PROJECT 3 – 3D PERCEPTION

Youssef Khaky

AUGUST 31, 2017

Pipeline flow:

1. Input PC in ROS format into the callback function
2. We change it to PC in PCL format. pcl_cloud (stuff that are highlighted yellow is the variable name in my python code, it may not be accurate as I may have changed the variable name after writing this report)

Section 1 Filtering and segmentation (Criteria 1)

3. Statistical Outlier Filtering
   a. outlier_filter is the filter object
   b. cloud_filtered is the filtered output
4. Voxel Grid Downsampling is reduce the number of points to a reasonable number
   a. Vox is the filter object
   b. cloud_filtered is the voxel grid down sampling output

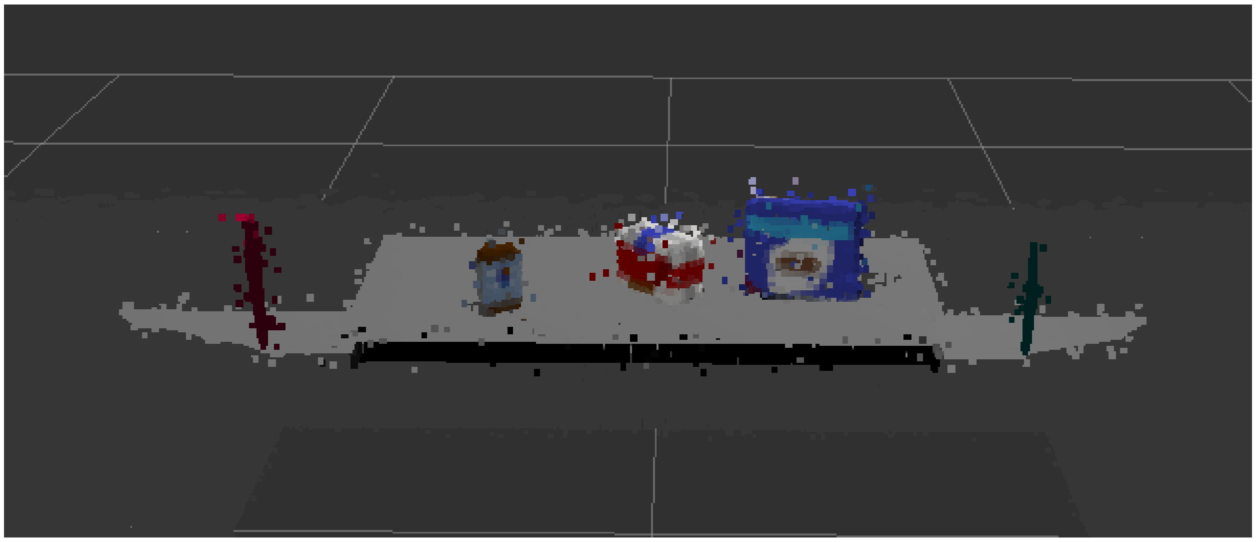


Figure 1 PC after down grid sampling

5. Passthrough Filter
   a. Passthrough is the filter object
   b. cloud_filtered is the filtered output

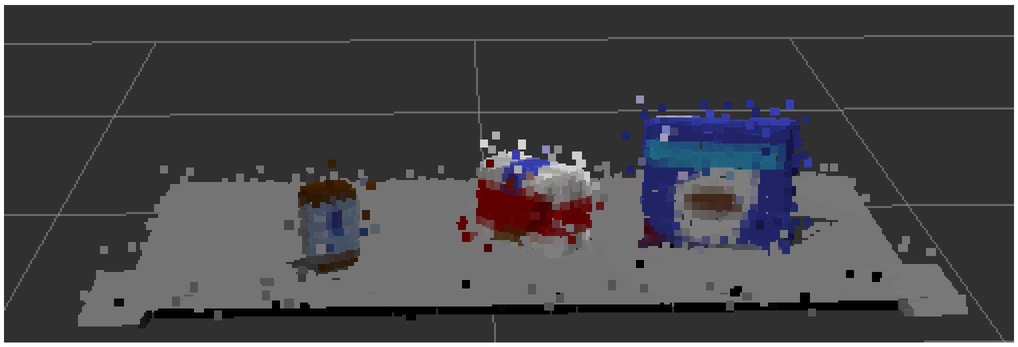


Figure 2 PC after passthrough filter

6. RANSAC Plane Segmentation
    a. Seg is the filter object
    b. We get two outputs, the table and the objects on the table
        i. extracted_inliers is the table
        ii. extracted_outliers are the objects

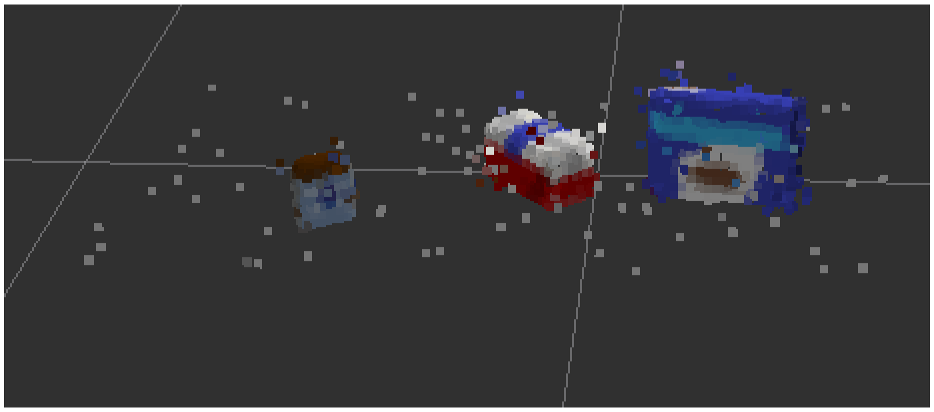


Figure 3 PC after RANSAC segmentation

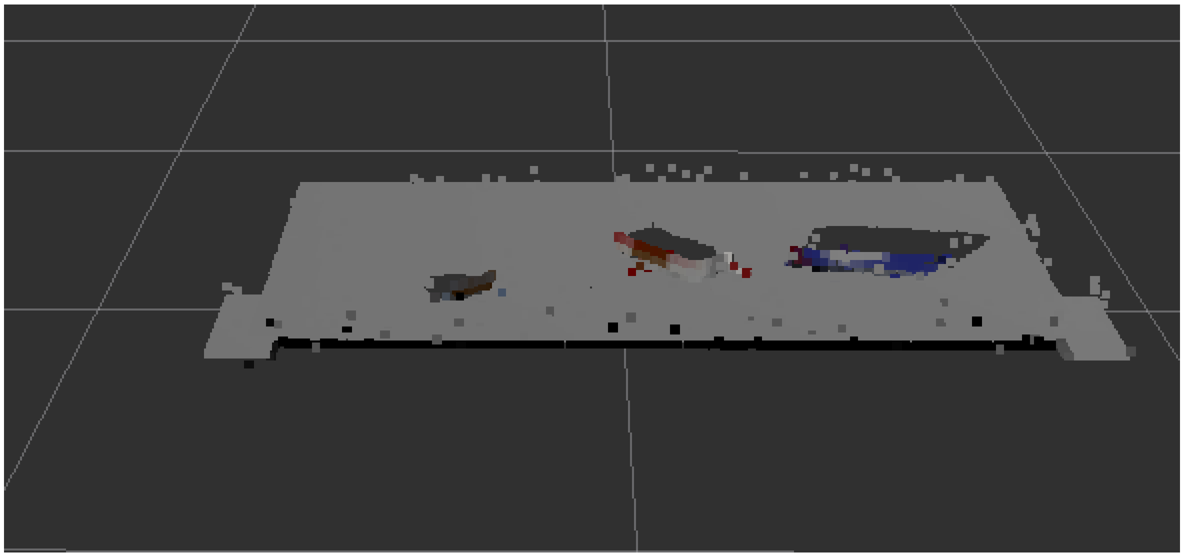


Figure 4 PC of table after running the RANSAC

Section 2 Euclidian clustering

7. Turn all points in the extracted outliers into white points and call that variable white_cloud
    a. Put that into the tree format required, that format is a tree of some sort, call the tree, tree
    b. Create clustering object ec and run it on the cloud
    c. Extract indices for the discovered cluster cluster_indices
        i. I do not know how this is formatted, is this a list of lists for example? [cluster1, cluster2,…, cluster n]??
    d. List of colors for each cluster object cluster_color
        i. What does get_color_list() do? What library is it under?
    e. Create new cloud containing all clusters, each with unique color cluster_cloud
8. Convert extracted_outliers to ROS PC → ros_cloud_objects. Publish that
9. Convert cluster_cloud to ROS PC → pc. Publish that

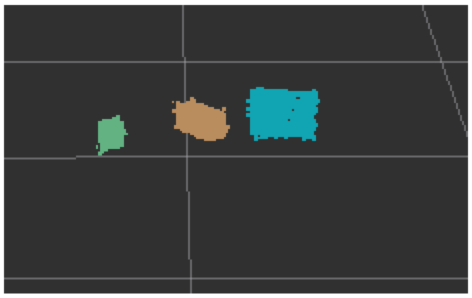


Figure 5 Clustered objects from world 1

Section 3 Cluster classification

10. Loop though cluster_indices and convert each cluster individually to ROS PC
11. Extract its histogram features feature
12. Predict the object prediction and extract its label
13. Add object detected label to detected_objects_labels
14. Add detected object do to list of detected objects do is of type DetectedObject()

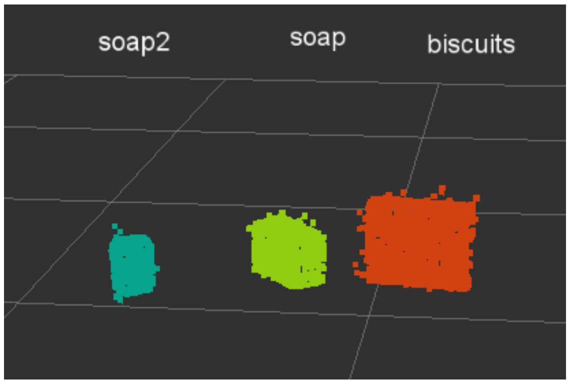


Figure 6 classification of the clustered objects

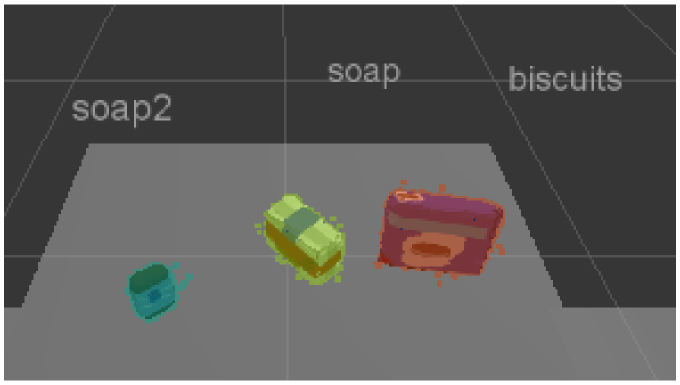


Figure 7 camera point of view

Classification of other worlds is in the appendix.

Confusion matrix can be found in the appendix.

Section .4 YAML

15. For each world output a yaml file which outlines which box an item would go, and the centroid of that item at which the arm would pick the item up

Appendix

RVIZ world2

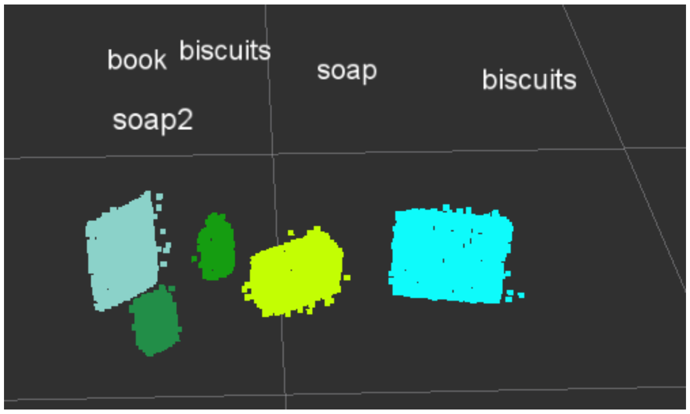


Figure 8 PC of classified objects for world 2

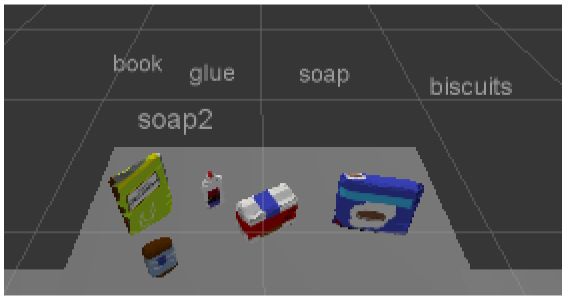


Figure 9 World 3 from camera perspective

RVIZ world3

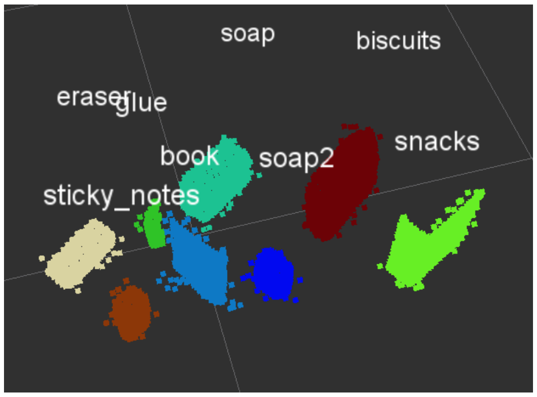


Figure 10 PC of classified objects for world 3

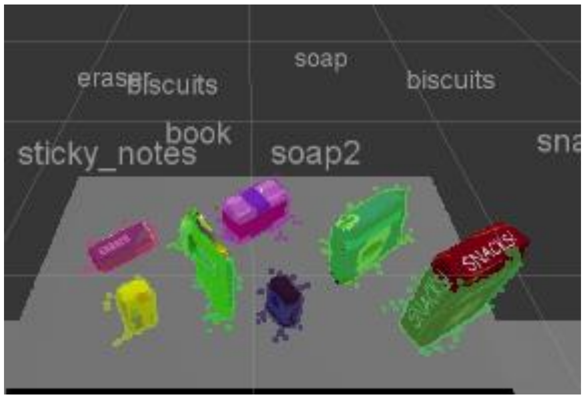
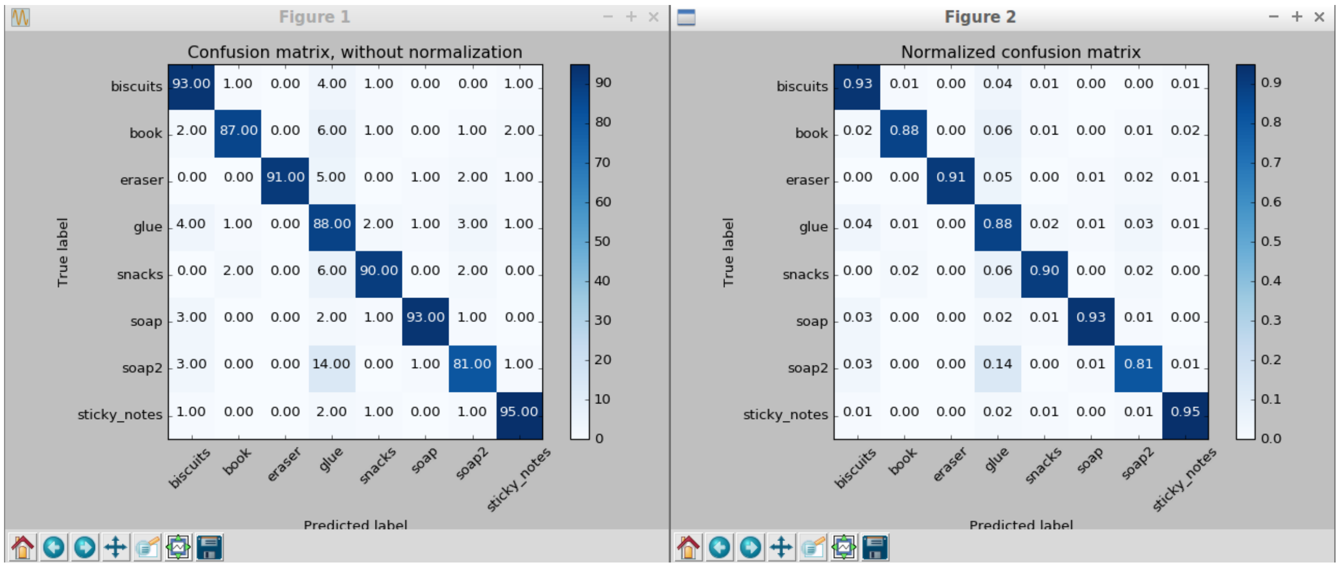


Figure 11 world 3 from camera perspective

SVM training

For section 3 above to work, the training algorithm needs to generate a model for each object so it can be identified/ classified later. The training algorithm loops through all the items summoning each 100 times and extracting the normal vectors features and the color features. These features are then fit into one histogram. That histogram can be thought of as the fingerprint of an item. And theoretically speaking, they should be different for objects with different colors and shapes.

After generating the features, we run a training algorithm. That algorithm outputs a confusion matrix.



Confusion matrix, without normalization

True label \ Predicted label	biscuits	book	eraser	glue	snacks	soap	soap2	sticky_notes
biscuits	93.00	1.00	0.00	4.00	1.00	0.00	0.00	1.00
book	2.00	87.00	0.00	6.00	1.00	0.00	1.00	2.00
eraser	0.00	0.00	91.00	5.00	0.00	1.00	2.00	1.00
glue	4.00	1.00	0.00	88.00	2.00	1.00	3.00	1.00
snacks	0.00	2.00	0.00	6.00	90.00	0.00	2.00	0.00
soap	3.00	0.00	0.00	2.00	1.00	93.00	1.00	0.00
soap2	3.00	0.00	0.00	14.00	0.00	1.00	81.00	1.00
sticky_notes	1.00	0.00	0.00	2.00	1.00	0.00	1.00	95.00

Normalized confusion matrix

True label \ Predicted label	biscuits	book	eraser	glue	snacks	soap	soap2	sticky_notes
biscuits	0.93	0.01	0.00	0.04	0.01	0.00	0.00	0.01
book	0.02	0.88	0.00	0.06	0.01	0.00	0.01	0.02
eraser	0.00	0.00	0.91	0.05	0.00	0.01	0.02	0.01
glue	0.04	0.01	0.00	0.88	0.02	0.01	0.03	0.01
snacks	0.00	0.02	0.00	0.06	0.90	0.00	0.02	0.00
soap	0.03	0.00	0.00	0.02	0.01	0.93	0.01	0.00
soap2	0.03	0.00	0.00	0.14	0.00	0.01	0.81	0.01
sticky_notes	0.01	0.00	0.00	0.02	0.01	0.00	0.01	0.95

Figure 12 confusion matrix showing the probability of correct classifications and the probability of false positives