Improving Single-Stage Cranioplasty Prosthetics:

An expansion of single-stage cranial defect repairs and implants

Final Report

Project 12

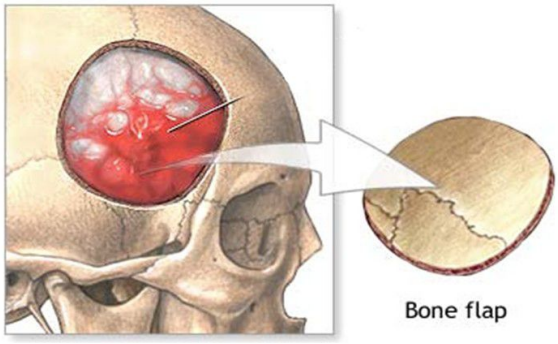
Members: Erica Schwarz, Willis Wang

Mentors: Dr. Mehran Armand, Dr. Chad Gordon, Dr. Ryan Murphy

Project Summary

Summary/Abstract

During some cranial surgeries, pieces of the skull must be removed in order to gain access to the brain. Directly following these surgeries, cranioplasty procedures are used to repair defects in the skull. However, in many cases the original bone flap is damaged during removal from the skull is unable to be replaced. In these cases, a cranial implant must be made to cover the exposed defect. A implant that perfectly conforms to the defect shape is ideal as it prevents "dead space" in the skull and associated infection. However, creating a fitted implant presents its own challenges. Using a machine to create one requires the patient to come back at a separate time creating a two-stage surgery which during the interim the patient does not have an ideal implant. To prevent the necessity of two-stage surgery, many reconstructive surgeons will have an oversized implant of the expected defect area machined beforehand and then manually carve it to be the correct shape of the actual defect during the surgery. This is done by trial and error and takes a considerable amount of time (10 - 80 min). Recently Dr. Gordon, Dr. Armand, and Ryan Murphy developed a method that projects the implant outline onto the oversized implant which reduces this time significantly, but this method is limited by the complexity of implant. Last year, a new system was developed for using 3D scanner to create a machined single-stage implant, but the effectiveness of using 3D scanners and point cloud models to completely capture defect geometry and accurately register it back into the patient space is currently unknown. This project will create ground truth models of cranial defects to test and validate accuracy of the 3D scanning system. During this process we will refine and improve the 3D scanning system from implant capture to patient registration.



Background and Current Issues

Cranioplasties are used to reconstruct the site of craniotomies and other cranial surgeries that remove sections of the skull. These cranioplasties are also known as secondary cranial reconstructions and are performed for patients who require staged reconstruction after craniotomies. These craniotomy procedures involve the removal of a section of the skull. The resulting skull flap is often not suited for immediate replacement due to issues such as risk of infection or excess removed material. As a result, these skull flaps are often frozen or thrown away altogether and a cranioplasty is performed instead. The cranioplasty is usually performed to alleviate concerns of safety and protection, cosmetic appearance restoration, and treatment of issues associated with leaving a portion of the skull removed, but can carry its own risks. These

procedures are generally performed with an implant made of Poly-Methyl Methacrylate (PMMA) or a titanium mesh.

Due to risk of infection after such a procedure, creating a well-fitting prosthetic is important for increasing quality of life and risk management. Recently, an alternative method which involves the implementation of on-site fabrication of the prosthesis has been effective in cutting down on the number of separate surgeries performed. In this system, surgeons use a Customized Cranial Implant, or CCI, made of PMMA. These CCIs are fabricated preoperatively from patient CT scans and modified through Computer Aided Design. These CCIs are made as an oversized section of the operating area based on information from the CT scan. The main advantage of these CCIs is their ability to conform more closely to the unique curvature of the skull. Specifically, the thickness of the skull is taken into account when making these CCIs whereas a prosthetic made of titanium would be unable to achieve the same precision. During the surgery, the surgeon machines the CCI to match the size and shape of the defect. However, this is labor intensive and can take upwards of an hour.

Although this single-staged format is already a significant advancement from previously used multi-staged reconstruction, there is still room for improvement. In an effort to further improve procedure times, Dr. Gordon, Dr. Armand, and Ryan Murphy have devised a system which includes a Polaris optical tracker and a laser projection system. This system projects the trace onto the oversized CCI for more accurate cutting and shorter operation times. However, the system is not without its drawbacks. Specifically, it struggles with more complicated geometries and has difficulty collecting points describing the bevel angle of the defect. Additionally, the polaris system itself can be difficult to setup and is very expensive.

As 3D handheld scanners become cheaper and more accurate, the viability of replacing the polaris system with newer technology becomes more feasible. In the previous year, there was a group that built upon Dr. Gordon, Dr. Armand, and Ryan Murphy's system by incorporating a relatively inexpensive 3D scanner in the form of the Structure Sensor (an attachment for the iPad) as a cheaper and more effective alternative to the Polaris system. The project was generally a success, but was limited in that it did not incorporate defect bevels and more complicated geometries and also did not evaluate scan-to-patient registration accuracy. This project proposes to further develop this system with updated segmentation algorithms that allow for more complex feature detection and incorporate defect-to-patient registration in order to put the oversized CCI implant and the scanned defect in the same space (a necessary step for later implant fabrication). Will will do this using ground truth test cases that incorporate a variety of realistic defect geometries.

Procedure Outline

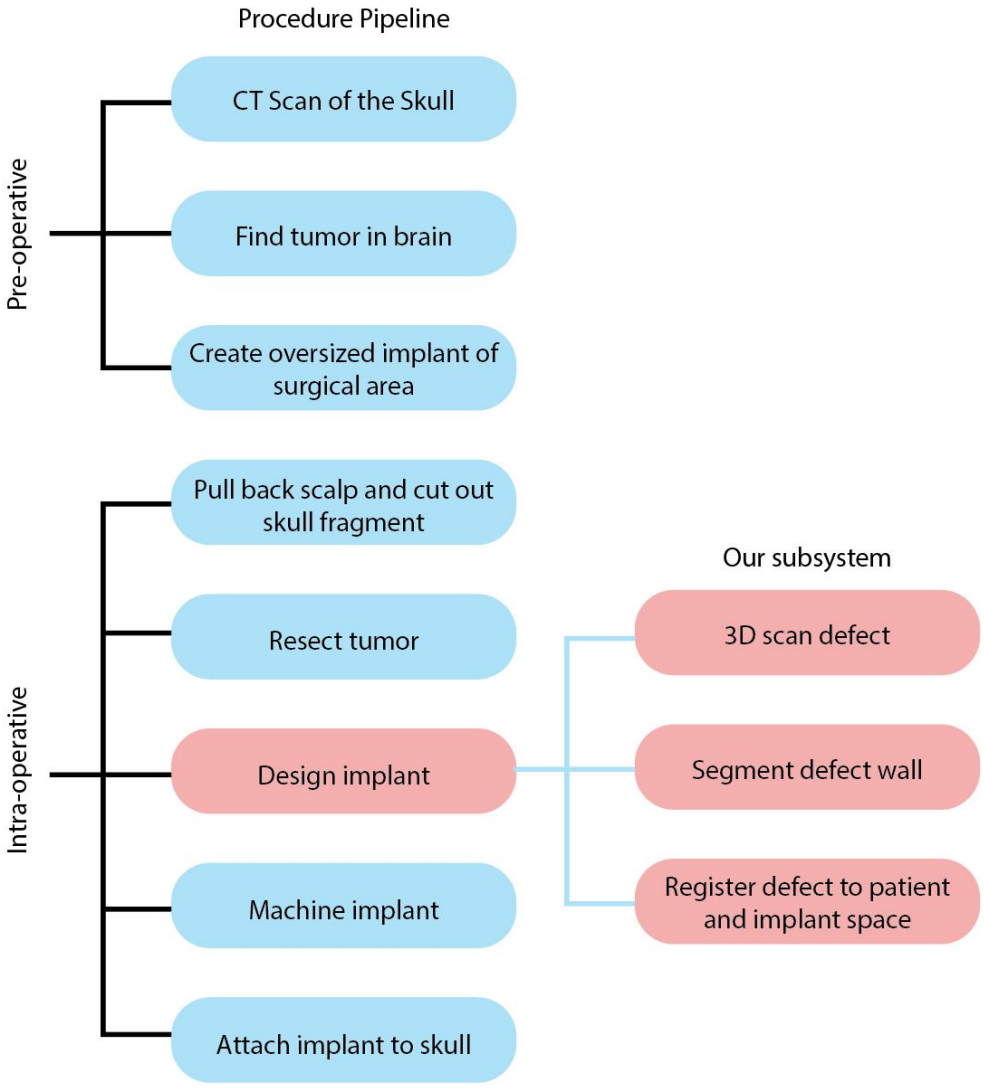
This projects proposes to further expand on the previous years project of integrating a new subsystem into the cranioplasty pipeline. Specifically, the subsystem includes the use of a

3D scanning hardware to render a mesh model of the defect. This model is then registered to the patient's CT scan and subsequently the CCI to allow for quick and accurate machining. This project will be focusing on extending the segmentation subroutines to allow for faster and more accurate parsing of the bevel angles.

The beginning workflow of the cranioplasty procedure remains greatly unchanged from the current standard of care. After the cranial defect is exposed, the operator will use a handheld, wireless device to take a 3D scan of the defect. In our project, we use the iPad mounted Structure Sensor. The iPad provides a convenient and intuitive interface, but other devices (such as the Kinect) with similar accuracies would work equally as well. For the Structure Sensor workflow, the optimum accuracy is obtained when scanning from within half a meter distance from the desired object. Scanning is done by pointing the scanner at the defect site and then slowly moving it around the site area until the Structure Sensor interface notifies the user that it has completed its scan.

After the scan is complete, the Structure Sensor can automatically send the model to an uplinked computer. From there, the operator will open up Slicer and load a module called "Defect Registration". This module asks for three models: the scanned data, the previously made patient skull model (produced from patient CT scan), and the CCI model (which is in the patient model space). The operator will then press "Apply" to run all necessary functions. The program will first segment the defect wall from the defect and then will transform both into patient space using the method outlined below. All of this is automatic and does not require any additional user input (though there are some parameters that can be changed if desired). After that, an algorithm with calculate the machining path necessary to cut the CCI down to match the defect wall in the implant space which will later be passed on to a laser cutter for final fabrication.

Procedure Pipeline



Procedure Evaluation

Our expansion of the 3D scanning system will be evaluated on the following metrics:

1. Accuracy
   We evaluated our accuracy using average point cloud to surface distance. Our registration method had an average model to ground truth error of 0.9mm after point to surface registration. Since the 3D scanner in this procedure is documented to have an accuracy of 1mm when placed within 0.5m of the scanning target, our registration is well within the error propagated inherently through the scanning process.

2. Robustness
   We were able to validate the robustness of the segmentation method by using ground truth models, and showed that it was able to segment a complete defect ring even when handling complex or irregular defects. We verified the robustness of the registration method by showing that the same final pose was obtained regardless of the initial pose of the defect scan.

3. Ease of Use
   When completed, this system should greatly increase ease of use compared to the current standard of care. There is no quantitative measurement for this procedure evaluation metric, but using an iPad and an automated implant process is easier to implement than the current standard of care which involves larger machines or trial-and-error cutting of the implant. In addition, our workflow is completed automated, requiring no user input other than taking the scan and selecting the necessary models.

4. Time
   This system will not require extensive equipment setup and would not require manual cutting of the implant. This will reduce time costs before, during, and after the surgery compared to current single-stage and double-stage cranioplasty workflows. However, currently our program runtime is ~10 minutes. Though this is still an improvement on the current method of implant design (which can take over an hour), we believe that we can greatly reduce this time even further with future development.

5. Cost
   This system should cost less than $2000 to implement with the primary cost being the handheld 3D scanner. In addition, the reduction in cranioplasty time and reoperations due

to complications associated with ill-fitting prosthetics should significantly reduce the overall cost of the cranioplasty procedure.

Approach and Algorithms

The approach of the subsystem is to first take a 3D scan of the skull's area of defect. This scanned data then has segmentation performed on it to retrieve the walls of the defect area. Once the segmentation algorithm is complete, the segmented defect needs to be registered to the patient's skull, and in doing so, the CCI. This process works because the oversized implant is fashioned originally from the patient's CT scan data - its point cloud is already correctly aligned with the patient's skull. After registration, a machine path around the defect can be found in the CCI space which allows for fast and accurate implant creation. An overview of the separate segmentation and registration algorithms will be outlined below.

Segmentation

A scan of the defect site of the skull is taken with a 3D camera. In the case of the project, a Structure Sensor is used in combination with Skanect. The outputted 3D point cloud is then segmented using a directional filter and a scoring system. The segmentation algorithm used is from the previous year's project and the implementation will not be covered in-depth.

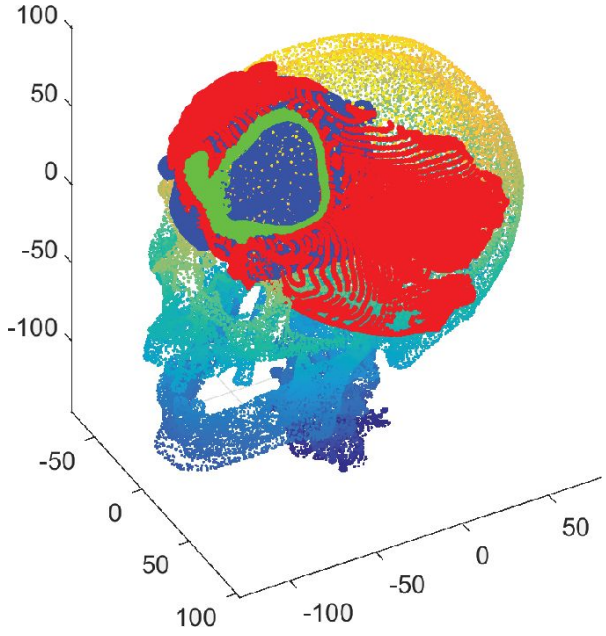
Physical ground truth models were not used as the construction time and price were both out of the project's scope; however, plans are being made to construct 3D prints during the continuation of the project. To quickly and effectively test the robustness of the segmentation algorithm, a number of simulated defects were constructed via patient CT scans of skulls. This was done by creating a defect geometry, creating a defect in the skull model using a boolean operation, remeshing the resulting model to have the same point cloud density as the 3D scanner, and then introducing noise and smoothing that was quantitatively similar to the scanner (i.e. average of 1mm error). At the end of this process the simulated model was quantitatively and qualitatively identical to actual scanned models and provided the ability to validate the segmentation algorithm against a myriad of realistic geometries. These simulated defects were constructed with geometries of varying regularity classified as either regular if their bevel angle was constant or irregular if their bevel angle ranged within the model itself. The regular geometries had bevel angles ranged from 45-90 degrees around their entire circumference. Models with irregular geometries had a bevel range of 10 degrees taken from this 45-90 degrees overall range.

Registration

The registration of the defect data back to the skull model proved to be bulk of the work done during the course of the project. The process will primarily revolve around registering the defect area scan to the skull. This defect area point cloud will have the previously segmented defect walls removed via a nearest-neighbors algorithm. This is done so that the geometry of the newly modified defect area is more in line with the original geometry of the skull before the defect.

During the course of the registration process four meshes are imported: The patient pre-operational CT skull scan, the defect area scan, the segmented defect walls, and the oversized implant generated from the CT skull scan. The first step of the registration algorithm is to align the surfaces of the defect scan to the patient CT skull data. We can do this by first considering the fact that the human skull can roughly be approximated by a sphere. This similarly implies that its subsections can be fitted against a sphere. With this in mind, a RANSAC-based sphere fitting algorithm can be used to find the rotational centers of each "sphere" that the respective point clouds are aligned to. Transformation matrices can then be constructed with the point cloud's respective sphere centers as the translational components to center the skull to the origin and the defect area to the surface of the skull. Additionally, the transformation matrix of the skull and defect area scan is applied to the oversized implant and the segmented defect walls. This is done so that a good initial guess for ICP using center of mass alignment can be done.

Now that all the meshes have their surfaces approximately aligned, we must provide ICP with a good initial guess. This is done by rotating the defect site of the defect area scan to be in line with the oversized implant location. This is done by finding the geometric center of both the oversized implant and the segmented defect wall (since this is considered the 'true' geometric center of the defect area scan as well). A rotation matrix that rotates the center of the segmented defect wall and the oversized implant is generated via Rodrigues' Rotation Formula. This generated matrix is then applied to the segmented defect wall and the defect area point cloud.

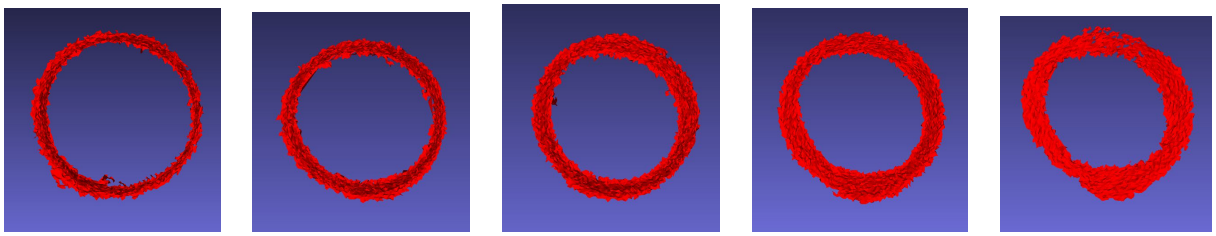


With a good initial guess properly calculated and aligned, registration can begin. Unfortunately, ICP by itself is incredibly prone to local minima problems. Therefore, a two-fold approach is taken. First, the initial guess is perturbed in an area of [-15°, 15°] along each three dimensional axis. In each new location, the defect area point cloud is also rotated a full 360° along the geometric center of its defect site. Both

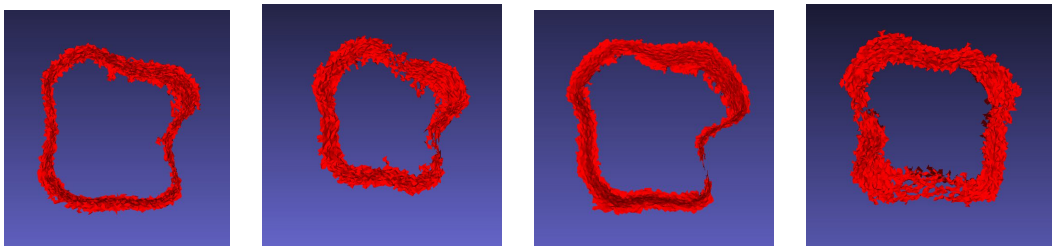
ranges are segmented a number of times defined by an input value: four was the number we chose. Similar to above, Rodrigues' Rotation Formula is used in the computation of the 360º rotation along the geometric center axis. Second, a point-to-surface based ICP is run with the lowest error value transformation found in the previous step as an initial seed to further refine the final registration. This point-to-surface based ICP algorithm is taken from Seth Millings and works by using surface features as another criteria for the error function. This output is our final result.

Results and Significance

Using the method outlined above we gained the following defect walls from the different geometries:



Regular, 90° *Regular, 80°* *Regular, 70°* *Regular, 60°* *Regular, 45°*



Irregular, 90-80° *Irregular, 80-70°* *Irregular, 70-60°* *Irregular 60-45°*

As can be seen from the images, for every geometry intered, the segmentation method was able to produce a complete ring around the defect wall. Some of these models, particularly the irregular ones, do have noise in the resulting output, however our current outline for calculating the machining pose and path is robust to noise. As long as there is some defect wall data around the entire circumference it will be able to create the defect. This validates the robustness of our segmentation method.

Using the registration method described above, our average scan to ground truth defect model error was 0.9mm and we were able to confirm by visual inspection that this minima was in the actual defect location and not just a coincidental local minima. As the error associated with the scanning process is 1mm, this result was well within our expected margin of error. In addition,

the final pose was obtained regardless from initial scan position showing that this method is robust and repeatable.

These results are significant because they are completely automated and robust to initial scan data. This means that they can easily fit into the operator's work flow. The registration method in particular represents significant development in the implant design process as previously there had been no robust way to automatically transform the defect scan into the patient model space.

Management Summary

Division of Labor

The division of labor for this project was distributed equally with both team members working on all components of the project. However, the primary responsibilities of the two members were as follows:

- Erica Schwarz: Creating ground truth models, implementing sphere fitting, Slicer module creation, registration algorithm creation.
- Willis Wang: Matrix transformations, Python conversion, registration algorithm creation, testing and debugging.

Proposed Deliverables

- Minimum
    - Segment and process point cloud of defect to create defect mesh
    - Register defect mesh to patient
    - Register mesh to oversized prosthetic
- Expected
    - Create ground truth models
    - Validate and improve process accuracy
    - Quantify accuracy of implant creation
    - Package process as Slicer module
- Maximum
    - Test process with cadavers
    - Register oversized prosthetic to UR5 machine
    - Define UR5 path for cutting fitted prosthetic

Actual Deliverables

- Minimum
    - Segment and process point cloud of defect to create defect mesh
    - Register defect mesh to patient
    - Register mesh to oversized prosthetic
- Expected
    - Create ground truth models
    - Validate and improve process accuracy
    - Quantify accuracy of implant creation
    - Package process as Slicer module
- Maximum
    - Define UR5 path for cutting fitted prosthetic (in progress)

Future Plans

- Test process with cadavers
- Register oversized prosthetic to UR5 machine
- Define UR5 path for cutting fitted prosthetic

Dependencies

Status	Dependency	Description
Completed	Structure Sensor	Sensor to be used for scanning incision site. Provided by Dr. Armand.
Completed	iPad	iPad to use with structure sensor. Provided by Dr. Armand.
Completed	Software Repository	Provided by Ryan Murphy. Contains existing lab code, system, and test data. This will also be where we store and document our software modules.
Completed	Patient CT Scans	Will be used to create ground truth models. Provided by Ryan.

All dependencies were resolved during the development process.

Milestones

Expected Date	Objective	Completed Date

2/15 - 3/14	Testing Data Production and Planning	3/21
2/22 - 4/25	Registration and Segmentation Development	4/25
3/14 - 5/02	Accuracy Testing	5/04
4/25 - 5/9	Procedure and UR5 Integration	In progress

Future Plans

We have three additional goals for this project that we plan to complete:

1. We plan to do more thorough testing and complete trials with cadavers rather than 3D printed models. We will be working with Mehran Armand, Chad Gordon, and Ryan Murphy to do this. After this data is obtained we plan to write a paper for journal publication or conference that discusses our registration method.
2. We plan to complete work on calculating the machining path. After this is done, the pipeline from defect design to defect creation will be complete. Though currently we believe that the UR5 will be an effective means for machining the personalized implant from the CCI, we will keep the machine pose information general. This is because it may be possible to machine the implant with a laser cutter instead if work on it is complete within our timeline.
3. We also plan to compile a very thorough manual of our process in addition to the basic documentation we have for our code now. As our process deals with several different data types, we want to ensure that future developers can easily identify how to use individual modules from our code.

All of these are to be completed over the upcoming summer, with the main goal being to produce a paper for publication.

Technical Appendices

Below is our annotated Matlab code that outlines our novel registration method for registering a scanned defect to the patient skull model. It is a more straightforward predecessor to our later Python implementation that we use in our final Slicer module and implements the same algorithm. Note that the read_stl and point to surface registration require code from the BIGGS code repository.

Matlab Code:

```
tic;
%Load skull and defect point clouds
[verticesDefect, facesDefect] = read_mesh('ScannedSkullFeaturelessUnits.ply');
[verticesSegment, facesSegment] =
read_mesh('ScannedSkullFeaturelessUnitsSegLargest.ply');
[verticesSkull, facesSkull] = read_mesh('PreDefect.ply');
[verticesImplant, facesImplant] = read_mesh('oversizedImplant.ply');
[verticesPost, facesPost] = read_mesh('PostDefect.ply');
SkullSurface = f_stlMesh_to_surface(f_read_stl('PreDefect.stl'));
PostSurface = f_stlMesh_to_surface(f_read_stl('PostDefect.stl'));

% Remove wall from scan data
rem = ones(length(verticesDefect), 1);
[~, dist] = knnsearch(verticesSegment, verticesDefect);
while min(dist) < 0.0003
      for i = 1:length(dist)
      if dist(i) < 0.0003
            rem(i) = 0;
        end
      end
      verticesDefect(rem(:) == 0,:) = [];
      rem = ones(length(verticesDefect), 1);
      [~, dist] = knnsearch(verticesSegment, verticesDefect);
end

ptCloudSkull = pointCloud(verticesSkull);
ptCloudSegment = pointCloud(verticesSegment);
ptCloudDefect = pointCloud(verticesDefect);
ptCloudImplant = pointCloud(verticesImplant);
ptCloudPost = pointCloud(verticesPost);

%Denoise the point cloud data of defect
ptCloudDefect = pcdenoise(ptCloudDefect);

%Fit sphere to defect
maxDistance = 0.1;
[defectSphere,defectInlierIndices] = pcfitsphere(ptCloudDefect,maxDistance);
defectInliers = select(ptCloudDefect,defectInlierIndices);

%Fit sphere to skull
maxDistance = 0.1;
[skullSphere, skullInlierIndices] = pcfitsphere(ptCloudSkull,maxDistance);

%Shift models to be concentric at origin.
centerDefect = defectSphere.Center;
centerSkull = skullSphere.Center;

ADefect = [1 0 0 0; ...
      0 1 0 0; ...
      0 0 1 0; ...
      -centerDefect(1) -centerDefect(2) -centerDefect(3) 1];
```

```
tform = affine3d(ADefect);
ptCloudDefectCentered = pctransform(ptCloudDefect,tform);
ptCloudSegmentCentered = pctransform(ptCloudSegment, tform);

ASkull = [1 0 0 0; ...
      0 1 0 0; ...
      0 0 1 0; ...
      -centerSkull(1) -centerSkull(2) -centerSkull(3) 1];

tform = affine3d(ASkull);
ptCloudSkullCentered = pctransform(ptCloudSkull,tform);
ptCloudImplantCentered = pctransform(ptCloudImplant,tform);
ptCloudPostCentered = pctransform(ptCloudPost, tform);


translateBase = [-centerSkull(1) -centerSkull(2) -centerSkull(3)];
translate = repmat(translateBase, size(SkullSurface.nodeData, 1), 1);
translatePost = repmat(translateBase, size(PostSurface.nodeData, 1), 1);

SkullSurface.nodeData = SkullSurface.nodeData + translate;
PostSurface.nodeData = PostSurface.nodeData + translatePost;

%Match defect center of mass to oversized implant center of mass
centerImplantMass = mean((ptCloudImplantCentered.Location));
centerDefectMass = mean((ptCloudDefectCentered.Location));
centerSegmentMass = mean(ptCloudSegmentCentered.Location);


% Rotate to match center of masses

%Convert vectors to unit vectors
unitImplantCenter = centerImplantMass/norm(centerImplantMass);
unitDefectCenter = centerDefectMass/norm(centerDefectMass);
unitSegmentCenter = centerSegmentMass/norm(centerSegmentMass);

%Use Rodrigues' Rotation Algorithm
v = cross(unitImplantCenter, unitSegmentCenter);
s = norm(v);
c = dot(unitImplantCenter, unitSegmentCenter);

v_cross = [0 -v(3) v(2); ...
      v(3) 0 -v(1); ...
      -v(2) v(1) 0];

R = eye(3) + v_cross + v_cross*v_cross*(1-c)/(s*s);
AMass = [R(1,:) 0; ...
      R(2,:) 0; ...
      R(3,:) 0; ...
      0 0 0 1];

tform = affine3d(AMass);
ptCloudDefectCentered = pctransform(ptCloudDefectCentered,tform);
ptCloudSegmentCentered = pctransform(ptCloudSegmentCentered,tform);
```

```
figure();
pcshow(ptCloudDefectCentered.Location, 'r');
hold on
pcshow(ptCloudSkullCentered);
pcshow(ptCloudImplantCentered.Location, 'b');
pcshow(ptCloudSegmentCentered.Location, 'g');
hold off;

%%
%The number of times each separate axis is partitioned. The total call
%number will be this number to the power of 3.
numberOfTests = 4;

%Initialize values
currentLowestError = inf;
finalDefectCentered = [];

%Downsample the skull data
ptCloudSkullDown = pcdownsample(ptCloudSkullCentered,'random',1);
ptCloudImplantDown = pcdownsample(ptCloudImplantCentered, 'random',1);

%Create list of degrees to iterate through. Last index is removed since 2pi
%is equivalent to 0.
degrees = linspace(-pi/12, pi/12, numberOfTests);

%Counter to see how far into the program we currently are
count = 0;

%For each axis...
for i = degrees
      for j = degrees
      for k = degrees
            %Create various "about axis" rotations
            x_transform = [1 0 0 0; ...
                        0 cos(i) -sin(i) 0; ...
                        0 sin(i) cos(i) 0; ...
                        0 0 0 1];
            y_transform = [cos(j) 0 sin(j) 0; ...
                        0 1 0 0; ...
                        -sin(j) 0 cos(j) 0; ...
                        0 0 0 1];
            z_transform = [cos(k) -sin(k) 0 0; ...
                        sin(k) cos(k) 0 0; ...
                        0 0 1 0; ...
                        0 0 0 1];

            %Apply "about-axis" rotations
            tform = affine3d(x_transform.');
            tempCloud = pctransform(ptCloudDefectCentered, tform);
            tempSegment = pctransform(ptCloudSegmentCentered, tform);

            tform = affine3d(y_transform.');
            tempCloud = pctransform(tempCloud, tform);
            tempSegment = pctransform(tempSegment, tform);
```

```
            tform = affine3d(z_transform.');
            tempCloud = pctransform(tempCloud, tform);
            tempSegment = pctransform(tempSegment, tform);


            tempCloudCenter = mean(tempSegment.Location);
            unitTempCenter = tempCloudCenter/norm(tempCloudCenter);


         %Rotate along the center-of-mass axis
            interval = 2*pi/(numberOfTests);
            for l = 1:numberOfTests
            K = [0 -unitTempCenter(3) unitTempCenter(2); ...
                  unitTempCenter(3) 0 -unitTempCenter(1); ...
                  -unitTempCenter(2) unitTempCenter(1) 0];

            R = eye(3) + (sin(interval*l)*K) + (1-cos(interval*l))*K*K;
            R = [R(1,:) 0; ...
                  R(2,:) 0; ...
                  R(3,:) 0; ...
                  0 0 0 1];

            tform = affine3d(R);
            icpCloud = pctransform(tempCloud, tform);


            %ICP from segmented defect to skull
            ptCloudDefectDown = pcdownsample(icpCloud,'random',1);
            [tform, movingReg, error] = pcregrigid(ptCloudDefectDown,
ptCloudSkullDown);

            %Check error
            if (error < currentLowestError)
                  currentLowestError = error;
                  finalDefectCentered = movingReg;
                  finalTform = tform;
            end

            %Display counter
            count = count + 1;
            disp([num2str(count), '/', num2str(length(degrees)^4)]);
            end
      end
      end
end

figure();
pcshow(finalDefectCentered.Location, 'r');
hold on;
pcshow(ptCloudSkullCentered);
hold off;

%%
```

```
surfReg = SurfaceRegistration;
surfReg.SamplePoints = finalDefectCentered.Location.';
surfReg.SurfaceModel = SkullSurface;
surfReg.Register;

tform = affine3d(surfReg.T_Final.');
finalDefectCentered = pctransform(finalDefectCentered,tform);

%Show final point clouds
figure();
pcshow(finalDefectCentered.Location, 'r');
pcwrite(finalDefectCentered,'finalDefect','PLYFormat','binary');
hold on;
pcshow(ptCloudSkullCentered);
hold off;
toc;
```

References

1. Aspert, Nicolas, Diego Santa Cruz, and Touradj Ebrahimi. "MESH: measuring errors between surfaces using the Hausdorff distance." *ICME (1)*. 2002.
2. Cates JE, Lefohn AE, Whitaker RT. GIST: an interactive, GPU based level set segmentation tool for 3D medical images. Med Image Anal. 2004 Sep 8 (3):21731.
3. Cignoni, Paolo, Claudio Montani, and Roberto Scopigno. "A comparison of mesh simplification algorithms." *Computers & Graphics* 22.1 (1998): 37-54.
4. Gordon CR, Fisher M, Liauw J, Lina I, Puvanesarajah V, Susarla S, Coon A, Lim M, Quinones Hinojosa A, Weingart J, Colby G, Olivi A, Huang J. Multidisciplinary Approach for Improved Outcomes in Secondary Cranial Reconstruction: Introducing the Pericranial Onlay Cranioplasty Technique. Neurosurgery. 2014 Jun 10 Suppl 2:17989.
5. Herbert M, Pantofaru C. A Comparison of Image Segmentation Algorithms. Carnegie Mellon University 2005. The Robotics Institute
6. Huang GJ, Zhong S, Susarla SM, Swanson EW, Huang J, Gordon CR. Craniofacial Reconstruction with Poly (Methylmethacrylate) Customized Cranial Implants. The Journal of Craniofacial Surgery. 2015 Jan;26(1):6470.
7. Murphy RJ, Wolfe KC, Gordon CR, Liacouras PC, Armand M, Grant GT. Computer-assisted Single-stage Cranioplasty. IEEE. Jan 2015.