

Udacity Robotics Project 3 Writeup – Perception

Exercise 1. Filtering pipeline and RANSAC plane filtering

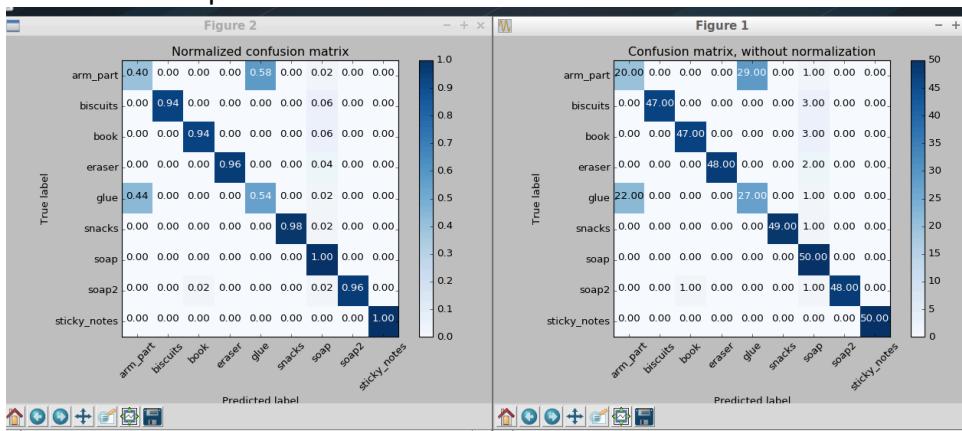
- Used filters below to take tabletop.pcd and output a filtered point cloud
 - o VoxelGrid Downsampling Filter
 - o PassThrough Filter
 - o Extract Indices Filter
 - o RANSAC Plane Fitting (Random Sample Consensus)
 - o Statistical Outlier Removal Filter

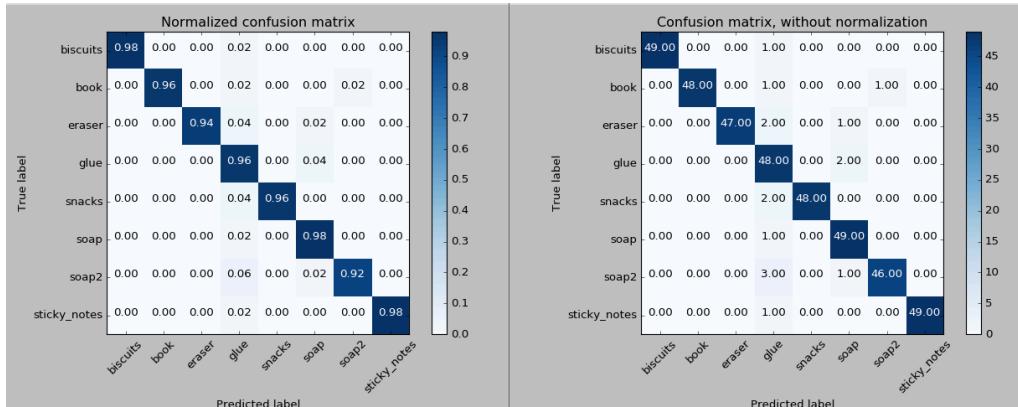
Exercise 2. Clustering for segmentation pipeline

- Subscribe to /sensor_stick/point_cloud topic
- Apply Euclidian clustering on filtered point clouds
- Publish clusters to separate pcl_table and pcl_objects topics

Exercise 3. Feature Extraction and SVM classifier training for object recognition

- Use object shape and color to classify and recognize objects, and apply a label to them
- capture_features.py
 - o I tuned various parameters before settling on training my model on 50 poses per object (8 objects from 3 worlds, and 1 ‘arm_part’ object), used a linear kernel on my SVM and achieved about 85% accuracy overall. The “arm_part” and ‘glue’ were frequently miscategorized as each other, so in the future I would exclude arm_part and train only on the 8 objects. The reason I used arm_part was because without it, my sensor was recognizing more objects than there existed in the world, as it was considering the robot arm to be an object.
 - o In features.py, I tried different binning schemes before settling on the one I used. The range of color hist values is 0 – 256 and the range of normal hist is -1 to 1.
 - o The first set of confusion matrices below is what I used initially, but it did not attain a passing submission in test_scene_3. Then, I retrained my model without arm_part, to get the 2nd set of confusion matrices, which identified the objects quite well!

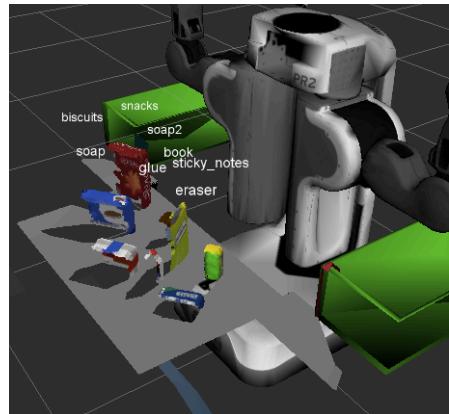




Project!

- The 3 exercises are collected in pcl_callback()
- In pr2_mover, I loop through the list of possible objects, dropbox parameters, and the pick list for each individual world. When an object matches that which is in the pick list, I assign the pick_pose and place_pose for it, and send that request through mack_yaml_dict. If it is not detected, I use default values of (0,0,0) for position and (0,0,0,0) for orientation.
- My terminal outputs the objects that were detected, along with "True" or "False" if it was correctly identified or not. The centroid position gets passed to a new topic which handles the pick and place operation.

Success rate in scenes 1 and 3 (Sorry, forgot to take screenshot of scene 2!) :



1

Identified 3/3

```

/home/robond/catkin_ws/src/Ro...ct.launch http://localhost:11311 - + x
[INFO] [1510522133.614837828, 1749.056000000]: Dropping first 1 trajectory point(s) out of 25, as they occur before the current time.
First valid point will be reached in 0.035s.
[INFO] [1510522133.614837828, 1749.056000000]: Combined planning and execution request received for MoveGroup action. Forwarding to planning and execution pipeline.
[INFO] [1510522133.684136756, 1749.066000000]: Planning attempt 1 of at most 1
[ERROR] [1510522133.735690338, 1749.074000000]: Found empty JointState message
[INFO] [1510522134.455379631, 1749.129000000]: Planner configuration 'left_arm' will use planner 'geometric::RRTConnect'. Additional configuration parameters will be set when the planner is constructed.
[INFO] [1510522134.466987648, 1749.131000000]: RRTConnect: Starting planning with 1 states already in datastructure
[INFO] [1510522134.603248819, 1749.135000000]: RRTConnect: Created 5 states (3 start + 2 goal)
[INFO] [1510522134.603360403, 1749.135000000]: Solution found in 0.135850 seconds
[INFO] [1510522134.949477931, 1749.153000000]: SimpleSetup: Path simplification took 0.346002 seconds and changed from 4 to 25 states
[WARN] [1510522136.524165471, 1749.266000000]: Dropping first 1 trajectory point(s) out of 25, as they occur before the current time.
First valid point will be reached in 0.589s.

```

2

Identified 4/5

```

/home/robond/catkin_ws/src/Ro...ct.launch http://localhost:11311 - + x
[INFO] [1510514739.224781293, 1890.258000000]: RRTConnect: Starting planning with 1 states already in datastructure
[INFO] [1510514739.374668851, 1890.272000000]: RRTConnect: Created 4 states (2 start + 2 goal)
[INFO] [1510514739.376112760, 1890.273000000]: Solution found in 0.159010 seconds
[INFO] [1510514739.556217728, 1890.282000000]: SimpleSetup: Path simplification took 0.179994 seconds and changed from 3 to 33 states
[WARN] [1510514739.896775048, 1890.299000000]: Dropping first 1 trajectory point(s) of 33, as they occur before the current time.
First valid point will be reached in 0.545s.
[INFO] [1510514739.284275118, 1894.450000000]: grasp_pose: position:
  x: 0.634
  y: 0.13
  z: 0.96
orientation:
  x: 0.14
  y: 0.693
  z: -0.14
  w: 0.693
[INFO] [1510514775.615705945, 1894.474000000]: Incorrect pick_pose for: glue

```

3

Identified 8/8 ... Sorry, I forgot to take screenshot of terminal output 😊

Roadblocks

- Major hurdles in updating the pcl library, for which I had to download a new VM
- Found a silly error in my for loop which was causing me to recognize the first object and none of the rest, due to which I wasted a lot of time and energy...

Improvements for the future

- Collision avoidance during pick and place task
- Build an even better classifier by trying different kernels, training sizes, binning criteria