**Report on the Perception Project:**

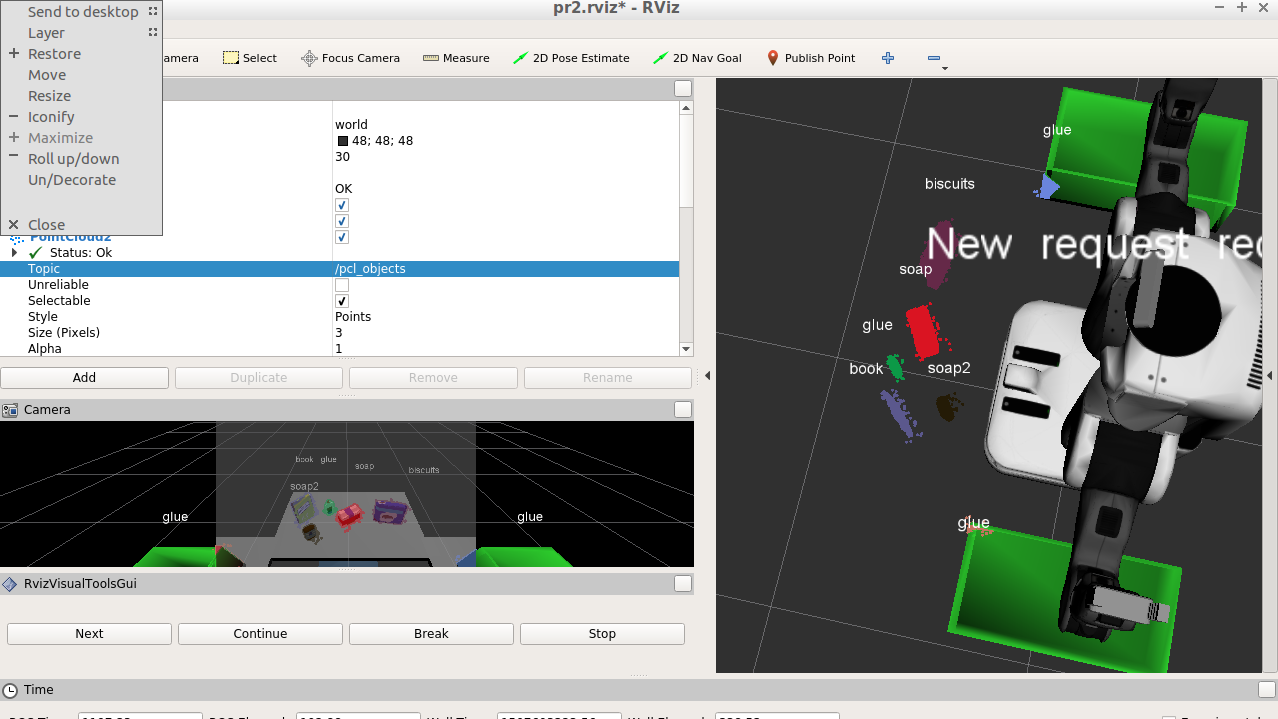
The perception project is divided into 4 parts: The first 3 parts are exercises and the next part is applying all the exercise content on a robot. Please note that my code is available in the project\_template1.py.

**Part 1:**

The first part is about the camera calibration, filtering and segmentation to analyze 3D point cloud data. For this we apply noise filtering using statistical outlier filter. And then continued to voxel grid down sampling and then pass through filter and then to the RANSAC plane segmentation and then to extract inliers and outliers. These filters are applied to filter out the objects in the plane that are needed for further segmentation which in this case are the objects on the table and the table which is not need is filtered out.

**My observations in this part of code:**

If we apply pass through filter through z axis it is enough for the exercise 1, but when it comes to most complex environment like the pr2 environment and the testworld\_2 and 3 where there are more objects, the pass-through filter is applied through x, y and z axes to filter out the table as well as bins that are attached to the table. So that I got the exact number of clusters that are detected. Before applying that filter, I got 2 extra clusters that are not useful. This is illustrated in the figure below where two extra labels glue are pointed.



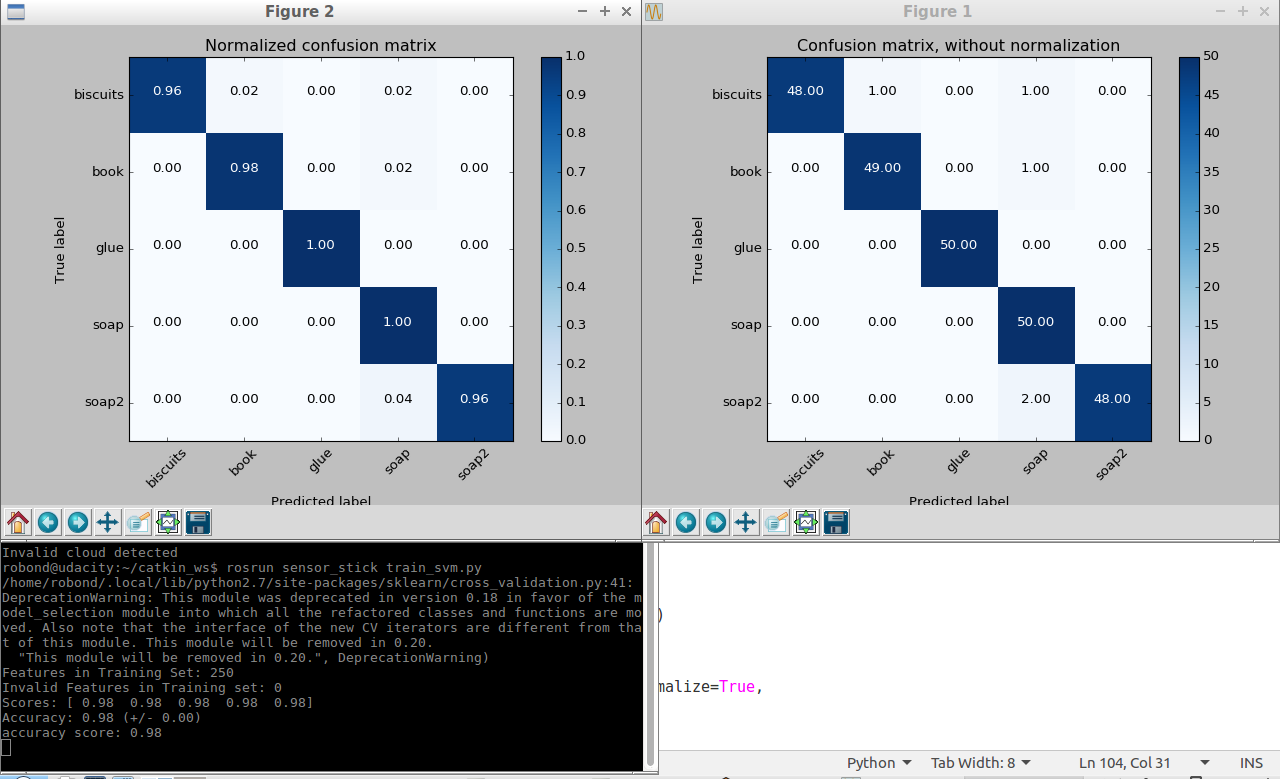
**Part 2:**

This part is based on Euclidean clustering, where there are three parameters 1. Cluster tolerance, min cluster size and max cluster size. Small change in the cluster tolerance brought some big change in the clusters and min max size of the clusters are also very important for the clusters not to overlap.

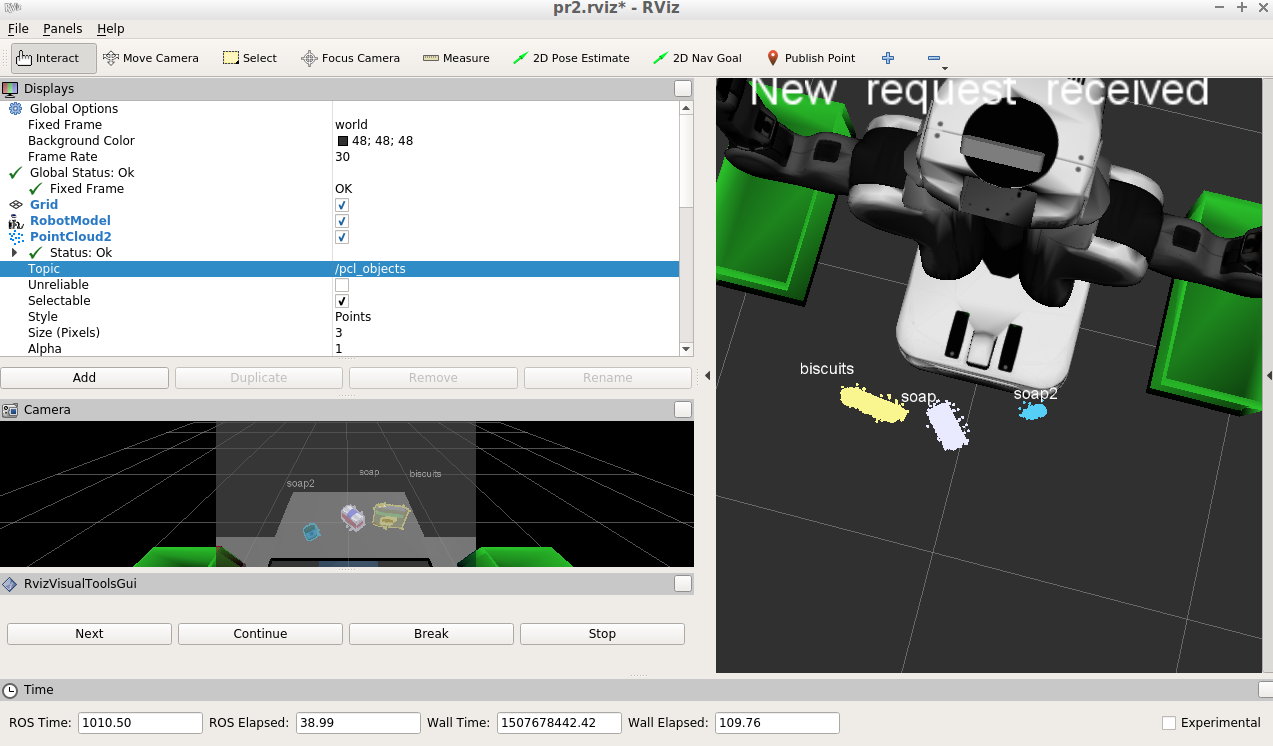
And then the cluster cloud which is a PCL (Point Cloud Library) data is converted to ROS cloud, and published as ROS messages.

**Part 3:**

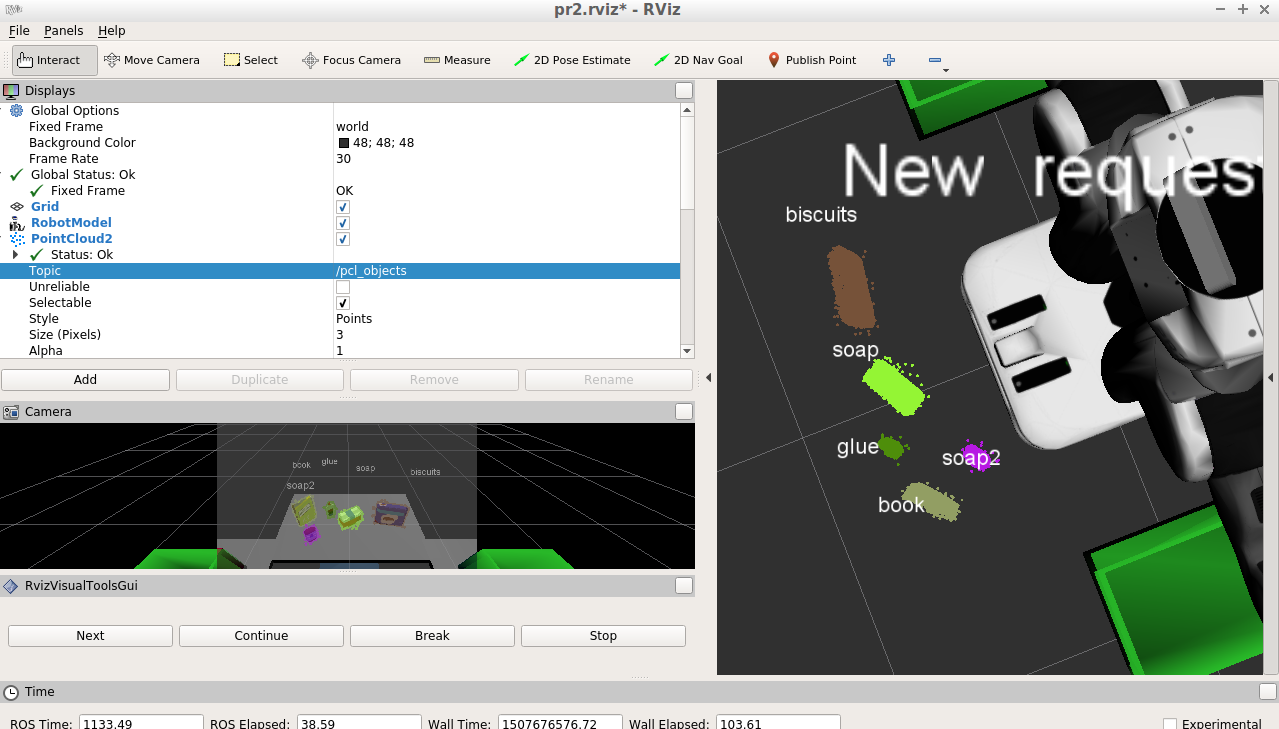
In this part object recognition steps are performed. First objects in the scenes are fed to SVM for it to train. The more the number of features, the correct label is given to it. I chose 50 features for each object and got 94-98 % accuracy in training of SVM. When these labels are featured in the RViz, the accuracy reduces little bit but in my case, In the test world\_2 where there are 5 objects and test\_world 3 where there are 8 objects I trained SVM by giving 50 features per object and the clusters detected are 5/5 and 7/8 respectively. By this I came to know that the more the number of features you capture, the better the accuracy is.



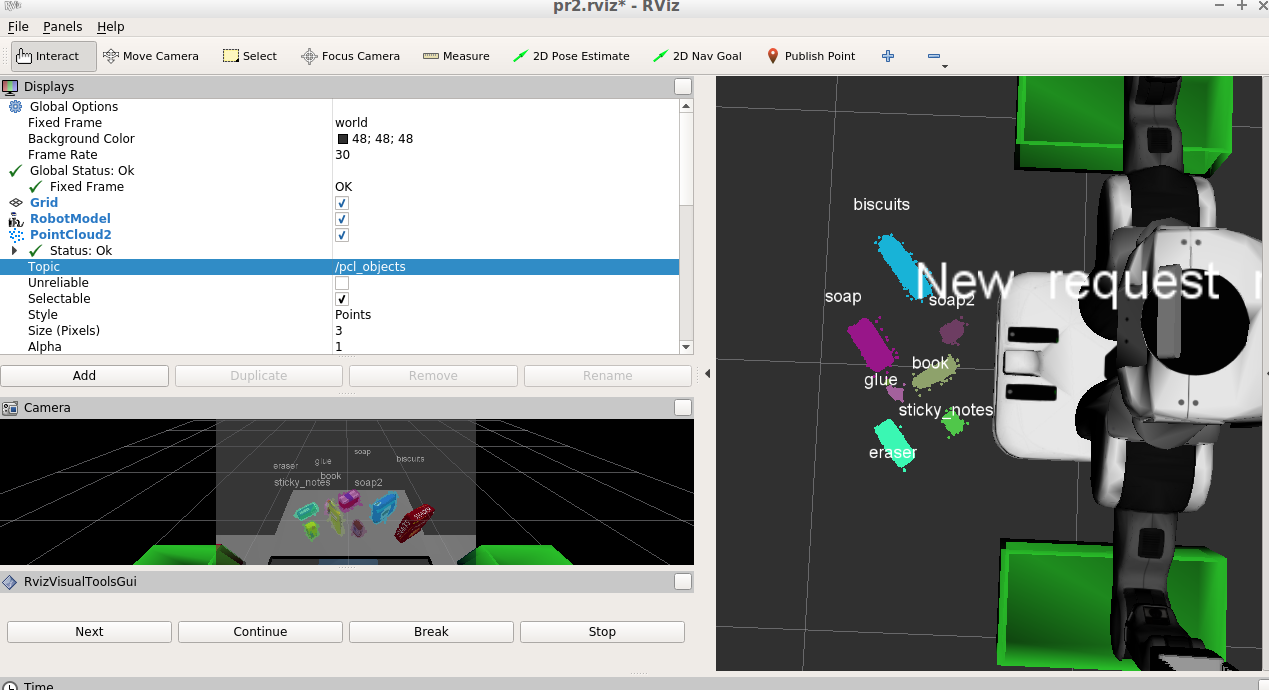
The above figure shows the 98% accuracy for the test world 2.



The above figure shows the clusters and the labels of the objects recognized with clusters. In this case 3/3 objects are correctly recognized in test world 1.



The above figure shows the clusters and the labels of the objects recognized with clusters. In this case 5/5 objects are correctly recognized in test world 2.



The above figure shows the clusters and the labels of the objects recognized with clusters. In this case 7/8 objects are correctly recognized in test world 3.

**Part 4**:

In this part to produce the output. yaml, which contains the list of objects detected, the position of the object (pick and place), and the right or left arm of the PR2 robot to be used to pick and place the object and the drop box name. With the details drawn all the variables above, the details are drawn to the output.yaml file.

The improvements I can make in the projects are mostly in getting the 100 % accuracy to detect the objects and the challenge part which I did not attempted. There are lot of things that I learned through the project that would help me in the future.