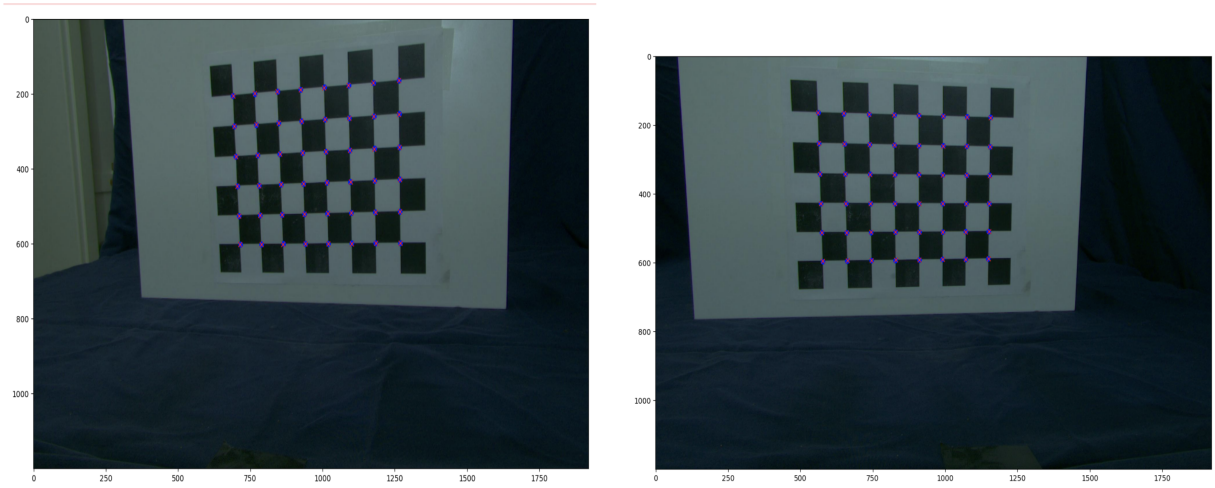
CS 117 Computer Vision Final Project

Section 1: Imports and Calibration

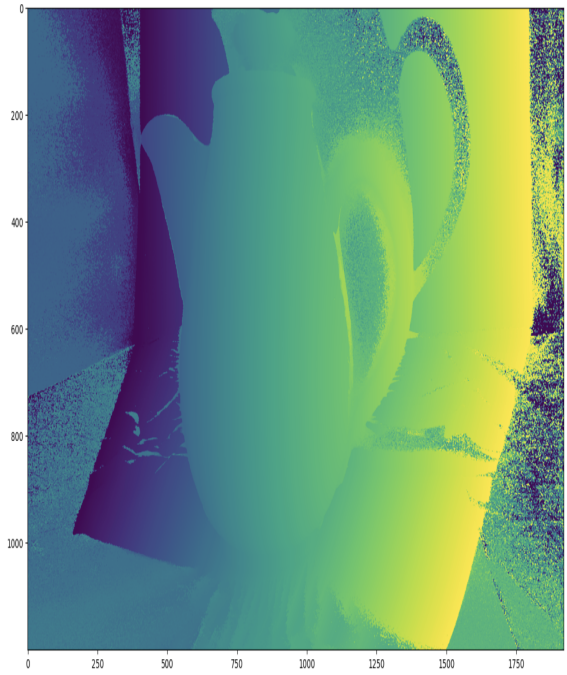
To start with the project, I began with calibrating the cameras using calibration.py and the chessboard provided like in assignment 4 (images are shown to the bottom). The thought process and instructions told us to base our project off of assignment 4 which familiarized us with the first few steps of the project by showing us how to scan data, prune, and reconstruct a mesh. Thus, the libraries provided in assignment 4 were used to start the calibration and reconstruction process of a 2D to 3D object. To calibrate the cameras, the program started by importing a Camera class and calibrating it by using a chessboard just like in assignment 4. This can be seen below. The different pieces of the board were recognized which allowed the camera to later recognize more details in the process. The other part to start the project came from assignment 3, where I took the pickle data to load in and calibrate with the new teapot image scans. A decode function was also defined in order to change the code of the images to be a scene of 10 bit gray code to decode a binary sequence into decimal values at each pixel. The images are rad from the "calib_jpg_u" folder, which allows us to calibrate the cameras to work as intended.

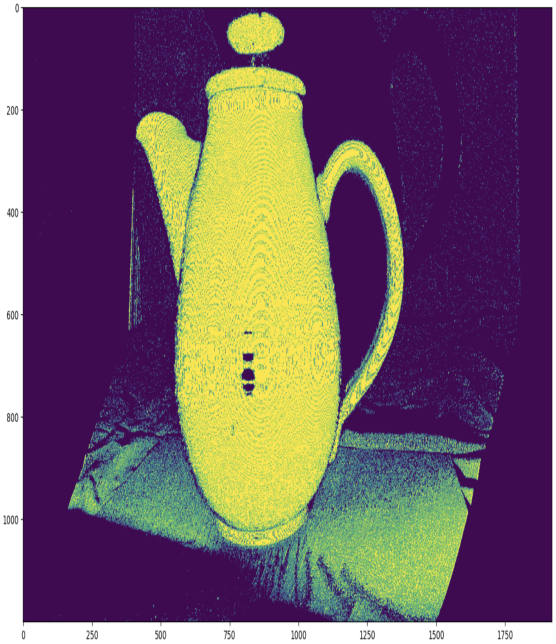


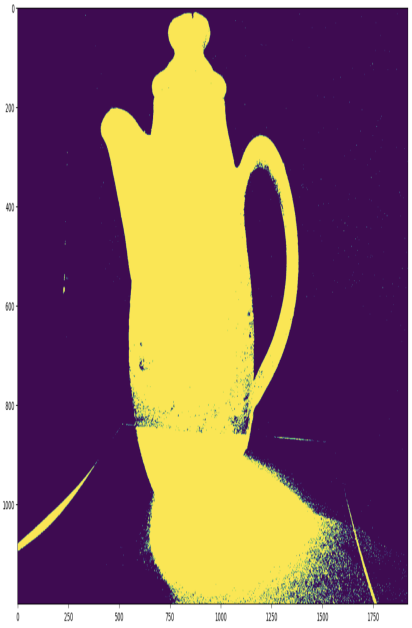
Section 2: Teapot Initialization

After calibration, masking the teapot images using the previously made decode function was next. Decoding allowed the teapot images to create reference images to what is able to be seen with the images in the beginning. The masking creates arrays that correspond to the images of dark spots, color spots, and light spots, which can help in reconstructing the mesh of the teapot later on that will eventually be used to create a 3D model. The images in a directory of teapot image scans are decoded and the general shape of a teapot was made. This is because the pixels correspond to different parts of the image so then we are able to differentiate between the

background and the teapot. The images created are shown below. In order, it corresponds to the coded image, the masked image, and the color mask image. Essentially, the images are set to greyscale.





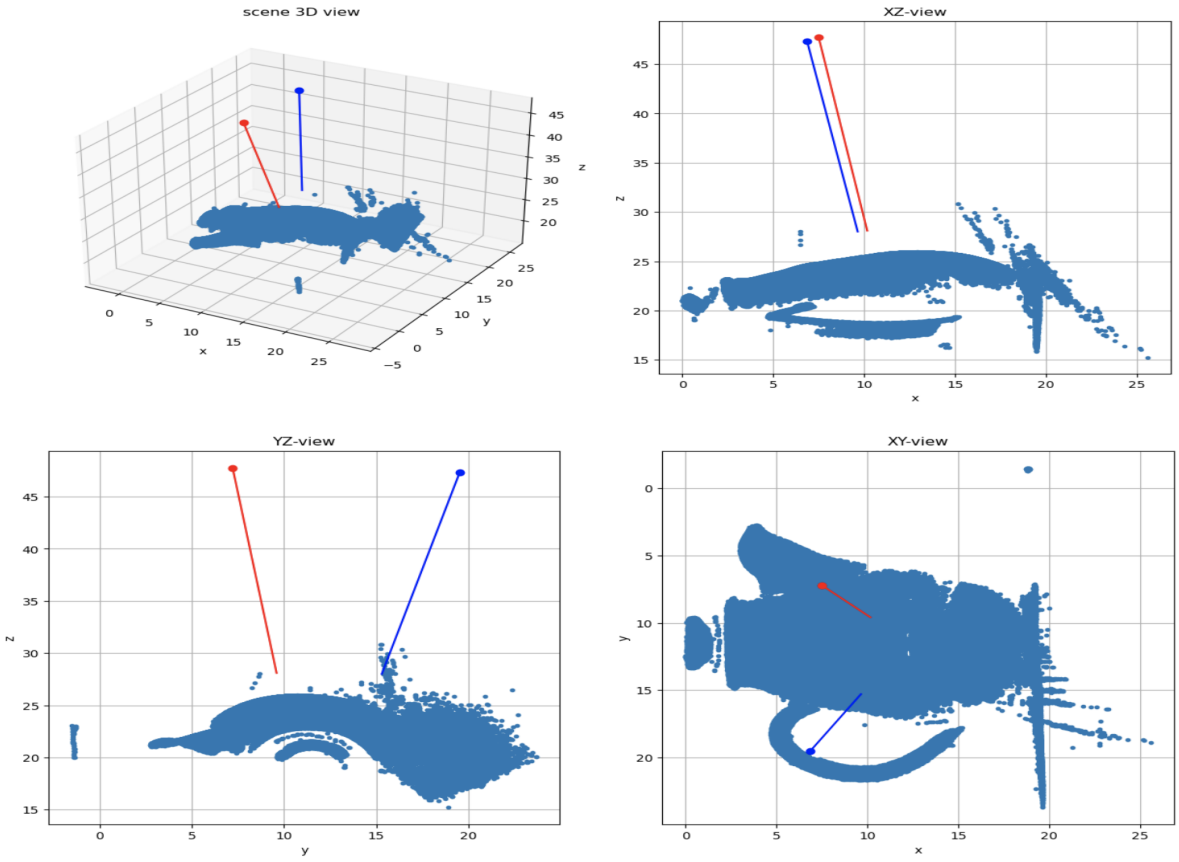


To go in more depth with the greyscale decoding code, it converts a matrix by taking the average of a RGB matrix which is our inputted teapot images. The data type is then converted if it is uint8 and converted into the inverse images. The binary is also converted with the binary to decimal equation. The initial scans are made with the image data from a folder with the prefix "color_C0_." After the data is read and decoded, the data is then triangulated to reduce bias of

data that is gathered from the image. This is because the image is looked at from a left and right camera, thus triangulating the data allows the images to become less biased towards either the left or right side.

Section 3: First Reconstruction of Teapots

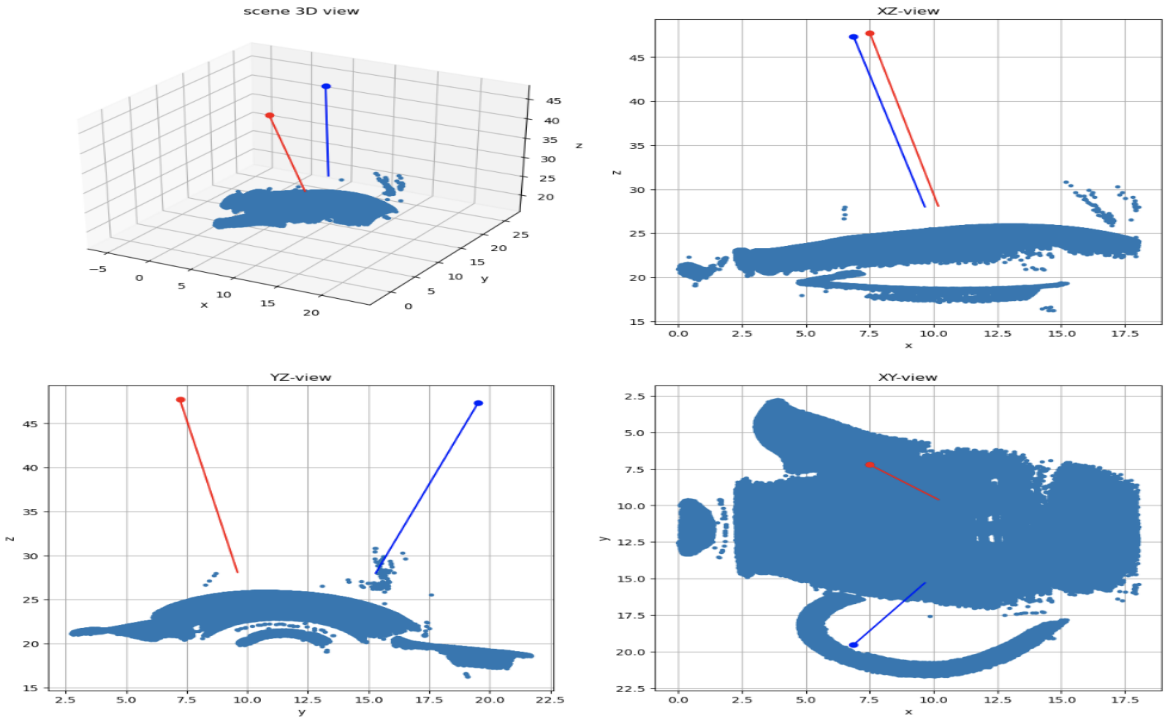
This next step is to read all the files from the first teapot folder which contains the prefix "grab_0_u." This is the first step of reconstruction. The reconstruction function is taken from assignment 4, which essentially takes in the different views of every camera along with the prefix of the folder to perform matching and triangulation of points on the surface using structured illumination. It recognizes gray code, matches it with matching gray pixels, and then triangulates the result to return 2D and 3D coordinates. The function has some slight modifications made to it so that it fits with the scope of the project. The images below show the different views of the reconstructed teapot images using all the data in the "grab_0_u" folder.



As seen, the data is still quite messy as no pruning or data cleaning has occurred yet. These pictures are based solely on multiple views of the camera which scanned images in the "grab_0_u" folder. The images should represent the color and non color reconstructions since they are both in the folder directory.

Section 4: Pruning and Data Cleanup

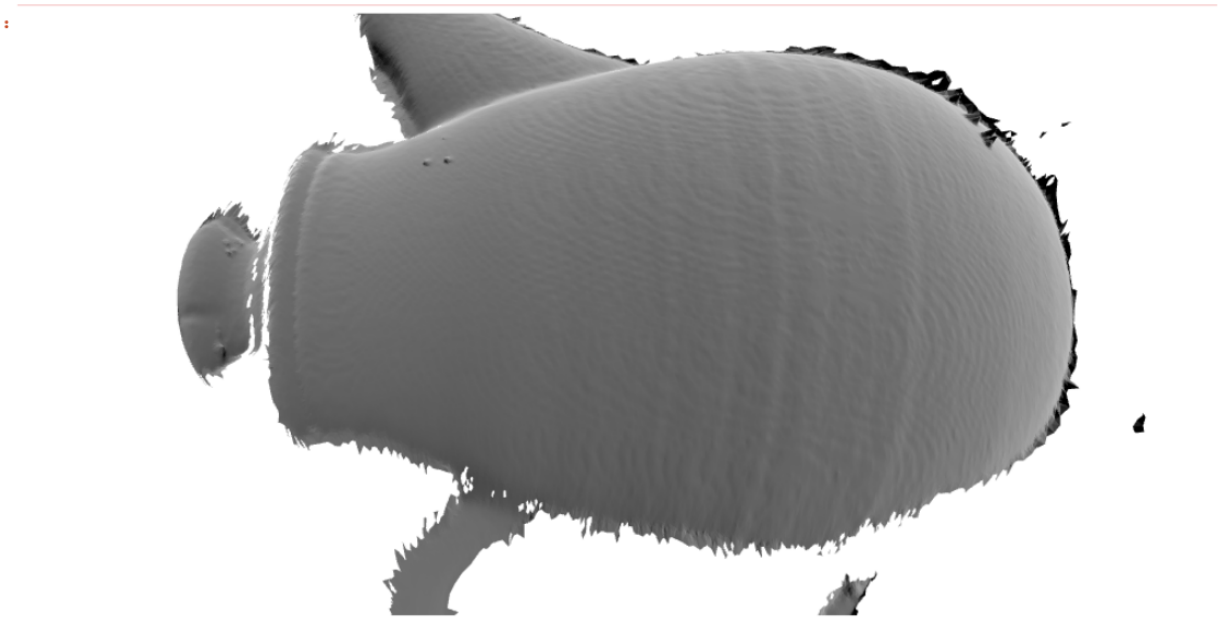
As seen in the previous section, the reconstructions of the teapot are still very messy. This has to do with the data not being cleaned with different pruning methods. To clean up the data, we can access the arrays that are made of the reconstructions and apply the bound box and triangle pruning methods. The bound box pruning method is a method which will only detects points that are inside a box with certain measurements. This was determined through trial and error on my



end. The triangle pruning method is a method which uses Delaunay Triangulation to find triangles that are too long so that we can delete them. The images seen below is the clean data. As seen, the images are very much clear and clean. Nothing changed from assignment 4 to now, the pruning methods stayed the same. The code essentially loops through the images and applies the logic stated for box pruning and triangle pruning. The box limits are defined in a array, and the Delauney library is used to triangulate.

Section 5: Continuation and 3D model

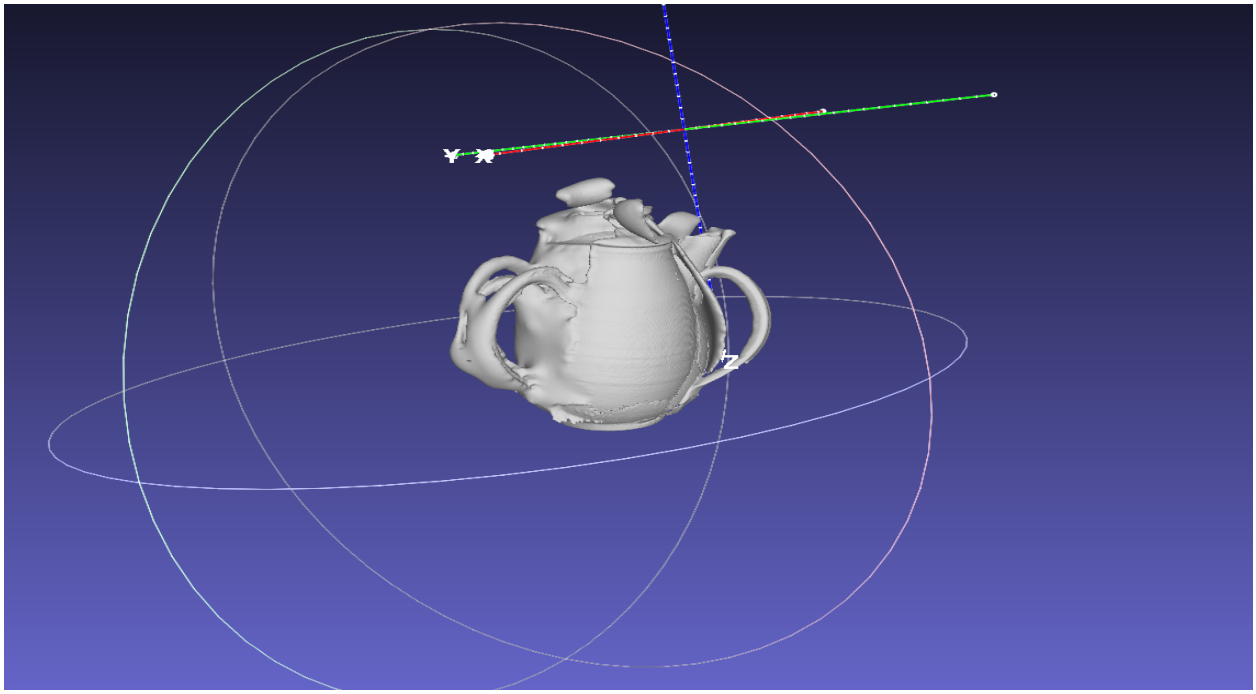
To finish off the project, the rest of the folders were iterated through. This means that the grab_1_u through grab_6_u had all their files read and reconstructed their images. The same methods are made and cleaned throughout these iterations. A mesh is also created with each one as seen below.



Each of these meshes are then exported to create ply files with the statement “trimesh.exchange.export.export_mesh(mesh=mesh, file_obj="mesh3.ply")” to allow the files to be exported to meshlab.

Within meshlab, the meshes made from the 7 folders are exported. Here, I used the translate function to meticulously put the parts together to create a coherent teapot. My final result is seen

mesh1.ply	Today at 11:04 PM	3.4 MB	Polygon...Format
mesh2.ply	Today at 6:58 PM	6.7 MB	Polygon...Format
mesh3.ply	Today at 7:03 PM	7.5 MB	Polygon...Format
mesh4.ply	Today at 7:04 PM	3 MB	Polygon...Format
mesh5.ply	Today at 7:07 PM	6.9 MB	Polygon...Format
mesh6.ply	Today at 7:08 PM	2.5 MB	Polygon...Format
mesh7.ply	Today at 7:10 PM	3.4 MB	Polygon...Format



below.

Thet teapot I made is my reconstruction. Clearly, it is not a clean remake of an actual teapot. I made mistakes with the translations of the pieces. It was hard to fit them together and this was the best I could do. I utilized the meshlab Poisson smoothing after putting together each of the meshes very roughly. The meshes worked out, but due to user inadequacies, the remodeled image does not look right. However, as seen, my process did work, I was able to create meshes and ultimately create a 3D object that resembles a teapot, albeit a very ugly one.