Project: Perception Pick & Place

Rubric Points

Here I will consider the rubric points individually and describe how I addressed each point in my implementation.

Writeup / README

1. Provide a Writeup / README that includes all the rubric points and how you addressed each one. You can submit your writeup as markdown or pdf.

You're reading it!

Exercise 1, 2 and 3 pipeline implemented

1. Complete Exercise 1 steps. Pipeline for filtering and RANSAC plane fitting implemented.

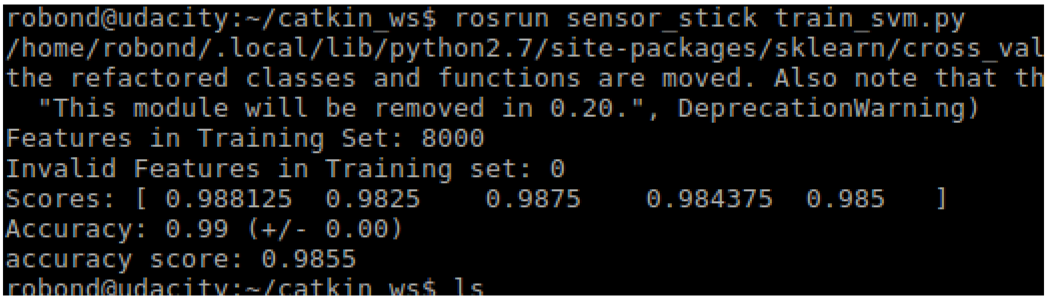
- Convert the incoming ROS cloud to PCL format.
- Statistical Outlier Filtering is done with std_dev=0.3 to reduce noise. My experiment shows it works.
- Statistical Outlier Filtering with LEAF_SIZE=0.005. It shows good balance of reducing computational effort without the loss of too much detail.
- PassThrough Filter on z-axis is to remove other parts of PCL except the area of interest. Additional Filter on y-axis is to remove drop-boxes from cloud.
- RANSAC Plane Segmentation is to identify the table. Hence, it splits the cloud into (1) the table by itself, and (2) the objects on the table.

2. Complete Exercise 2 steps: Pipeline including clustering for segmentation implemented.

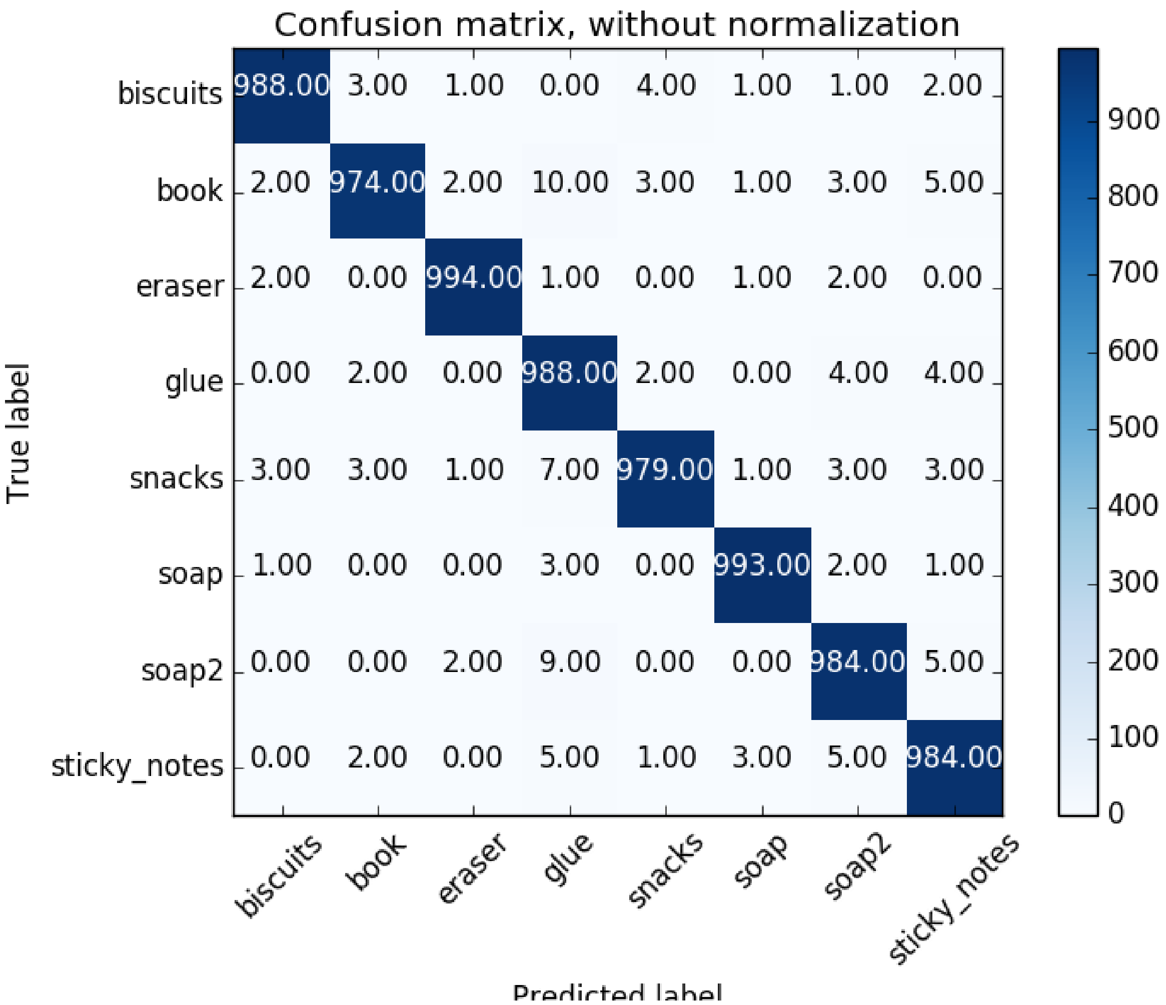
- Euclidean Clustering to do partitioning. Then, apply color to visualize each cluster.
- Next, convert PCL data back to ROS messages, then, publish to various topics for RViz.

2. Complete Exercise 3 Steps. Features extracted and SVM trained. Object recognition implemented.

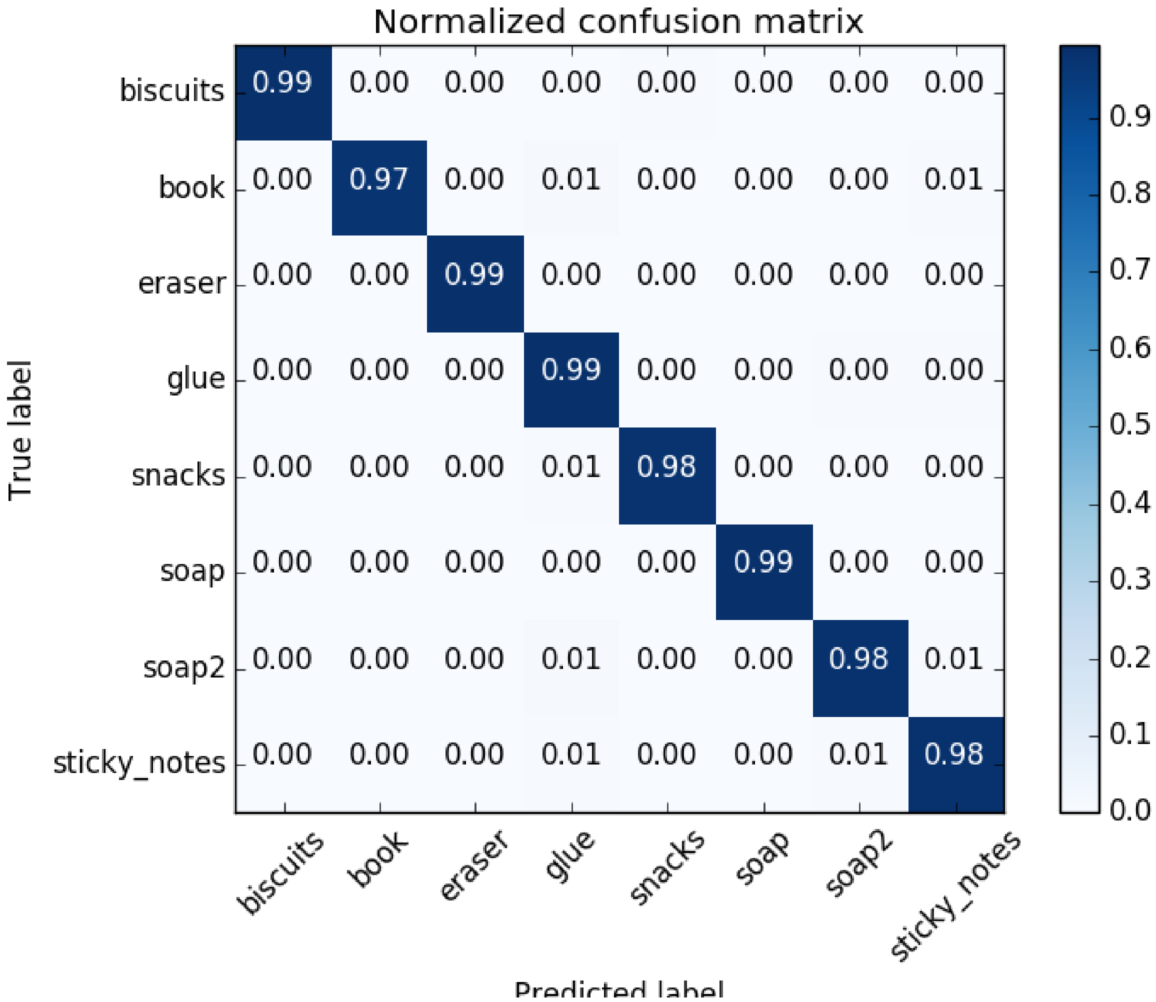
- compute_color_histograms and compute_normal_histograms are implemented as suggestion from lesson with the range [0,255] for color, and [-1,1] for normal. #bins are 32.
- Output of SVM training



```
robond@udacity:~/catkin_ws$ rosrun sensor_stick train_svm.py
/home/robond/.local/lib/python2.7/site-packages/sklearn/cross_val
the refactored classes and functions are moved. Also note that th
  "This module will be removed in 0.20.", DeprecationWarning)
Features in Training Set: 8000
Invalid Features in Training set: 0
Scores: [ 0.988125  0.9825    0.9875    0.984375  0.985    ]
Accuracy: 0.99 (+/- 0.00)
accuracy score: 0.9855
robond@udacity:~/catkin_ws$ ls
```



	biscuits	book	eraser	glue	snacks	soap	soap2	sticky_notes
biscuits	988.00	3.00	1.00	0.00	4.00	1.00	1.00	2.00
book	2.00	974.00	2.00	10.00	3.00	1.00	3.00	5.00
eraser	2.00	0.00	994.00	1.00	0.00	1.00	2.00	0.00
glue	0.00	2.00	0.00	988.00	2.00	0.00	4.00	4.00
snacks	3.00	3.00	1.00	7.00	979.00	1.00	3.00	3.00
soap	1.00	0.00	0.00	3.00	0.00	993.00	2.00	1.00
soap2	0.00	0.00	2.00	9.00	0.00	0.00	984.00	5.00
sticky_notes	0.00	2.00	0.00	5.00	1.00	3.00	5.00	984.00



Finally, I use model in percaption pipeline to identify individual object clusters, then, publish the label to RViz

Pick and Place Setup

1. For all three tabletop setups (`test*.world`), perform object recognition, then read in respective pick list (`pick_list_*.yaml`). Next construct the messages that would comprise a valid `PickPlace` request output them to `.yaml` format.

As all the objects are recognized, pr2_mover() then creates the output *.yaml.

- After grouping parameters into individual variables, here comes the main loop to iterate through the pick list. The pick_pose is get from the list of detected object whose label matches.
- Next, the simple rule based on group color of 'red' or 'green' is used to assign the arm.
- Finally, the test_scene_num is used to produce correct output file name.

I didn't have time to complete the challenge of pick and place. Will spend some time to re-visit after completing the term.