CSE 152A Final Project Writeup (Daryl Shen Ngiap Foo #A16535281)

Approach:

Our input is as a collection of 49 images of a scene taken from different angles.

First, I identified our points of interest. To do this I carried out corner detection using the SIFT corner detection algorithm from OpenCV. The reason the SIFT algorithm was used instead of the Shi-Tomasi corner detector was because the SIFT algorithm is not only rotation invariant like the Shi-Tomasi algorithm, but in addition, SIFT is also scaling invariant, which means even if an image is scaled, the corners can still be found. To do so, I converted each image to grayscale, then obtain its keypoints and descriptors using SIFT. I then drew those keypoints on the original image.

Next, for a pair of adjacent images, I obtained the correspondences based on their keypoints to get the k best matches from the two images. I also used the ratio test to obtain better correspondences. I then used the correspondences (transformed to Euclidean coordinates), as well as the provided intrinsic and extrinsic matrices to triangulate the points.

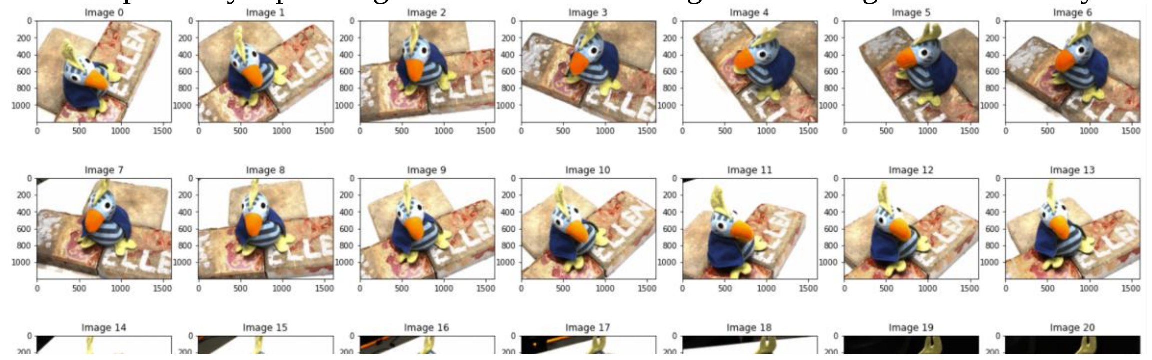
After that, I validated the correspondences using the fundamental matrix, obtained from the corresponding pair's intrinsic and extrinsic matrix. Next, I used the calculated fundamental matrix to effectively remove some outliers from the initial correspondences.

I then repeated this process for all 50 pairs of neighboring images, to obtain the final visualization of correspondences from all 49 images. However, the result produced still had a good amount of outliers.

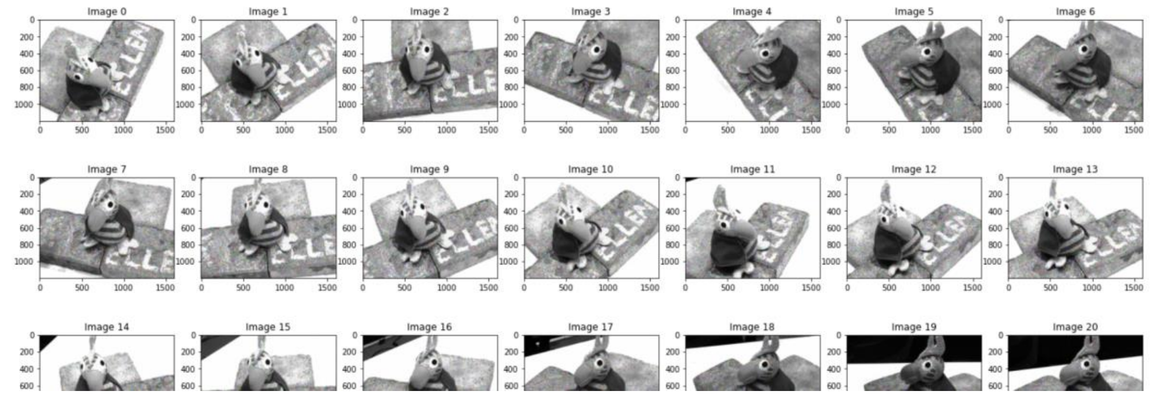
Finally, to reduce the number of points for faster computation, I used voxel down sampling to down sample the points. I ensured that the points were reasonably down sampled, but not to the point that important keypoints were removed. Last but not least, I used statistical outlier removal to the remove the major outliers to obtain the final result.

Process/Results:

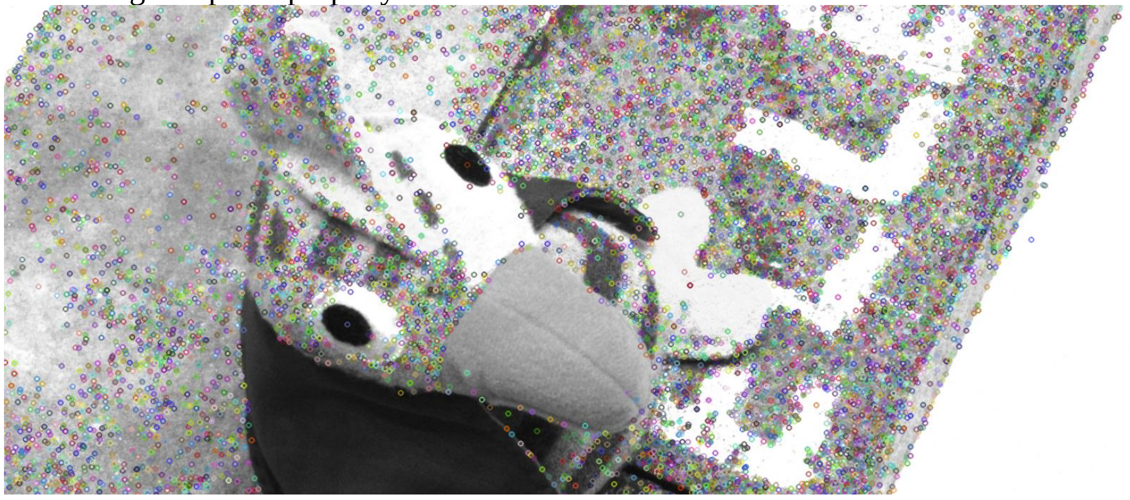
I first output every input image to ensure that the images were being read in correctly.



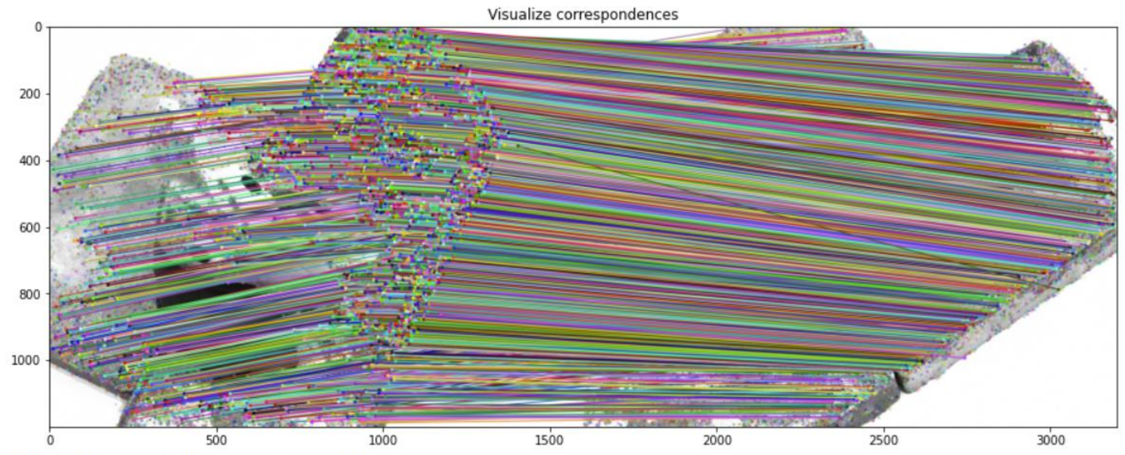
Next, I converted every image to grayscale and computed the keypoints and descriptors, and drew the keypoints on the grayscale image.



The visualization of the keypoints were hard to see because of how small the images above were, so I output a bigger version of the first image to ensure that the keypointsw were being computed properly.

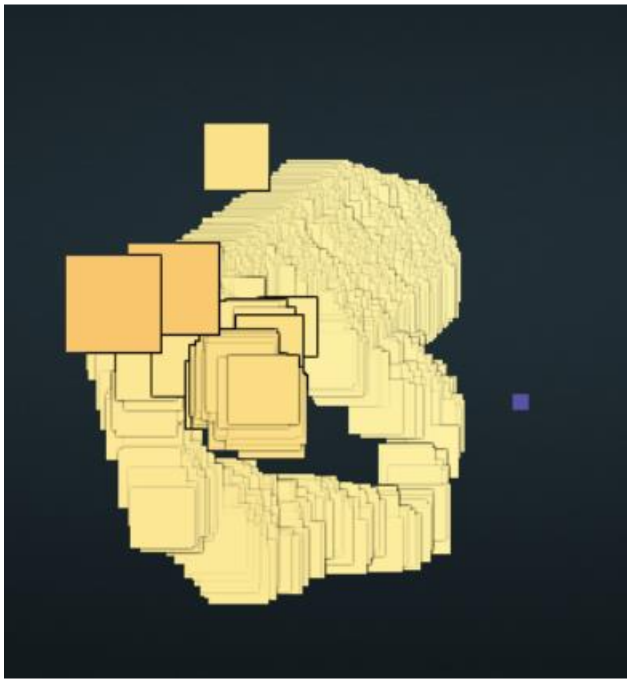


Next, for the first pair of adjacent images, I obtained the correspondences between the two images using KNN matching. I used a k value of 2 so that the ratio test could be used later on to obtain better correspondences. For the ratio test, I used a ratio of 0.65, because that is the ratio that gave me results with not too many outliers. With the above tuning, I obtained the correspondences between the first and second image that looked like below:



```
Number of correspondences derived: 3522
```

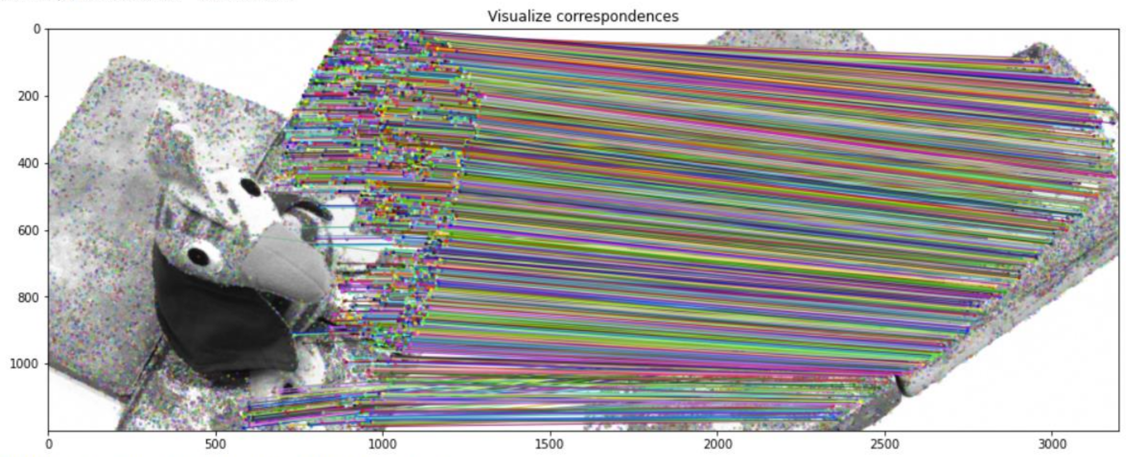
I then triangulated the correspondences using the provided intrinsic and extrinsic matrices to obtain the below visualization:



At this point, it did not really look like much, because this was only the correspondences between the first and second image, as well as the fact that there were still many outliers among these correspondences.

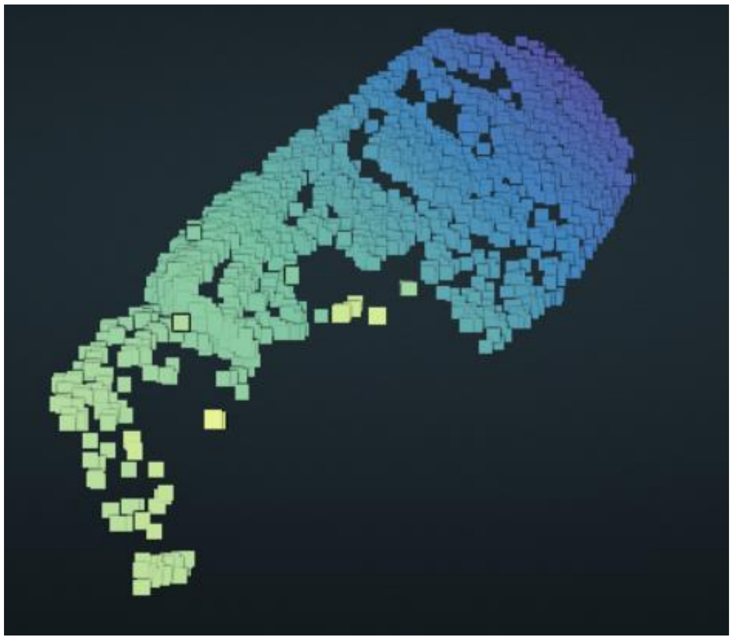
After that, I calculated the fundamental matrix using the given intrinsic and extrinsic matrices. I then used the fundamental matrix to validate the correspondences obtained from before, so that a large amount of outliers could be removed. I set the threshold to 1% of the image's width. By doing this, more than a quarter of the correspondences were removed (mostly outliers), and we obtained the below correspondences.

```
Correspondence after validation:
```



```
Number of correspondences after validation: 2612
```

After doing this, I revisualized the correspondences, and it was starting to look much more like the base of the object, which is what I expected from the correspondences above.



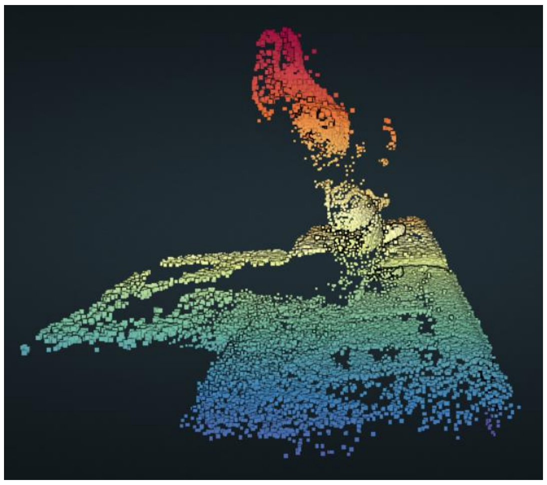
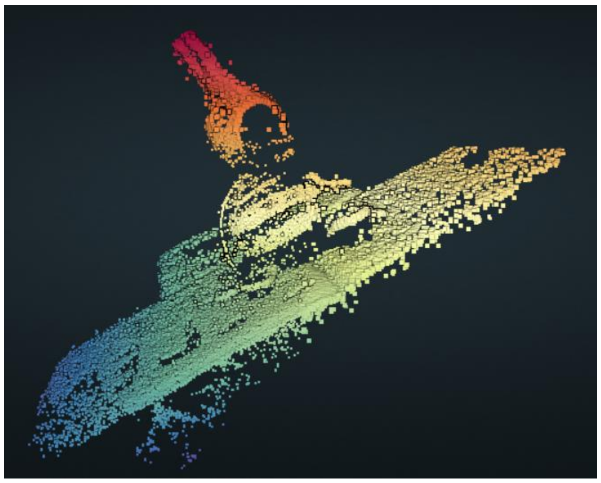
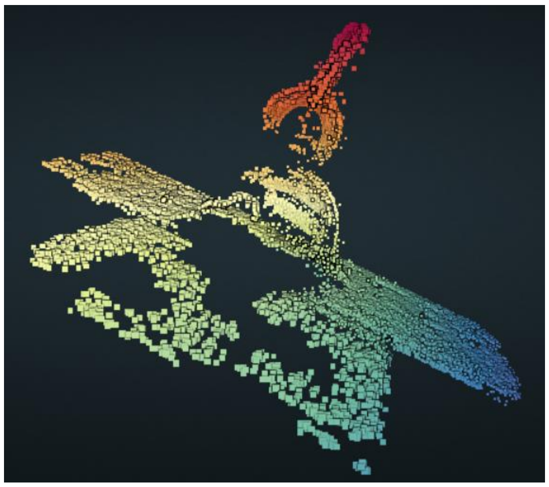
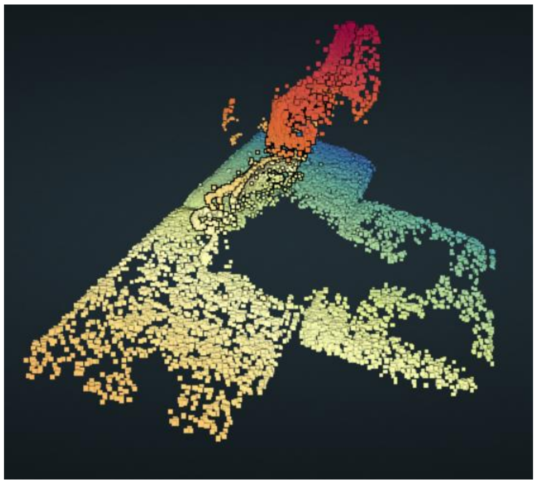
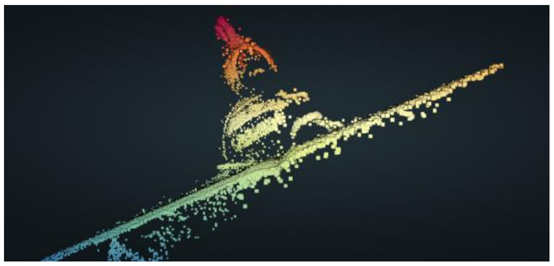
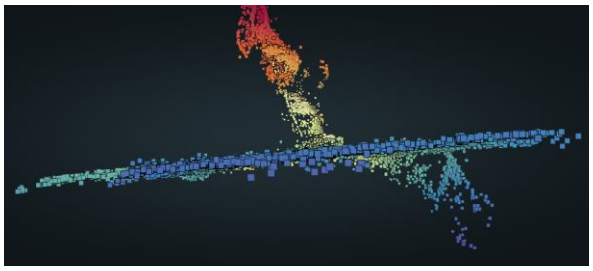
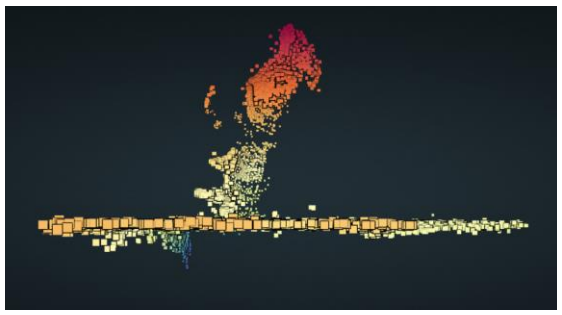
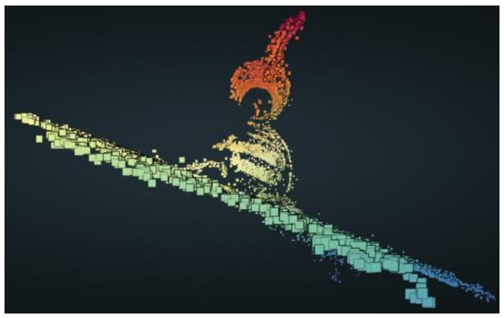
Following that, I redid the above steps with every pair of adjacent images, obtaining 50 pairs from the 49 images and visualized the correspondences of all these pairs.

It seemed like this result was even worse than before, but this was because there were more outliers due to the fact that we had more correspondences, and the object was hidden among those outliers.

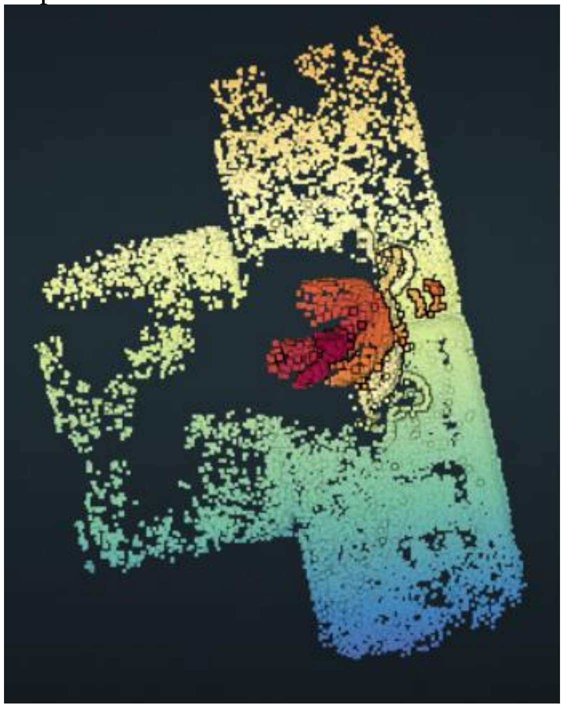
The final step was to remove these outliers. Before doing that, I first converted the points to a PointCloud object, and down sampled the points using voxel down sampling. I set the voxel_size to 0.02 so that I could remove some points for faster computation, but not to the point that important keypoints were removed. Finally, I used statistical outlier removal to remove most of the outliers. I set the number of neighbors to 50, and std ratio to 0.1 to obtain a good result.

The following pages are a compilation of the final result from different angles:

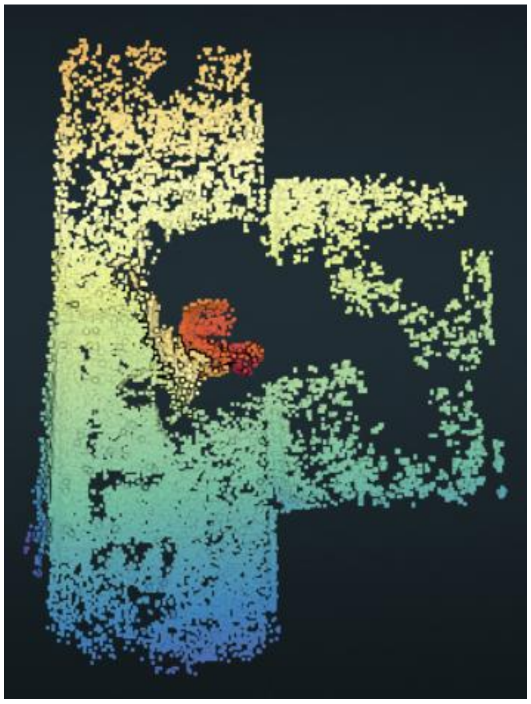
Side view:

Top view:



Bottom view:



Best view (In my opinion)