundaries have different effects on the conformal mapping and therefore the non-rigid registration and metric accuracy. Of the ten patients, we were able to align the transformation of their faces using this pipeline in eight of them, using at least one of the boundaries. Another result that is important to note is the runtime. We outline a sample in Appendix D. As result, this method can be used currently to reduce and in some cases eliminate the manual effort of constraining a boundary from these scans.

4.2.2 Importance of Consistent Boundaries

Our goal was to see if the consistent boundary registered the facial scans' conformally mapped surfaces better than inconsistent boundaries. For each boundary type (outer, middle, inner, custom), we performed registration with the consistent boundary plus all six inconsistent boundaries. To measure registration error, we chose five metric points relevant to OME and calculated the distance between their location on the source and target surfaces, taking the sum of the total error. We repeated this for each of the ten patients, resulting in a matrix $\underset{4\times10\times7}{M}$.

Given the small sample size of our patients and testing that our error for consistent boundary was always less than the each inconsistent boundary, we chose a one-sided t-test.

We ran the test pairwise with the consistent boundary (cb) as one parameter and each inconsistent boundary (IB) as the other.

$$\forall ib \in IB. \tag{8}$$

$$cb.size, ib.size \leq 10 \tag{9}$$

$$pvalue = ttest(cb, ib) \tag{10}$$

$$\alpha \geq pvalue \; ? \; reject \; h_0 \; : \; do \; not \; reject \; h_0 \tag{11}$$

We built the results into a heatmap, labeled by boundary type and inconsistent boundary used.

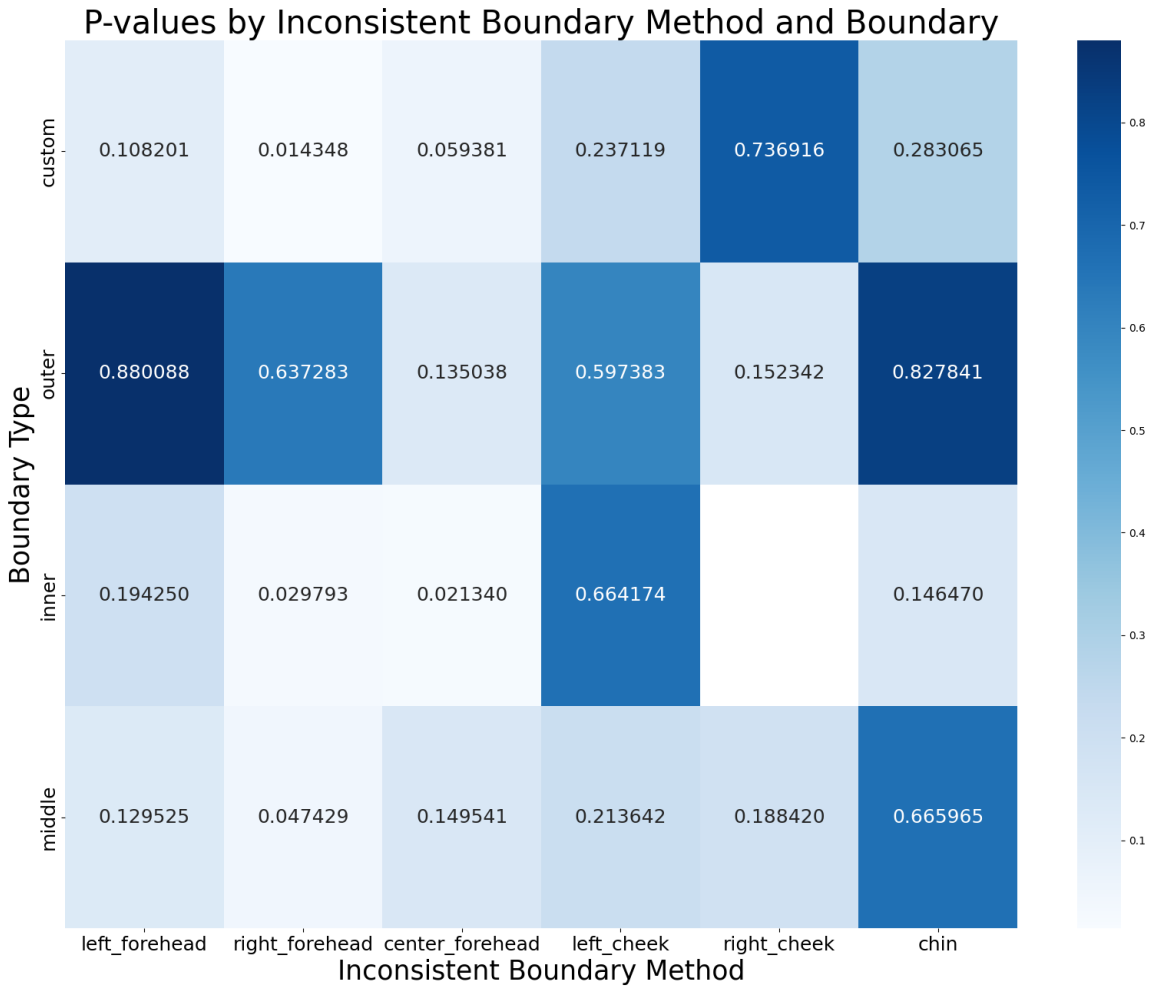


Figure 19: p-value Heatmap

The test yielded ambiguous results, failing to decisively accept or reject the null hypothesis in all cases. In some cases, the p-values are small enough to reject the null hypothesis

h_0 when compared to a standard α value of 0.05. However, higher values of α such as 0.10 or 0.20 should be given some consideration given the size of the samples. Alternatively, certain combinations of boundary types and boundary methods are comfortably within the null hypothesis. An outlier exists with the *inner* boundary type with the *right cheek* boundary method. No p-value was calculated. This occurs when the number of samples is too low, which means there were issues upstream calculating this boundary. Likely, the boundary method was built incorrectly, which led to the harmonic map step skipping the mapping altogether, resulting in no metric points downstream. This conjecture leads me to my final point - there is a lightly noticeably trend with the heatmap. The *inner* consistent boundary leaves the smallest area of the boundary types, the *middle* the second smallest, *custom* third smallest, and *outer* the largest. Any additional masking would affect the registration proportionately. Ignoring the NaN value in the *inner* row for a moment, all but one of *inner*'s five p-values are less than 0.20, arguably a good alpha given the sample size. The trend continues with *middle* having all but two p-values less than 0.20, custom has three of six, and *outer* has four of six less than 0.20. This method offers both tentative support and insufficient evidence to reject the null hypothesis that a consistent boundary returns a better registration than an inconsistent boundary.

5 Conclusion

Conclusion and Future Work

For future researchers interested in this type of problem, we suggest looking at several opportunities. The first is to investigate more thoroughly the definition of a consistent boundary. If that can be more clearly mathematically established, it may be possible to solve for the boundary programmatically. Another opportunity would be to apply alternative methods to conformally map the manifolds to a surface to improve the runtime. Our focus of this work was to use computer vision techniques to aid with discrete differential geometry techniques; advanced methods that could improve runtime exist[10]. We used a deep learning facial model to constrain a simplicial complex to perform geometric algorithms upon. Our work serves to demonstrate the effectiveness of the combination of these techniques.

Our results did not conclusively prove consistent boundaries are better than inconsistent boundaries for non-rigid registration. They do suggest there a potential monotonic relationship between the area of the constrained boundary and the size of the chunk removed. A logical next step would be to research consistent boundaries compared to other boundary methods and more conclusively quantify the effects of changes to the boundary.

References

[1] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[2] G. P. Choi, D. Qiu, and L. M. Lui. Shape analysis via inconsistent surface registration. *Proceedings of the Royal Society A*, 476(2242):20200147, 2020.

[3] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia. MeshLab: an Open-Source Mesh Processing Tool. In V. Scarano, R. D. Chiara, and U. Erra, editors, *Eurographics Italian Chapter Conference*. The Eurographics Association, 2008. ISBN 978-3-905673-68-5. doi: 10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136.

[4] M. Garland and P. Heckbert. Simplifying surfaces with color and texture using quadric error metrics. In *Proceedings Visualization '98 (Cat. No.98CB36276)*, pages 263–269, 1998. doi: 10.1109/VISUAL.1998.745312.

[5] X. Gu. Harmonic Map Tutorial, Oct. 2013. URL `https://www3.cs.stonybrook.edu/ gu/tutorial/HarmonicMap.html`.

[6] X. Gu and B. C. Vemuri. Matching 3d shapes using 2d conformal representations. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2004: 7th International Conference, Saint-Malo, France, September 26-29, 2004. Proceedings, Part I 7*, pages 771–780. Springer, 2004.

[7] X. Gu and S.-T. Yau. Computing conformal structure of surfaces. *arXiv preprint cs/0212043*, 2002.

[8] X. Gu, Y. Wang, T. F. Chan, P. M. Thompson, and S.-T. Yau. Genus zero surface conformal mapping and its application to brain surface mapping. *IEEE transactions on medical imaging*, 23(8):949–958, 2004.

[9] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, Sept. 2020. doi: 10.1038/s41586-020-2649-2. URL `https://doi.org/10.1038/s41586-020-2649-2`.

[10] R. Lai, Z. Wen, W. Yin, X. Gu, and L. Lui. Fast and robust algorithms for harmonic energy minimization on genus-0 surfaces. *UCLA CAM Reports*, 60:2, 2011.

[11] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. Yong, J. Lee, et al. Mediapipe: A framework for perceiving and processing reality. In *Third Workshop on Computer Vision for AR/VR at IEEE Computer Vision and Pattern Recognition (CVPR)*, volume 2019, 2019.

[12] A. Muntoni and P. Cignoni. PyMeshLab, Jan. 2021.

[13] K. Qian, K. Su, J. Zhang, and Y. Li. A 3d face registration algorithm based on conformal mapping. *Concurrency and Computation: Practice and Experience*, 30(22):e4654, 2018. doi: https://doi.org/10.1002/cpe.4654. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.4654`. e4654 cpe.4654.

[14] S. Wang, Y. Wang, M. Jin, X. D. Gu, and D. Samaras. Conformal geometry and its applications on 3d shape matching, recognition, and stitching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(7):1209–1220, 2007.

[15] Y. Wang, M. Gupta, S. Zhang, S. Wang, X. Gu, D. Samaras, and P. Huang. High resolution tracking of non-rigid 3d motion of densely sampled data using harmonic maps. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 1, pages 388–395. IEEE, 2005.

6 Appendices

Appendices

A OME Duration

Patient No.	Days Between Visits
4	77
5	14
9	21
11	87
12	57
13	50
14	78
16	22
18	43
30	43

B Boundary Landmark IDs

Boundary name	Description	Ordered Landmark IDs
Silhouette	Landmarks detailing the outer most boundary of the face	10, 338, 297, 332, 284, 251, 389, 356, 454, 323, 361, 288, 397, 365, 379, 378, 176, 149, 150, 136, 172, 58, 132, 93, 234, 127, 162, 21, 54, 103, 67, 109
Second	Landmarks detailing the second most outside boundary of the face	151, 108, 69, 104, 68, 71, 139, 34, 227, 137, 177, 215, 138, 135, 169, 170, 140, 171, 175, 396, 369, 395, 394, 364, 367, 435, 401, 366, 447, 264, 368, 301, 298, 333, 299, 337
Third	Landmarks detailing the third most outside boundary of the face	9, 107, 66, 105, 63, 70, 156, 143, 116, 123, 147, 213, 192, 214, 210, 211, 32, 208, 199, 428, 262, 431, 430, 434, 416, 433, 376, 352, 345, 372, 383, 300, 293, 334, 296, 336
Custom	Landmarks detailing a boundary of the face that is a combination of landmarks from the Second and Third boundaries	151, 108, 69, 104, 68, 156, 143, 116, 123, 147, 213, 138, 135, 169, 170, 140, 171, 175, 396, 369, 395, 394, 364, 367, 433, 376, 352, 345, 372, 301, 298, 333, 299, 337

C Boundary Detection Results

I cannot publish several of the below states because they were only present in pediatric patients. To protect their confidentiality, I am only using my scan in the thesis.

Best - Unit Disc Aligned	The ideal outcome of the pipeline. After the Möbius transformation, the discs are aligned with the nose tip at the origin and the inside corner of each eye in a horizontal line above the eyes.
Unit Disc flipped 180°	The same as Aligned, except either the source or the target unit disc is rotated 180 ° such that the eyes are below the nose tip.
Misaligned	The source and target unit discs are misaligned by less than 180°.
Unit Disc Majority Shadow	One or both of the unit discs is blacked or grayed out due to the amount of shadows in the initial scan.
Warped Unit Disc	One or both of the unit discs are warped such that the human facial features are compressed and twisted.
Face Behind Manifold	After running boundary detection, there are triangular faces spanning the width of the manifold present behind the human face.
Worst - Unit Disc straight boundary	The resulting unit disc from the conformal mapping has straight edges around part of the boundary rather than a consistent circle.

D Sample Runtime Table

This table gives the reader an idea of runtime of all 24 combinations of boundary types and boundary methods for a single individual.

Pipeline Step	Average Runtime (seconds)	Total Runtime (seconds)
Boundary Detection	9.18458	220.43
QECD	5.20750	124.98
Conformal Harmonic Map	33.76708	810.41
Möbius Transform	0.97292	23.35
Non-Rigid Registration	11.44292	274.63
Total	**60.575**	**1453.8**