3D Object Segmentation

S. Bardeewa • C. Ferris • S. Hendrickson • J. Schiffer • M. Whiteside

Introduction

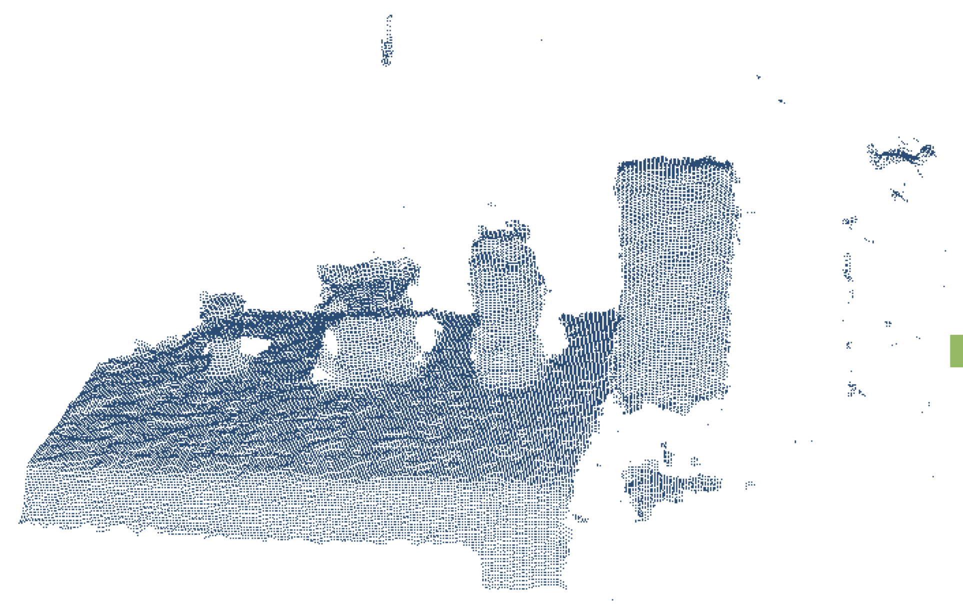
There are two known paths in the area of object segmentation: one using 2D imagery, and one using 3D depth imagery. We chose to use 3D depth imagery because of the recent influx of cheap stereo depth cameras with color like the Intel Realsense R200, and because the dimensions of perceived objects will be easier to calculate with depth data. At a high level, our process for object segmentation is as follows:

1. Convert the 3D depth imagery to a point cloud.
2. Process the point cloud and segment out the objects.
3. Identify the robot hand among the segmented objects.
4. Choose the most appropriate object to pick up.
5. Verify that the object is not the robot's hand.
6. Expose results to the robot control software.

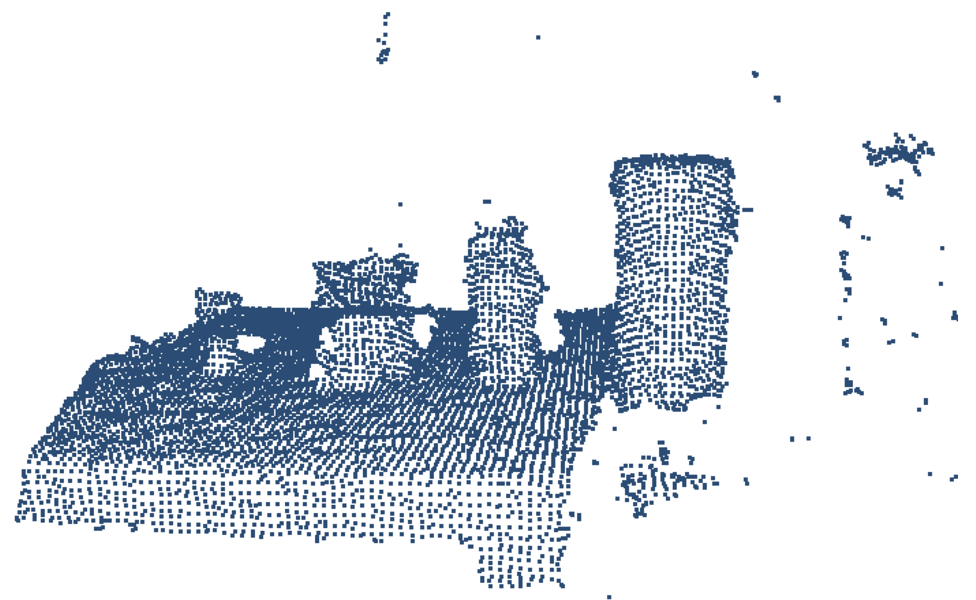
We decided to use the Point Cloud Library (PCL) for our point cloud processing and the Robot Operating System (ROS) for modularizing the code so it can work with different robots and in different applications.

Some other stuff

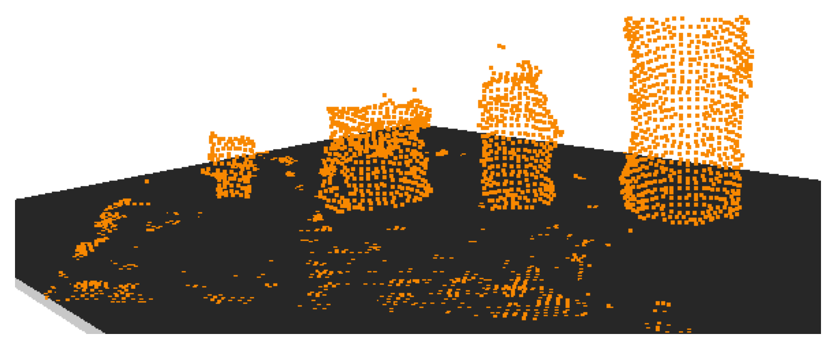
Segmentation Process



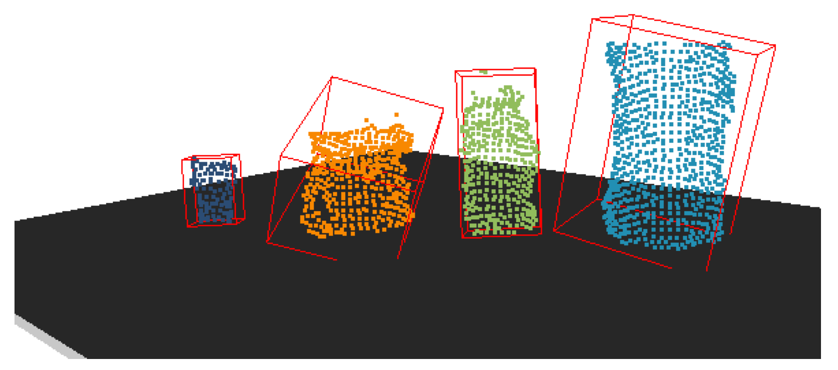
Stage 1: get the raw image from the camera



Stage 2: downsample the image, i.e., combine nearby points to reduce amount of data to process

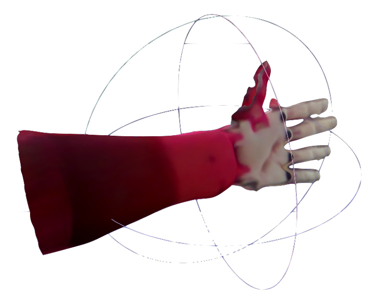


Stage 3: locate the plane, remove everything below it

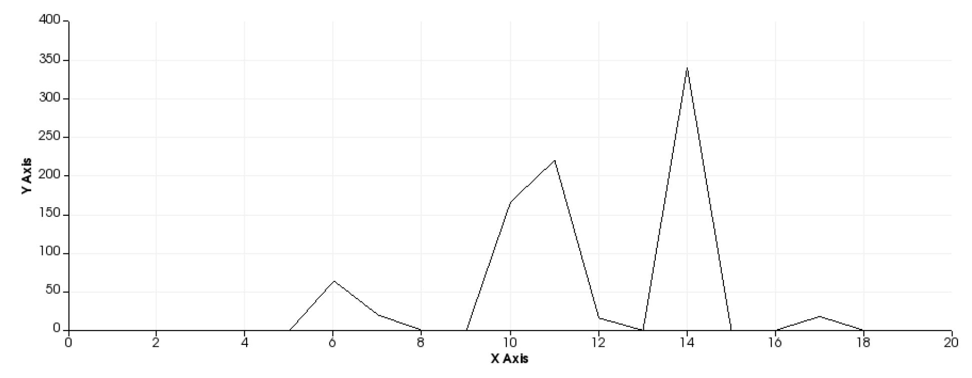


Stage 4: clusterize remaining points, calculate their bounding boxes

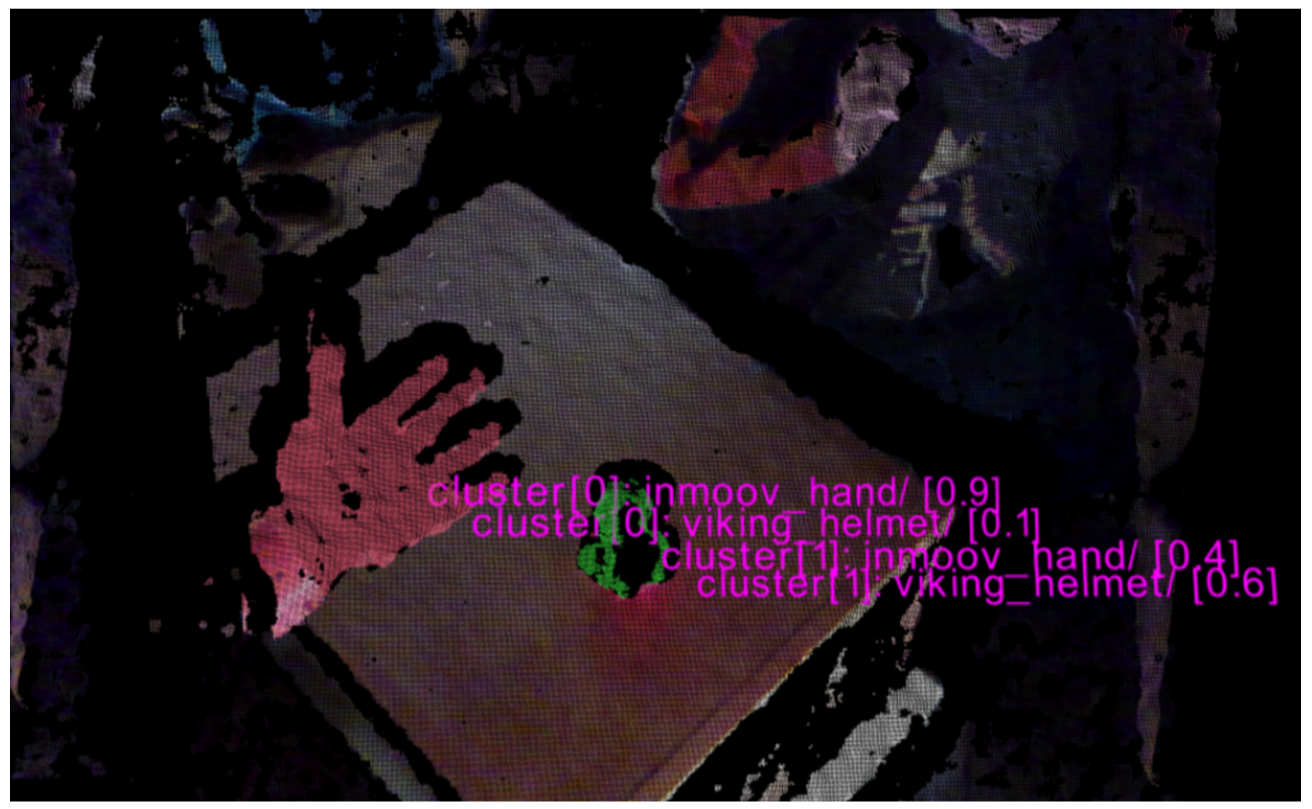
Hand Recognition/training (fix)



The robotic hand is first scanned with a depth camera, such as the Intel Realsense R200™, by revolving the camera around the hand. The output of this process is a 3D CAD model, which is ideally in the .ply format. This .ply file is then fed into a 'virtual scanner' utility, which takes point cloud data from multiple viewpoints around the hand.

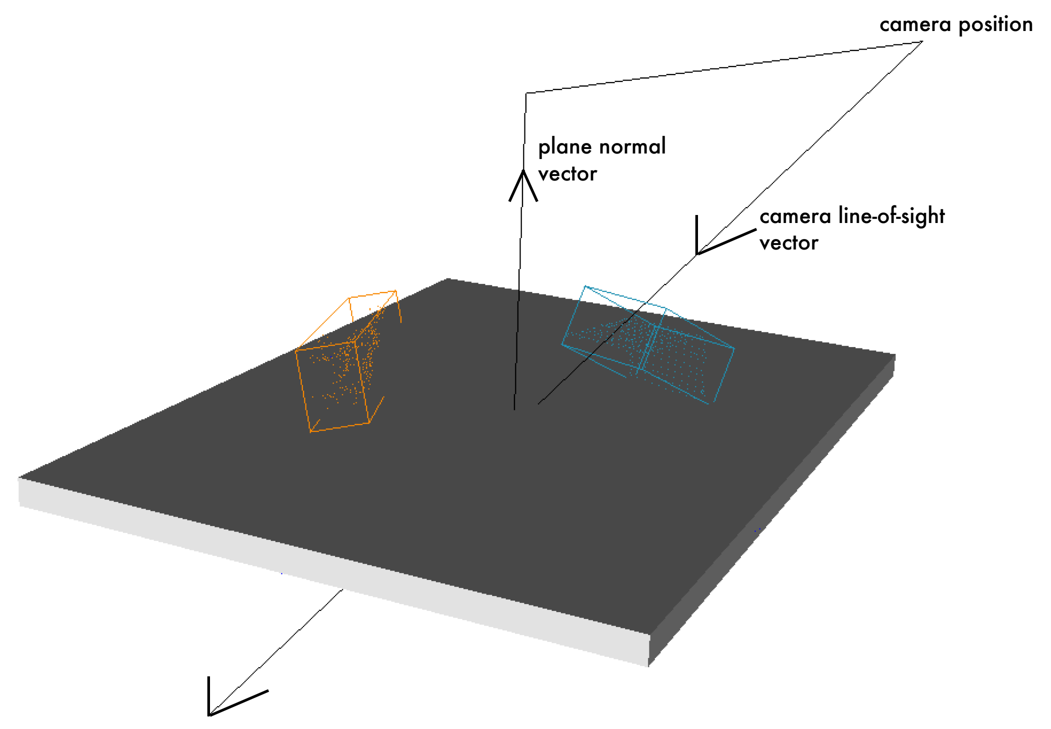


The point clouds generated by the virtual scanner are then analyzed and a histogram is generated for each of the viewpoints.


cluster[0]: inmoov_hand/ [0.9]
cluster[0]: viking_helmet/ [0.1]
cluster[1]: inmoov_hand/ [0.4]
cluster[1]: viking_helmet/ [0.6]

These histograms are then imported into a database which can be searched quickly to find the closest match against one of the point clusters from the scene. The cluster with the lowest 'distance' is then selected as the hand.

Calculating Camera Height Above the Plane



The intersection of the camera line-of-sight vector with the plane can be solved using standard linear algebra, and from there, it is straightforward to calculate the angle of the line-of-sight with the plane, and thus the height of the camera above the plane. This is done in order to simplify the coordinate system in which the robot operates.

Acknowledgements

We would like to thank:
- Dave Shinsel, Intel
- Matthew Curfman, Intel
- Dr. Marek Perkowski, PSU ECE Dept.