CIS II Final Report
Motion Compensation and Evaluation of 3D Head Reconstruction

May 9, 2022

Students:
Unnat Antani - uantani1@jhu.edu

Mentors:
Can Kocabalkanli
Ozgur Guler
Reza Seifabadi
Pediametrix Inc.

1 Introduction

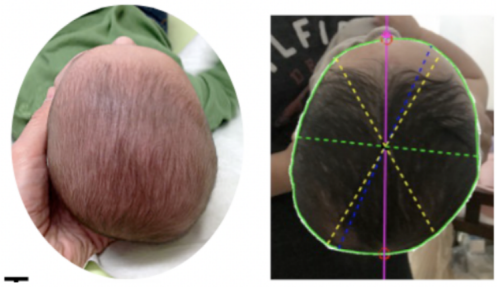
1.1 Goal

To be able to register the head of the patient, taking into account the motion of the head and there by be able to compensate for it to attain rigid registration to further use the registered points to generate a smooth mesh of the scan, to obtain certain parameters to detect deformities from the 3D model.

1.2 Background

It is difficult to detect and measure deformities on baby head. There are no specialized tools for physicians to use to detect such deformities other than a measuring tape.

If such deformities are left unattended, it can cause complications down the line and at the same time making it difficult to deal with. Hence, it is better to deal with the deformities early on to prevent the complications as well as make it easier to correct it at an early stage.



Currently, a vision based approach is used to extract the diameter and the diagonals and that is used to detect deformities. However, 2D techniques do not capture the details that a 3D model would be able to capture.

2 Technical Approach

Two approaches have been explored and assessed for solving the problem of distortion. For both methods, the same phantom head has been used, shown as below:

The two methods explored are:

- Multiway Registration
- Instant Neural Graphics Primitives with a Multi-resolution Hash Encoding

Additionally, we also tested out the MATLAB tools, which are readily available to test out how it performs on the custom data-set compared with our pipeline.

2.1 Using MATLAB [1]

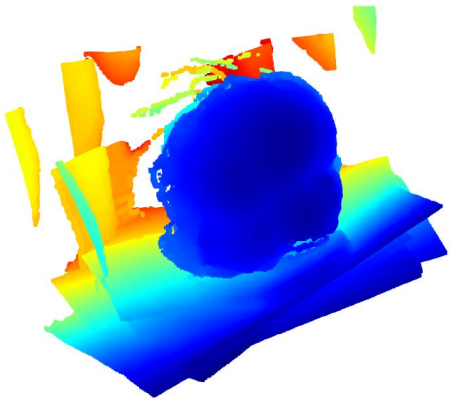
MATLAB has readily available tools for multiview registration that can be applied to any data-set. The goal was to compare the results with the existing pipeline as well as to compare with the new solutions.

2.1.1 Execution

The data-set from the 3D sensor was used to extract the point cloud data and the registration algorithm of MATLAB was implemented.

2.1.2 Results

A snapshot of the registration achieved is as follows;



2.1.3 Analysis

The registration failed on the custom data-set where the existing pipeline and the new solutions were able to register with acceptable margins of error.

2.2 Method 1: Multi-view Registration [2]

2.2.1 Overview

Multiway registration is the process of aligning multiple pieces of geometry in a global space. The input is a set of geometries(i.e., scans P_i) and output is a set of rigid transformations T_i so that the final output is $T_i * P_i$. It works on representing the geometries as Pose graphs.

The graph is such that the nodes represent the geometry and the edges represents the transformation between two geometries which overlap. This intermediate transformation is obtained via Point-to-plane ICP.
The global optimization performs twice on the pose graph. The first pass optimizes poses for the original pose graph taking all edges into account and does its best to remove any false edges and alignments. The second pass runs without them and produces a tight global alignment.

The pose graph edges are partitioned into two classes:
- Odometry edges connect temporally close, neighboring nodes. A local registration algorithm such as ICP can reliably align them.
- Loop closure edges connect any non-neighboring nodes. The alignment is found by global registration and is less reliable.

Then, the problem is converted to an optimization algorithm of minimizing the RMSE error between the overlapping nodes on the graph. We also display the fitness, which is the amount of overlap between the surfaces after every iteration.

2.2.2 Execution

An ipad with a 3d sensor is used to capture data. A custom app has been made by the mentor Dr.Ozgur Guler, which is used to capture the scans, and uploads it to AWS server. The data is stored in terms of YAML and a jpg. For this application, the scans are taken incrementally in a sequence based on the number of scans to take. For example, the below results is obtained from taking 12 scans. So, after starting the scane from a particular position, each subsequent scan is taken incrementally with angular interval of $\frac{360}{12} = 30$ degrees.
The scanner is shown below:

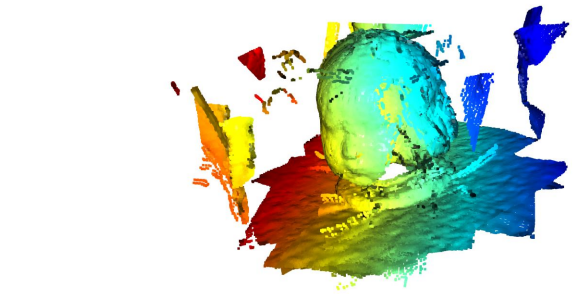
After capturing and downloading the data, a script is run which extracts the point cloud data and stores in terms of a text file. It can also store it in terms of a pcd file format.
Next, the registration algorithm is run which takes the first frame as the base

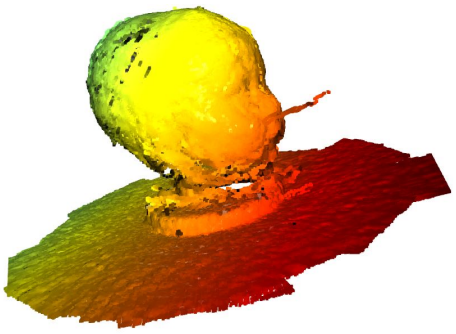
frame and registers all the other frames relative to that base frame. A snapshot of the algorithm running is as shown below:

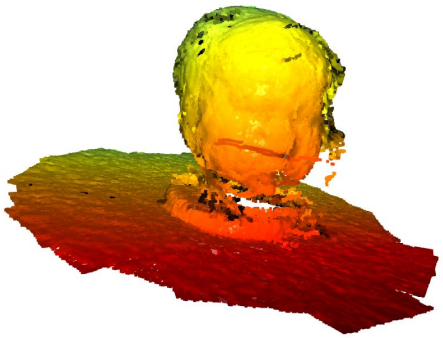
```
[Open3D DEBUG] ICP Iteration #84: Fitness 0.6763, RMSE 2.3144
[Open3D DEBUG] Residual : 3.07e+00 (# of elements : 29072)
[Open3D DEBUG] ICP Iteration #85: Fitness 0.6763, RMSE 2.3145
[Open3D DEBUG] Residual : 3.07e+00 (# of elements : 29072)
[Open3D DEBUG] ICP Iteration #86: Fitness 0.6763, RMSE 2.3144
[Open3D DEBUG] Residual : 3.07e+00 (# of elements : 29072)
[Open3D DEBUG] ICP Iteration #87: Fitness 0.6763, RMSE 2.3145
[Open3D DEBUG] Residual : 3.07e+00 (# of elements : 29072)
[Open3D DEBUG] ICP Iteration #88: Fitness 0.6763, RMSE 2.3144
[Open3D DEBUG] Residual : 3.07e+00 (# of elements : 29072)
[Open3D DEBUG] ICP Iteration #89: Fitness 0.6763, RMSE 2.3145
[Open3D DEBUG] Residual : 3.07e+00 (# of elements : 29072)
[Open3D DEBUG] ICP Iteration #90: Fitness 0.6763, RMSE 2.3144
[Open3D DEBUG] Residual : 3.07e+00 (# of elements : 29072)
[Open3D DEBUG] ICP Iteration #91: Fitness 0.6763, RMSE 2.3145
```

2.2.3 Results

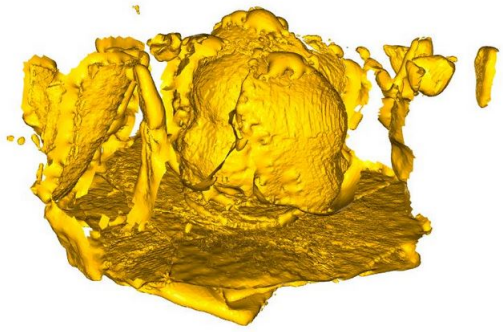
The results of the algorithm is as follows:







However, the 3D reconstruction didn't give a smooth and better model:



2.2.4 Analysis

As seen above, even with more scans and better registration, the 3d reconstruction doesn't output an acceptable 3D model for further analysis.

This method also requires more number of scans if one wants better registration. The above scan result was done with 12 scans. Each scan was taken sequentially with interval of 30 degrees plus some distortion.

Not to mention, this method requires a 3D scanner to be applied, which might not be readily available, and is expensive as well.

The final result also gives out a lot of artefacts, which might be a result of noise from the sensor or sometimes even during the upload and download of data from the AWS server, resulting in a corrupted file.

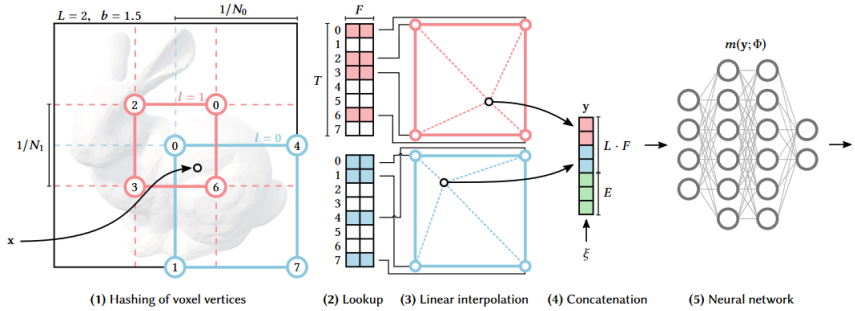
2.3 Method 2: NeRF [4]

2.3.1 Overview

The basis of this approach is to represent a scene with neural network as primitive. A neural network is overtrained on a scene, along with a hashing table at multiple resolution which is also trainable, which then represents the entire scene.

Given a fully connected neural network $m(y;\phi)$, we are interested in an encoding of its inputs $y = enc(x;\theta)$ that improves the approximation quality and training speed. Our neural network not only has trainable weight parameters ϕ, but also trainable encoding parameters θ. These are arranged into L levels, each containing up to T feature vectors with certain dimensionality. Each level stores feature vectors at the vertices of a grid, the resolution of which is chosen to be a geometric progression between the coarsest and finest resolutions. A

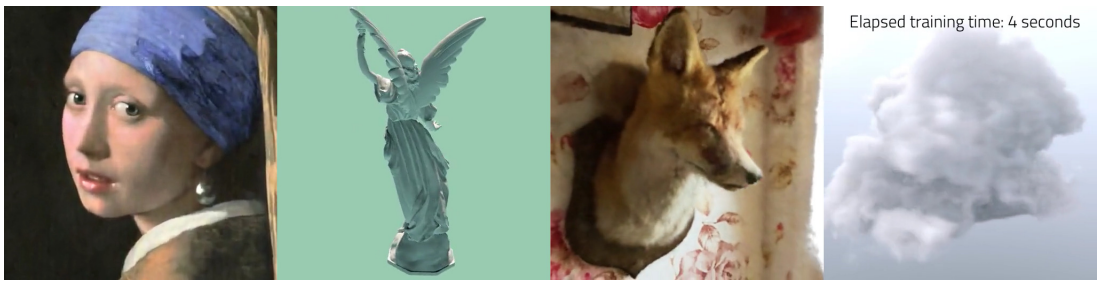
visual description is as follows:



Implementation is done with CUDA using the Nvidia's tiny-cuda-nn framework. For NeRF, the architecture of the Neural Network is as follows:
- 2 concatenated Multilayer Perceptrons(MLP) each with 2 hidden layers and width of 64 neurons:

- A density MLP
- Color MLP

The density MLP maps the hash encoded position y = enc(x;θ) to 16 output values the first of which we treat as log-space density.
The input of color MLP are the 16 coefficients of harmonic basis function which determines the projection and the output of density MLP. The final output is an RGB triplet for each position. Some of the reconstruction results from the author are [5]:

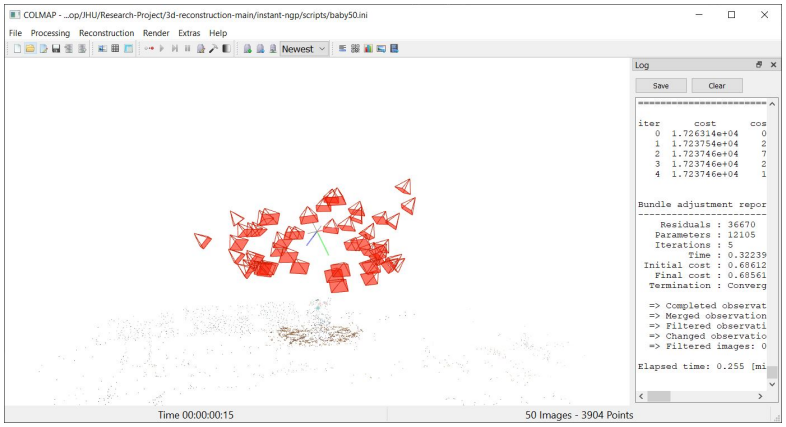


2.3.2 Execution

The images of the phantom are first taken with a phone. The sequence of capture doesn't matter. A render can be obtained with as few as 6 images taken from different angles. However, to obtain a better 3D model, at least 50 images or more should be taken. A few samples are as shown below:

In order to mimic the head motion, after 50 scans, the head was translated by 2 cm and 42 more images were taken, with a total of 92 scans.

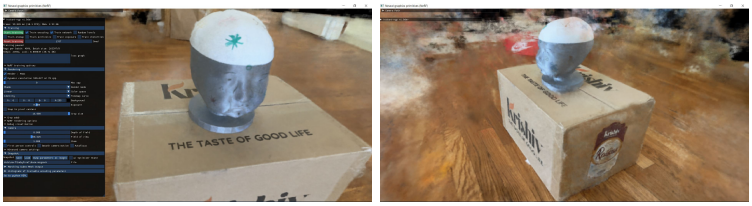
Since the images from phone obtained are of the type .HEIC, it is first converted to an acceptable format. In this case, it will be .jpg.
Next, a software is to be used, namely COLMAP [7], which uses the camera information from these images, and the images themselves, and reconstruct the frames from which these images were taken. The transformation of these frames is recorded as a model, which is then stored in a json format. A snapshot of the process is as shown below:



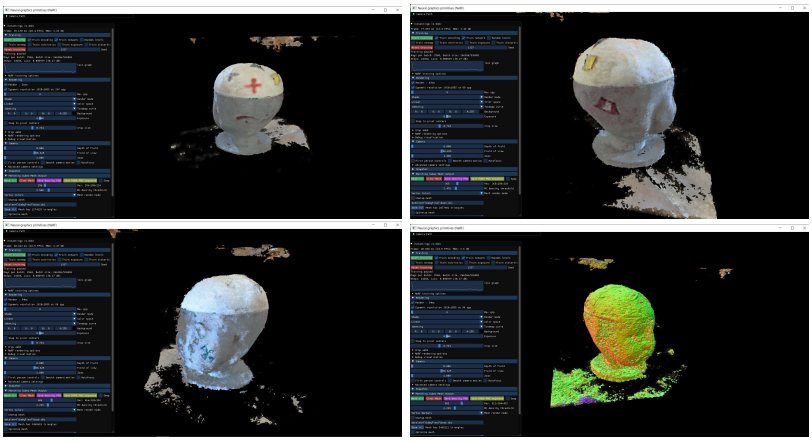
The transformation map generated along with the images are the NeRF model, which is utilized by the NeRF pipeline. The neural network is then called in to run on the scene generated and only after a few seconds of learning, it is able to render the 3D scene, after which Marching Cubes algorithm is used to generate a 3D model of the rendered scene.

2.3.3 Results

Snapshots of the renders are as shown below:

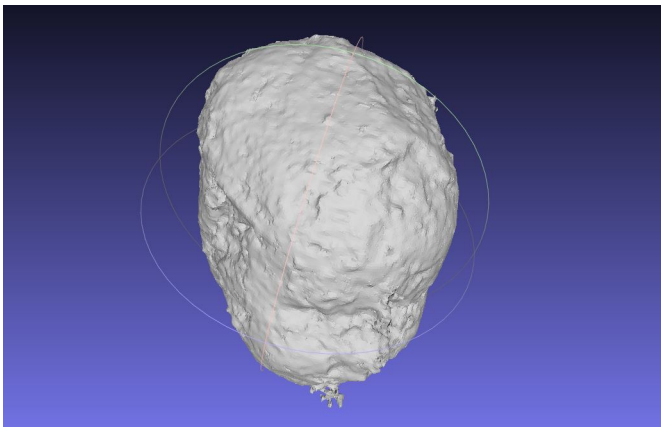


Marching cubes is used with varies resolutions to generate mesh. Some snapshots of the mesh are as shown below:

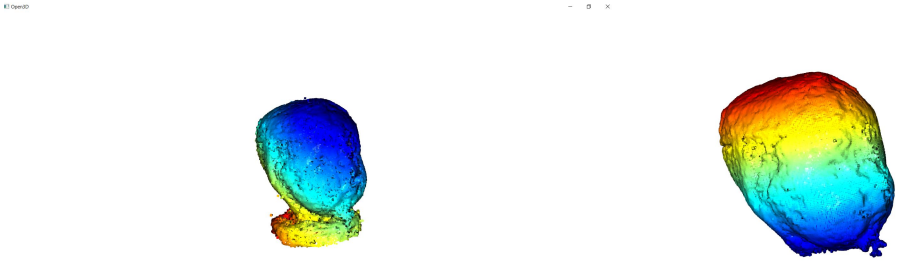


A script has also been made which automates the process after downloading the images, from changing the datatype to generating the render.

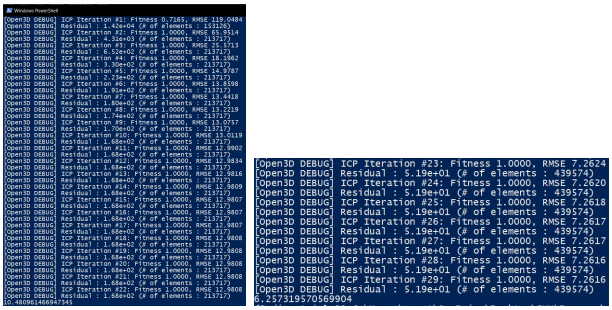
The 3D model was cleared manually using Blender [8]. The cleaned model is as shown:



It is then evaluated using a script which registers the model with a 3D model of the phantom as ground truth. The following are the results:



As seen above, the two models overlap completely without any issues. The evaluation results of the above is given in terms of fitness i.e., amount of area overlap and RMSE, i.e., the error between the corresponding points.



Note, the RMSE error is more, because the reconstruction happend for the entire volume instead of just the surface, hence, internal points are taken into consideration when evaluating which skews the RMSE. However, the fitness, i.e., the overlap shows that the registration is successful for the scaled model.

The CVA for the unprocessed model with internal points, is given below:

Measurements	Calculated
Head Circumference	369.2 mm
Largest Circumference	20.0 mm
CI	92.94
CVAI	27.01
CVA	5.99

The CVA for processed models is as shown below:

Measurements	Calculated	Measurements	Calculated
Head Circumference	463.9 mm	Head Circumference	205.1 mm
Largest Circumference	22.0 mm	Largest Circumference	20.0 mm
CI	81.74	CI	82.50
CVAI	-0.98	CVAI	-2.65
CVA	-1.52	CVA	-1.83

2.3.4 Analysis

The method introduced uses only 2D images, thus removing all the costs and problems associated with having a 3D sensor. Not only that, but the neural network only takes a few seconds to generate the scene. It is incredibly powerful and robust and performs well with distortions of upto a few cm.

Some disadvantages of the method include use of GPU hardware, in absence of which this cannot be applied. The performance and the speed will also depend on the available hardware on the system. A lot of fine tuning is required to generate the model using the Marching Cubes, which again is restricted based on the available hardware. In a system with 8 GB Graphics, the maximum resolution of the model possible before running out of memory is 512x384x432.

The CVA as well as the RMSE error is more because the reconstruction happens for the entire volume and needs lots of manual cleaning and processing before it generates acceptable results, as seen in the latest results where the CVA is around 1.7 mm.

3 Future Work

Future work includes refining the model, exploring different hashing methods and improving the 3D reconstruction. An app that can upload the images from the phone can be made, and automation of retrieving the images also needs to be carried out.

4 Management Summary

4.1 Acknowledgements

This project was made possible by the incredibly valuable insights by the mentors; Can Kocabalkanli, Ozgur Guler and Reza Seifabadi. I would also like to thank Babak V-Ghaffari, who joined the project later on and worked alongside me to give more insights and test the approach and the data. I would also like to thank Professor Russell Taylor for pushing to be more detail oriented by giving feedback on the presentations and the reports.

4.2 Deliverable

Minimum✓:

- 1 algorithm implemented in the pipeline
- Testing the algorithm on real world data set
- Accuracy of $\leq$ 2 mm average surface distance, $\pm$ 2.5% CI/CVAI

Expected✓:

- 2 algorithms implemented in the pipeline
- Testing the algorithm on real world data set
- Accuracy of $\leq$ 2 mm average surface distance, $\pm$ 1.5% CI/CVAI

Maximum:

- 3 algorithms implemented in the pipeline
- Testing the algorithm on real world data set
- Accuracy of $\leq$ 2 mm average surface distance, $\pm$ 1.5% CI/CVAI
- Creating a simulation environment for testing

4.3 Lessons Learnt

Some of the lessons learnt through the course of the project are:

- Important to maintain good code base. It is very easy to lose track of scripts and how to use them if not maintained properly.
- 3D Reconstruction is a hot spot for research.
- Most researchers will provide their work and help understand it when approached.
- Testing rigorously is important to get robust solutions.
- Most registration solutions now incorporate Deep Learning.
- A month in Computer Vision research is equal to a year's worth of new solutions.
- No amount of GPU is ever gonna be enough.

5 Technical Appendix

The code-base is stored and maintained on the company GitHub. The documentation of the process, a tutorial video on obtaining the render for NeRF method along with all the necessary data including images, point clouds, generated NeRF objects and 3D models are shared via Google Drive.

The project wiki can be found here.

References

[1] 3-D Point Cloud Registration and Stitching

[2] Multiway Registration

[3] Choi, Sungjoon, Zhou, Qian-Yi, Koltun, Vladlen(2015), "Robust Reconstruction of Indoor Scenes", 10.1109/CVPR.2015.7299195

[4] Instant-ngp

[5] NeRF Project Page

[6] Müller, Thomas and Evans, Alex and Schied, Christoph and Keller, Alexander(2022), "Instant Neural Graphics Primitives with a Multiresolution Hash Encoding",https://arxiv.org/abs/2201.05989

[7] COLMAP

[8] Blender