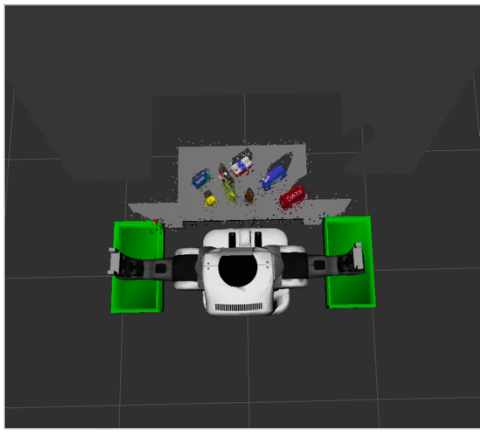
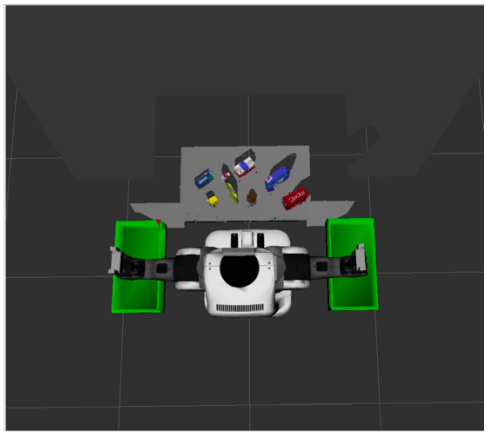
PICK AND PLACE PROJECT

The following sections illustrate the workings of each stage of the perception pipeline assembled for the Pick and Place Project.

STATISTICAL OUTLIER FILTERING

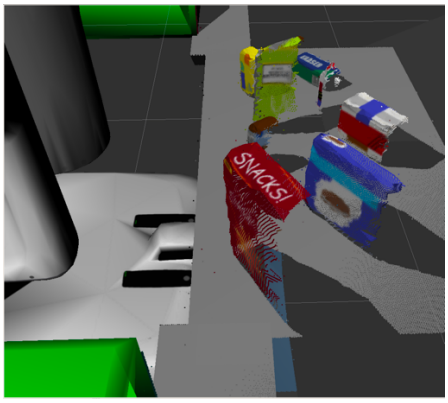
The point clouds captured by the PR2 robot's RGB-D camera are corrupted by outlier noise. To remove this noise a *statistical outlier filter* was used to remove any point whose average distance to $k = 10$ neighbors exceeds the global average of neighbor distances by 0.5 standard deviations. The images below illustrate the point cloud before (left) and after (right) application of the statistical outlier filter. Notice the removal of noisy points around object and table edges.

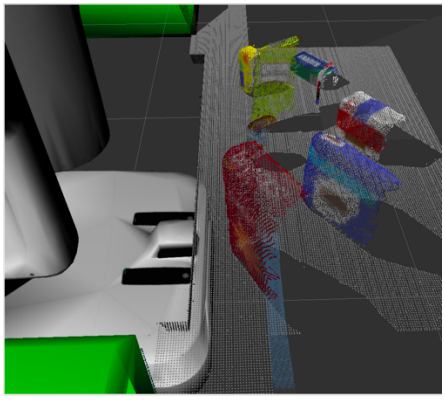




VOXEL GRID DOWNSAMPLING

The spatial resolution of the point cloud generated by the PR2 robot's RGB-D camera is much higher than required by our perception pipeline. Consequently, we spatially average all pixels with a volume of 0.5x0.5x0.5 cm^3 using a *voxel grid downsampling filter*. The images below illustrate the point cloud before (left) and after (right) the application of the downsampling filter.



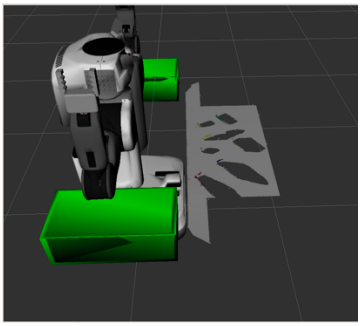


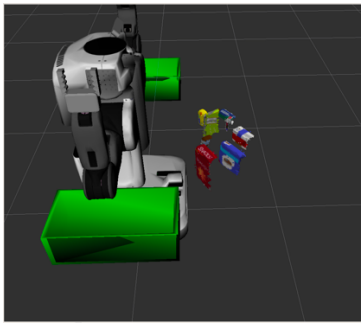
PASSTHROUGH FILTER

Since we know the objects of interest are located in a specific region of the workspace, we can ignore all points in the point cloud whose coordinates fall outside that region. This is accomplished using a *pass-through filter*. In this project only points with z-coordinate in the range 0.60-1.1 and x-coordinate in the range 0.35-1.1 where allowed to pass through the filter.

RANSAC PLANE SEGMENTATION

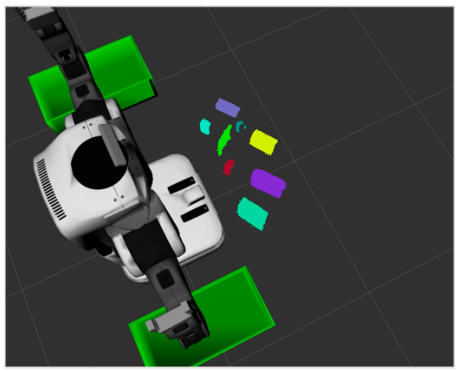
To isolate the points belonging to the table from those associated with objects, the *RANSAC* algorithm was used to find the best plane that fits the point cloud. The idea being that points on or close to the plane are table points while those far away are associated with "objects". The RANSAC algorithm was run with distance threshold of 0.0125. The images below illustrate segmentation of points belonging to the table (left) from those belonging to objects (right).





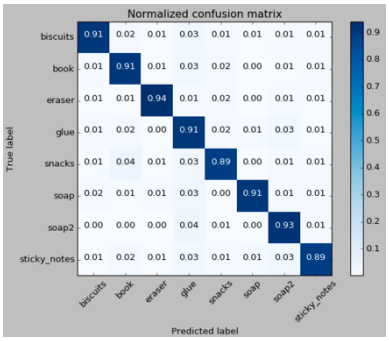
CLUSTERING

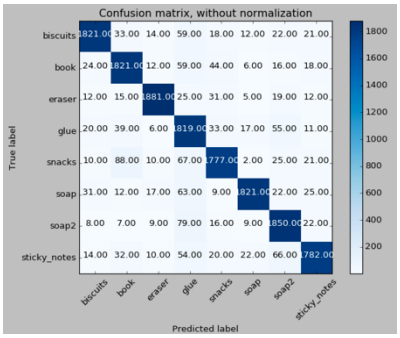
The next step involves using *Euclidean Clustering* (DBSCAN algorithm) to cluster the points the outlier points (object points) into groups where each group is a distinct object (e.g. an eraser or a book). The images below illustrate the results of applying the Euclidean clustering algorithm with cluster tolerance 0.045 and minimum and maximum cluster sizes set to 10 and 10,000 respectively.



CLASSIFICAITON

Finally, the points within each cluster are assigned a label using a *support-vector machine* (SVM). The SVM was trained (with C=0.1) on 16,000 feature vectors extracted from randomly generated poses of each of the 8 objects (2000 feature vectors were computed for each object). Each vector consists of the concatenation of 42-bin histograms of surface normals and colors (one histogram for each of the channels of the HSV color space). The SVM mean accuracy for 5-fold cross-validation was 91 +/- 0.01. Moreover, below are the confusion matrices produced by the script *train_svm.py*.





The images below illustrate the SVM labeling of objects in each of the test scenes. First, we see test scene #1 (left-most panel) with the robot correctly identifying 3/3 of the test objects (biscuits, soap and soap2). Next, we see test scene #2 (middle panel) with the robot correctly identifying 5/5 of the test objects (biscuits, soap, soap2, glue, and book. Finally, we see test scene #3 (right-most panel) with the robot correctly identifying 8/8 of the test objects (snacks, biscuits, soap, soap2, glue, eraser, and sticky notes).