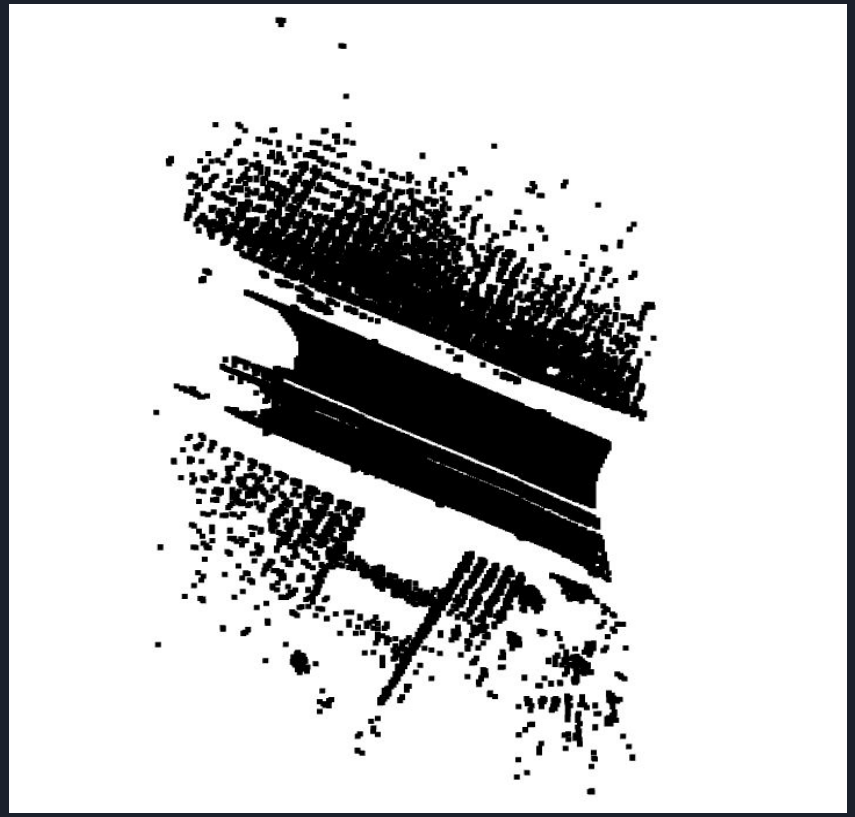
Object Detection in Point Cloud: Poles

-Chris Tsai
-Anuj Karnik

Approach/Methodology

- Point cloud data first needs to be converted from latitude-longitude-altitude coordinates to cartesian coordinates.
  - We use the “point_cloud.fuse” file that is given in the final project files. The .fuse file is similar to an Excel sheet file, and has columns of latitude-longitude-altitude.
- The data is then filtered so that only points that are likely to be on poles are separated.
  - The Open3D library is used to create point cloud objects and analyze the data.

Approach/Methodology



Raw Point Cloud Data from Open3D

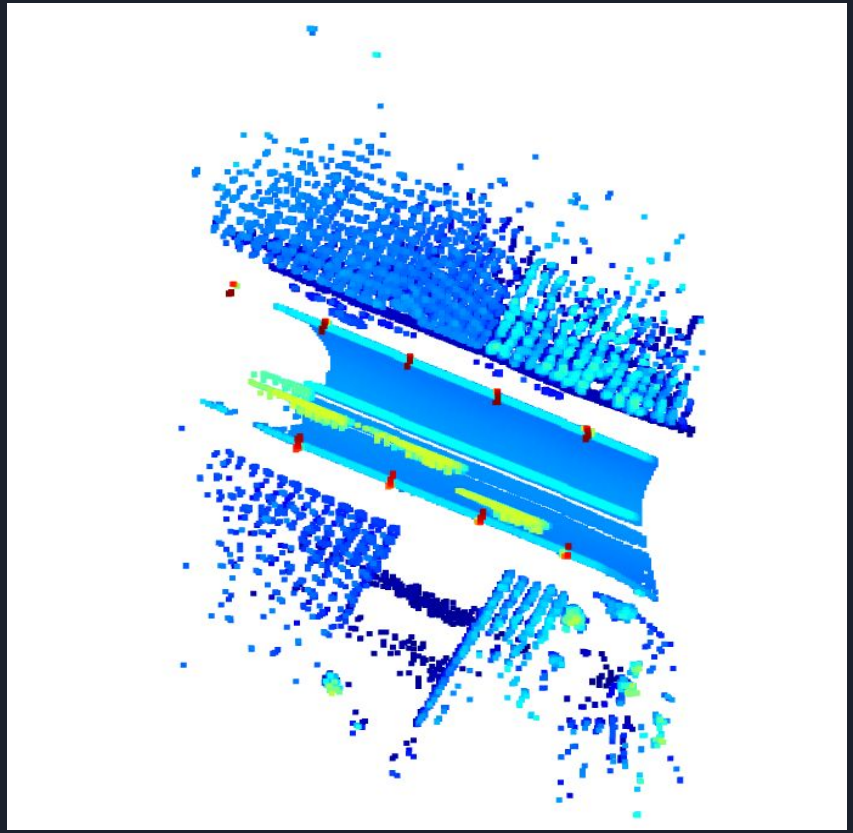
Approach/Methodology

- We next downsample data for efficiency.
  - Downsampling means reducing the number of points in the point cloud and getting rid of outliers.
  - We use voxel_down_sample, uniform_down_sample, and statistical_outlier_removal functions from Open3D.
- To find the poles, we use a planar fitting algorithm.
  - This involves projecting points into x-y plane and using height data to select the poles.
  - Since there's no trees or other tall objects, this method selects only pole data.

Approach/Methodology

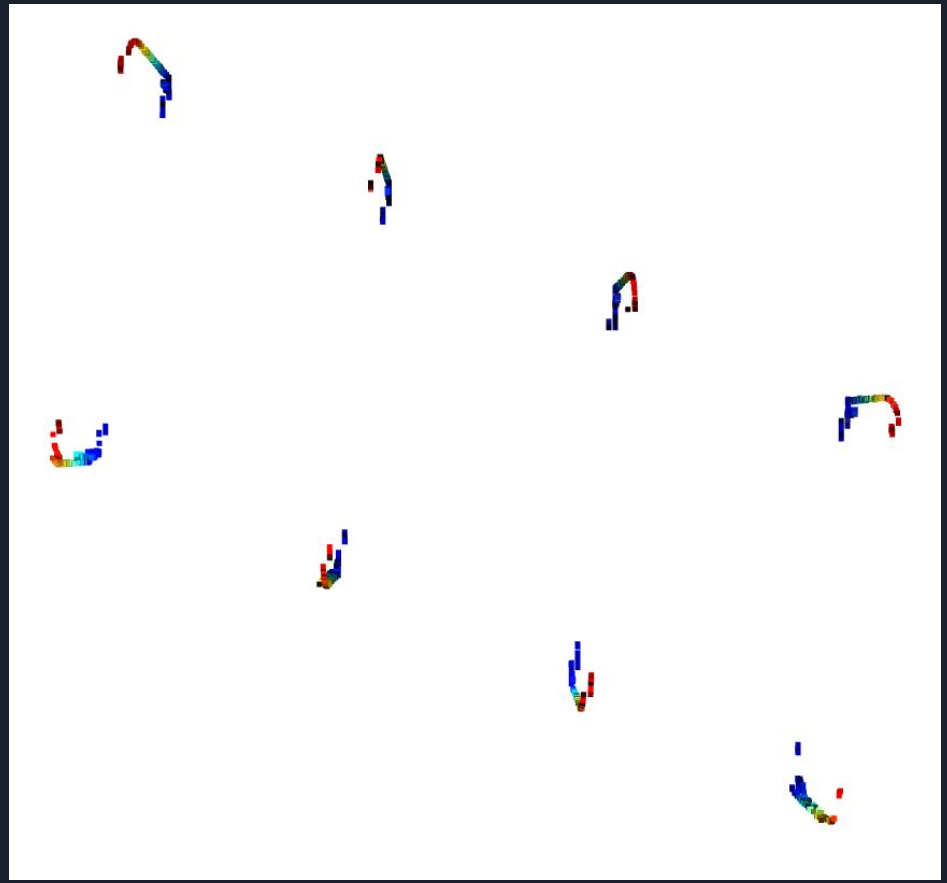
- Finally, we use cluster segmentation of light poles from the data.
  - This means assigning each point to a different cluster per pole.
  - We used k means clustering on x-y coordinates.

Approach/Methodology



Results after K means clustering

Results



Conclusion

- We our algorithm was able to successfully detect and highlight poles (Though we were not able to detect 1 pole in the extreme left corner)
- We were able to output the segregated data to separate files so that it can be used for other applications.
- For future work, we need to work on taking another approach for detecting the poles in the extreme corners as they have sparse points on them and hence are missed by the clustering algorithm.