Project Proposal

3D Reconstruction of Infants' Cranial Shape using Mobile Devices

EN 601.656 Computer Integrated Surgery II

David Shi
Biomedical Engineering,
Computer Engineering
dshi9@jhu.edu

Tara Tang
Biomedical Engineering
ttang14@jhu.edu

Table of Contents

1. Clinical Motivation

Before an infant is six months old, their skulls are easily deformed, which can result in a number of cranial defects. One of the most common defects, deformational plagiocephaly/brachycephaly (DPB), has seen a rise in case numbers ever since parents were recommended to put babies on their backs to reduce the risk of sudden infant death syndrome (SIDS). Cranial deformity is becoming a pediatric epidemic, affecting up to 46% of newborns in the United States [1]. While DPB is non-synostotic, meaning it does not damage the brain and only causes facial asymmetry, other cranial defects like craniosynostosis, which causes highly morbid brain damage and requires early surgical intervention, are more serious [2, 3].

Early detection of DPB, craniosynostosis, and other skull-related deformities can prevent long-lasting trauma and potentially allow for helmet therapy or minimally invasive surgery. If deformities are not detected after six months, treatment can get much more complicated and require open surgery to prevent brain damage and death [4].

With this project, we aim to improve the way skull shapes are measured and examined by pediatricians and primary care providers. By generating a 3D model of a baby's head, we can provide physicians with a more effective, accurate, and detailed method for evaluating and diagnosing children with skull deformities.

2. Prior Work

Currently, pediatricians only have a simple measuring tape to measure circumference and reference a head-circumference-for-age percentile chart for diagnosis of skull defects [1]. More often than not, when parents are concerned about their infant's misshapen head, the deformity is too subtle for easy detection, and the physician dismisses their concerns. Consequently, 86% of infants with DPB are not identified [4].

PediaMetrix has developed computer vision methods that operate on 2D images to provide a better approximation for circumference and other metrics, such as cranial index (CI) and cranial vault asymmetry (CVA). However, 2D images have limitations, as they come only from the top-down view of an infant's head. Thus, 3D parameters like volume and height cannot be measured. To address this, a 3D model of a baby's head would provide better detection of cranial defects, providing more information than what a 2D image can offer. Though conventional 3D scanners are available, they are expensive and only exist in the offices of neurosurgeons, plastic surgeons, and helmet clinics, not in the offices of general pediatricians. A mobile app solution that can generate accurate 3D models would be easily accessed by pediatricians through smartphone and tablet technology and enable point-of-care testing.

3. Goals

The goal of this project is to develop a software pipeline to reconstruct an accurate 3D model of a baby's head, using depth information from a sensor and mobile application. This project is motivated by the need for better techniques in diagnosing cranial deformities in babies younger than six months of age, as described above.

The work done in this project will be used as the initial foundation for PediaMetrix's deformity diagnosis tool. This tool will ultimately incorporate our 3D model reconstruction with a machine learning algorithm that can analyze the model to identify potential deformed areas and make a diagnosis.

By providing a starting point for a quicker and easier method of diagnosis for pediatricians, we hope to reduce the need for helmet therapy and physical therapy for children who were not diagnosed early enough, which is currently a large financial hurdle for many families [4]. Additionally, we also hope to enable the early detection of more severe conditions like craniosynostosis to avoid open surgery or brain damage.

4. Technical Approach

4A. 3D Reconstruction

For data collection, a depth sensor (Structure Sensor, Occipital) attached to an iPad and app will be given by PediaMetrix that can record color images (RGB), and depth maps (D). We will also be provided with a 3D-printed head phantom of a baby with severe plagiocephaly and its corresponding STL file, which contains the surface geometry information for the phantom.

A depth map is akin to a grayscale image that contains the spatial distance from camera to object. Each pixel is populated with this measured distance instead of an intensity value like the pixels of RGB images. To get the distance between adjacent pixels, we will need the intrinsic parameters (like focal lengths) of our camera, which we will have. Using this information and more depth maps taken from different angles, we can recreate a 3D space.

A feature-based approach to tackling this reconstruction with RGB-D datasets combines feature extraction from computer vision and registration techniques, resulting in an RGB-D SLAM framework [3]. We first start by extracting visual features from the RGB input of different frames using feature extractors like ORB, SIFT or SURF. After getting these features, we match them to the features in previous frames. Each of these features will then have a 3D point correspondence with the depth information. Then, we can estimate the transformation between frames using registration algorithms and optimize the sampling of features using RANSAC. If a frame is matched and transformed accurately, we add it to a pose graph that will help in computing the 3D reconstruction.

We can also use an optimizer to improve the pose graph and the global trajectory of our camera motion like the Python binding of the g^2o library. To store the information of our reconstruction, we turn to more efficient representations of the data space like the Python octree-based mapping, OctoMap, to manage the voxel storage.

4B. Registration

Once the 3D reconstructed model is created, it will be registered through iterative closest points (ICP) to a reference world frame. The preferred method for this registration process is an Iterative Closest Point (ICP) algorithm. ICP is a commonly used algorithm for iteratively comparing two point clouds in order to find a frame transformation (consisting of a rotation and translation) to convert from one to another.

In our case, we are defining the origin of the world frame as the mean point of the first point cloud we collect for our 3D reconstruction. The reconstruction will be generated in reference to that first point cloud, such that our resulting model will already be in the world frame. Then, during evaluation, our “test points” will be the points making up the ground-truth or Occipital reconstructed model, and we will register those to the same frame as our own model.

We begin the ICP procedure with an initial guess for the frame transformation, **F**, from the test points to the model points. Once we apply **F** to the test points, we search for the closest points in the model. This gives us a set of point pairs consisting of a test point (transformed to the world frame) and its corresponding closest model point.

We compare the distances between these point pairs with a predefined distance threshold and consider those with distances under the threshold to be “inliers.” We use these inlier pairs to generate a new guess for **F**. Then, we begin a new iteration, applying this new **F** to all the test points and finding the next transformation using inliers. As we continually iterate, the number of inliers should increase as our **F** brings the test points closer to the model points. Eventually, the overall error between the transformed test points and the model points will be small enough that we can terminate the algorithm and consider the current **F** to be our registration frame transformation.

Many different modules have been created in Python for performing ICP. We are currently planning on using a module created by Avinash Kak at Purdue University [7], but we also intend to look further into our options once we arrive at this part of the project.

4C. Robustness

In order to make our pipeline robust to baby motion, we will also be collecting data with our 3D phantom positioned on a moving shaker. This shaker will have controllable movement speed settings, which we will adjust to simulate increasing amounts of “fidgeting” from the baby while we collect data. Motion will compromise the fidelity of the depth data in varying degrees when we increase the speed of the motion or when we introduce more complex motion such as acceleration.

In general, we expect motion to present as increased noise in our RGB-D data. Thus, we intend to address these issues with noise-reduction algorithms that we can adapt for depth. We also want to explore options in analyzing how the depth data changes at different viewing angles with consistent unidirectional motion, which will give insight on the direction and magnitude of the motion. We have not fully determined how we will address this problem yet, but it will be a frequent topic of troubleshooting and discussion with the whole team throughout the semester.

4D. Schematic

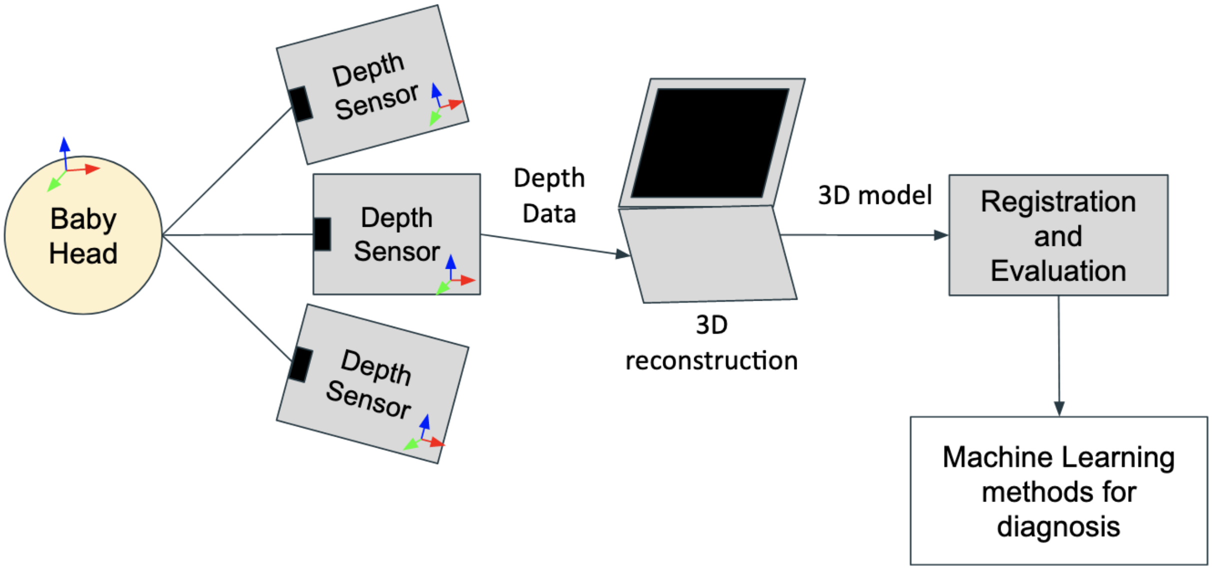


Figure 1: Schematic of software pipeline

We begin with the baby head (phantom), and we will use our depth sensor to collect RGB-D data of the head from multiple angles with their own local coordinate systems. Then, all of this data will go into our SLAM algorithm for 3D reconstruction and output a 3D model. This model will go into our ICP algorithm for registration, and then we can evaluate it against Occipital reconstructions and ground-truth. The final white box is not within the scope of this project, and it represents the direction that PediaMetrix intends to go with our project results. Ultimately PediaMetrix's goal is to input 3D models into a machine learning algorithm to perform automatic diagnosis.

5. Testing Plan

The pipeline will be evaluated on a PediaMetrix-provided baby head phantom with an associated STL file that we can use as ground-truth. The entire sequence, from sensor to 3D model will be tested and the final STL will be compared to the ground-truth STL file. Additionally, Occipital, the company whose sensor will be used for depth measurements, has its own software for generating reconstructions. The Occipital reconstructions will serve as another benchmark for validation and ground truth testing. However, this software does not handle the noise that occurs with movement well, so it necessitates a need for a robust pipeline. While we intend to use the Occipital reconstructions for both stationary and moving phantoms, our goal is to reconstruct a 3D model that will have better results than Occipital for the moving condition.

To calculate reconstruction accuracy, a variety of metrics can be calculated from the STL files, including, but not limited to: average surface distance of the reconstructions, cranial index (CI), cranial vault asymmetry (CVA), and head circumference. We will be using PediaMetrix's pre-existing software for calculating CI, CVA, and head circumference from a given STL file. However, we will need to create algorithms that compute the other evaluation metrics.

One of our primary evaluation metrics will be average surface distance, which will be calculated by comparing the surface contours of our reconstruction and the ground-truth and/or Occipital reconstructions. Our goal is to achieve an average surface distance between the two that is under a certain threshold (more information in the next section).

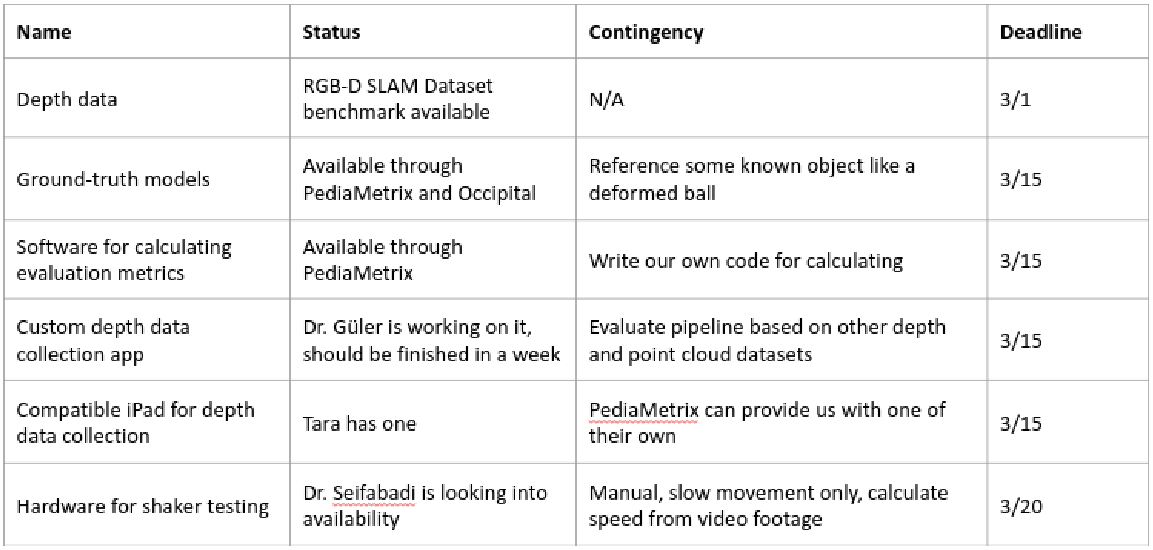
6. Key Activities and Deliverables

	Activity	**Deliverable**
Minimum	1. Collect and process depth data using sensor and phantom 2. Implement 3D reconstruction and registration algorithms in Python, document all code 3. Use PediaMetrix's software to calculate evaluation metrics, compare to ground-truth and Occipital 4. Use shaker on slow setting, evaluate results and refine code for robustness	1. Useable depth datasets 2. Working software pipeline and documentation 3. Accuracy evaluation results (average surface distance threshold #1)* 4. Robust code pipeline with documentation for moving head, slow head motion

Expected	1. Refine code to achieve more accurate results when comparing to ground-truth and Occipital 2. Refine code for faster motion on the shaker	1. Accuracy evaluation results (average surface distance threshold #2)* 2. Robust code pipeline with documentation for moving head, linear motion with no acceleration
Maximum	1. Refine code for more complex motion on the shaker	1. Robust code pipeline with documentation for moving head with acceleration

Figure 2: A table of our key activities and deliverables. (* indicates that we will determine these thresholds as we progress through the project)

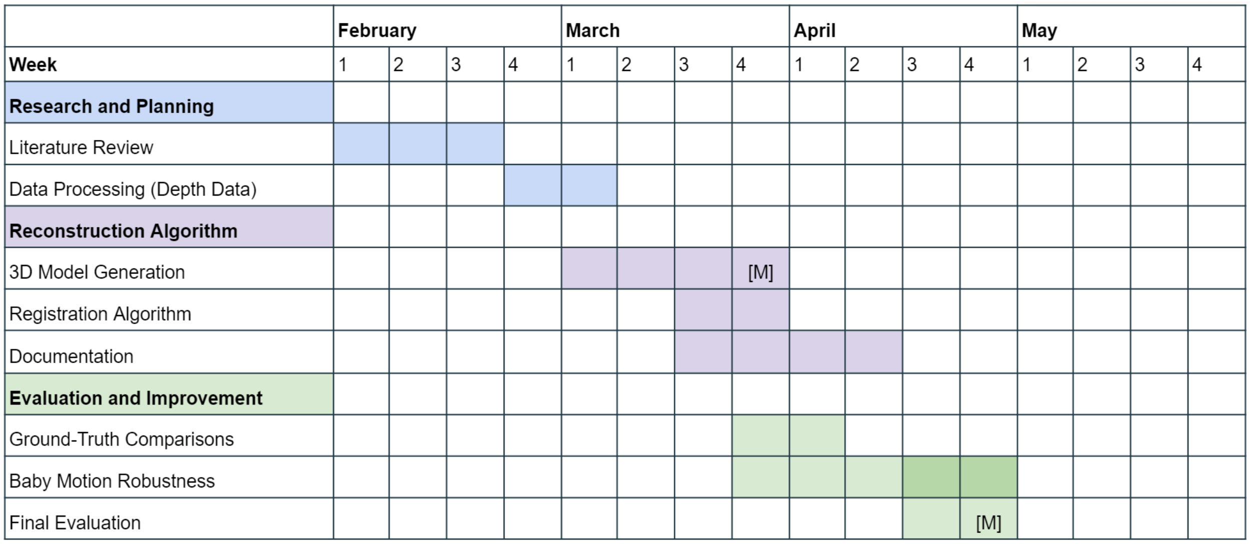
7. Dependencies



Name	Status	Contingency	Deadline
Depth data	RGB-D SLAM Dataset benchmark available	N/A	3/1
Ground-truth models	Available through PediaMetrix and Occipital	Reference some known object like a deformed ball	3/15
Software for calculating evaluation metrics	Available through PediaMetrix	Write our own code for calculating	3/15
Custom depth data collection app	Dr. Güler is working on it, should be finished in a week	Evaluate pipeline based on other depth and point cloud datasets	3/15
Compatible iPad for depth data collection	Tara has one	PediaMetrix can provide us with one of their own	3/15
Hardware for shaker testing	Dr. Seifabadi is looking into availability	Manual, slow movement only, calculate speed from video footage	3/20

Figure 3: Dependencies Table

8. Timeline



	February				March				April				May			
Week	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Research and Planning																
Literature Review	🟦	🟦	🟦													
Data Processing (Depth Data)				🟦	🟦											
Reconstruction Algorithm																
3D Model Generation					🟪	🟪	🟪	🟪 [M]								
Registration Algorithm							🟪	🟪								
Documentation							🟪	🟪	🟪	🟪						
Evaluation and Improvement																
Ground-Truth Comparisons								🟩	🟩							
Baby Motion Robustness								🟩	🟩	🟩	▨	▨				
Final Evaluation											🟩	🟩 [M]				

Figure 4: Timeline of Project

In this timeline, we have denoted major milestones with an [M]. We will also have a minor milestone halfway through March, when we expect to be done with a preliminary model generation algorithm so that we can begin working on registration. The dark green squares at the end of Baby Motion Robustness indicate potential weeks dedicated to our Maximum Deliverable if possible – otherwise, we will use that time to continue working on our Expected Deliverables.

9. Team Members/Mentors and Roles

9A. Team Members

The team consists of:

- David Shi (dshi9@jhu.edu)
  Biomedical Engineering & Computer Engineering
  Sharing all project responsibilities with Tara

- Tara Tang (ttang14@jhu.edu)
  Biomedical Engineering
  Sharing all project responsibilities with David

9B. Team Mentors

The mentors consist of:

- Can Kocabalkanli (can@pediametrix.com)
  PediaMetrix, Primary Mentor

- Dr. Reza Seifabadi (reza.seifabadi@pediametrix.com)
  PediaMetrix, Administrative Mentor and Project Owner
- Dr. Özgür Güler (ozgur.guler.phd@gmail.com)
  PediaMetrix, Technical Expert

10. Management Plan

10A. Meetings

Currently, there are weekly Zoom meetings on Tuesdays at 3:00PM with all team members for project updates and general questions. There will also be impromptu Zoom meetings throughout the week if necessary.

10B. Platforms

- **Communication:** A Slack channel for the project has been made for communication and file transfer.
- **Code:** All code will be stored in a private GitHub repository.
- **Report Writing:** All reports will be written using Google Docs/Overleaf.
- **Documents Storage:** All final documents, presentations, deliverables and links to resources will be on the CIIS Wiki page.

11. Reading List and References

[1] Boston Children's Hospital. Deformational Plagiocephaly. Retrieved from https://www.childrenshospital.org/-/media/Centers-and-Services/Departments-and-Divisions/Plastic-and-Oral-Surgery/PlagiocephalyBook_interactive.ashx?la=en&hash=153679AB2E613576AD9A07E4C32B0F66AF0130B7

[2] Center for Disease Control and Prevention (2020). Facts about Craniosynostosis. Retrieved from Facts about Craniosynostosis – CDC.gov

[3] Endres, F., Hess, J., Engelhard, N., Sturm, J., Cremers, D., & Burgard, W. (2012, May). An evaluation of the RGB-D SLAM system. In *2012 IEEE International Conference on Robotics and Automation* (pp. 1691-1696). IEEE.

[4] Fenton, R., & Gaetani, S. A. (2019). A pediatric epidemic: Deformational plagiocephaly/brachycephaly and congenital muscular torticollis. Contemporary PEDS Journal, 36(2). A pediatric epidemic: Deformational plagiocephaly/brachycephaly and congenital muscular torticollis

[5] Hartley, Zisserman. (2003). *Multiple View Geometry in Computer Vision* (2nd ed.). New York, NY: Cambridge University Press.

[6] Izadi, S., Davison, A., Fitzgibbon, A., Kim, D., Hilliges, O., Molyneaux, D., . . . Freeman,

D. (2011). KinectFusion. Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology - UIST '11. doi:10.1145/2047196.2047270

[7] Kak, Avinash. ICP.py, Version 2.1.1. Purdue University, College of Engineering. Retrieved from https://engineering.purdue.edu/kak/distICP/ICP-2.1.1.html

[8] Kalman, R. E. 1960. “A New Approach to Linear Filtering and Prediction Problems,” Transaction of the ASME—Journal of Basic Engineering, pp. 35-45 (March 1960).

[9] Losee, J. E., & Mason, A. C. (2005). Deformational plagiocephaly: diagnosis, prevention, and treatment. Clinics in plastic surgery, 32(1), 53–viii. https://doi.org/10.1016/j.cps.2004.08.003

[10] Wang, Chieh-Chih; Thorpe, Charles; Thrun, Sebastian; Hebert, Martial; Durrant-Whyte, Hugh (2007). "Simultaneous Localization, Mapping and Moving Object Tracking" (PDF). Int. J. Robot. Res. 26 (9): 889–916. doi:10.1177/0278364907081229

[11] Zhang, Zhengyou (1994). "Iterative point matching for registration of free-form curves and surfaces". International Journal of Computer Vision. 13 (12): 119–152. CiteSeerX 10.1.1.175.770. doi:10.1007/BF01427149

[12] Zollhöfer, M., Stotko, P., Görlitz, A., Theobalt, C., Nießner, M., Klein, R. and Kolb, A. (2018), State of the Art on 3D Reconstruction with RGB‐D Cameras. Computer Graphics Forum, 37: 625-652. https://doi.org/10.1111/cgf.13386