Validating and Improving Single-Stage Cranioplasty Prosthetics

An expansion of single-stage cranial defect repairs and implants, Computer Integrated Surgery II, Spring 2016

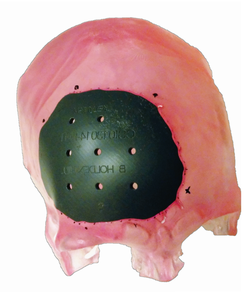
Team : Erica Schwarz and Willis Wang Mentors: Mehran Armand, Chad Gordon, Ryan Murphy

Introduction

- Using 3D scanned data to create a machinable model is an accurate, time-effective method of creating cranial implants. In order to do this, the skull defefct must be segmented fromt he scan and registerd to patient space
- We validated a segmentation method by using realistic skull defect geometries
- We developed a registration algorithm that was robust to initial data pose
- We created Matlab and Python implementations of our method
- We created a Slicer extension that performs segmentation and registration on selected models

Problem

- Cranioplasties are used to reconstruct the site of craniotomies and other cranial surgeries that remove sections of the skull.
- Due to risk of infection after such a procedure, creating a well-fitting prosthetic is important for increasing quality of life and risk management.
- Though improvements in creating personalized prosthesis have been made, time and accuracy are still significant issues that need to be addressed



- Last year, a team created a method for segmenting the defect from a 3D scan of the skull. However, there was no validation on realistic models and the 3D data was not registered to patient space.

Solution

Segmentation

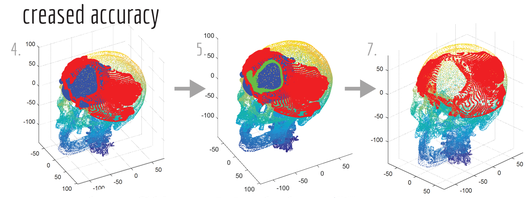
Validated with realistic geometries

- Created skulls with simulated defect
- Remeshed model to have same point density as scan
- Introduced smoothing and noise that simulated scanned data (average error = 1mm)
- Ran segmentation method to see if it could capture complete ring
- Repeated with varying bevels ranging from 45-90 degrees with bevel range ranging from 0-10 degrees

Registration

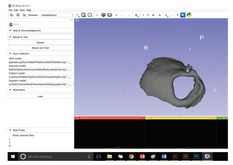
Registration algorithm robust to initial pose

1. Imported skull model, defect scan, and implant model
2. Removed wall from defect scan by using nearest neighbors
3. Fit sphere to defect scan and skull model using modified RANSAC
4. Aligned centers of fitted spheres to ensure concentricity and provide convenient rotation and translation space
5. Aligned centroid of defect and implant using rotation as initial guess of defect location
6. Iteratively perturbed initial pose of defect around implant space while performing ICP to find minimum within probable location
7. Final pose was found using a point-to-surface registration for increased accuracy



Exported to Slicer Module

- Rewrote algorithm as Python functions
- Converted registration method into VTK implementation
- Created Slicer module user interface

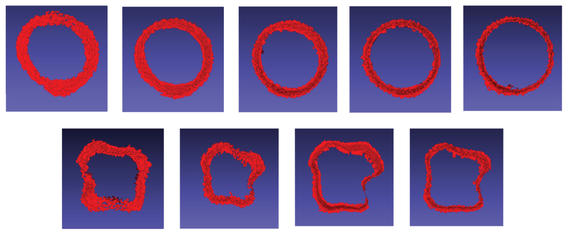


Results and Future Work

Segmentation

Results

- The segmentation method created a complete ring with all tested geometries.
- This posits enough information to create pose data for later machining



Future Work

- Complete work on program that automatically calculates machine pose and path around defect segmentation

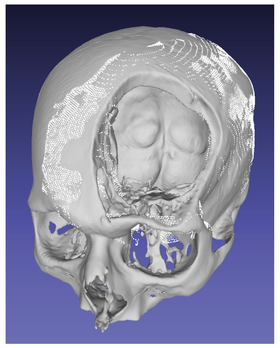
Registration

Results

- Algorithm outputted same final pose regardless of initial pose
- Error between defect registration and ground-truth defect pose had an average 0.9mm. This was within the error propagated by the 3D scanner.
- This validated that the algorithm is robust and accurate

Future Work

- Create more efficient implementation that takes less time to run (this might include making a C++ Slicer module instead of a Python one)
- Test with cadaver data
- Register implant to fabrication machine in order to create a fitted implant



Lessons Learned

- Point cloud registration with featureless and somewhat noisy data is highly susceptible to local minima
- Because of the wide berth of 3D model storage types and formats, it is important to establish what the workflow will be early on
- Though testing algorithms in high-level languages is useful for rapid development of modules, you must be aware of what functionality exists in the final programming language you intend to use
- It is extremely important to leave good documentation not just about the functionality, but also about the dependencies and file types
- Always have a back up plan in case dependencies fall through

Credit

- Erica Schwarz - Registration development, segmentation validation, slicer integration
- Willis Wang - Registration development, perturbation module, and testing

Acknowledgments

- Thanks to Ryan Murphy for helping us become familiar with the necessary programs and hardware and for giving feedback throughout our project
- Thanks to Alex Mathews and Joshua You for allowing us to use their initial work on this project as well as for their help in learning the program architecture
- Additional thanks to Mehran Armand for providing resources and mentorship, and Chad Gordon for providing cadaver models and CT data.