Lab session 2

This session is based on the Lab session 1. Further more, this is focusing on hands-on part of the PCL. Based on your experience/interest, you can choose to work in the command line or in VisualStudio (installed).

PCL demos for 3D data processing

This tutorial shows basic demos from the PCL from different sources, mainly from their documentation. Please try to follow the steps indicated bellow, and make sure to check the relevant code sections. To do so, please navigate to the *pcl_demo* folder, and inspect its content.

For building, you will have to create a *build* folder, than execute *cmake ..* and *make* inside the build folder. Here will be generated the executables.

I/O demos:

- write a pcd file containing randomly generated points. What is the signature of the point cloud saving function? Can you save it as ASCII/Binary too?

Run from the *build* directory the *./write_pcd* command.

- test your file using:

pcl_viewer test_pcd.pcd

- read in that generated pcd file

./read_pcd Is there a way to open *ply* or *obj* format too?

3D Normals

- compute point normals on a point cloud and use built-in visualizer

./compute_normals

Can you change the support for which the normals are computed? Please make sure that you compile the modified code, by issuing *make* in the *build* directory!

Filtering

- run one of 3 different filters on a point cloud

./filtering 0 (pass through filter)

./filtering 1 (downsample to a voxel grid)

./filtering 2 (perform statistical outlier removal)

- visualize the output side-by-side with the original

pcl_viewer -multiview 1 ../data/table_scene_lms400.pcd table_scene_lms400_filtered.pcd

press ‘r’ to zero the viewpoint, and ‘l’ to list the color handlers. How much change is in the filesize of the original/filtered clouds?

Keypoints:

- find SIFT keypoints in a point cloud and visualize

./keypoints ../data/robot1.pcd keypoints

Can you change the parameters of the algorithm in code? Try to recompile/run the modified code.

Features:

- Compute PFH features on SIFT keypoints for two point clouds and then compute their correspondences

./keypoints ../data/robot correspondences

Can you modify the correspondence filtering part?

Sample Consensus:

- generate some points that fit a planar model as well as a bunch of outliers

./sample_consensus

- generate points as before, but use sample consensus to find inliers to a planar model

./sample_consensus -f

Try to modify the threshold of the plane segmentation, and watch the result!

Segmentation:

- perform iterative plane segmentation on real point cloud data

./plane_segmentation

- Visualize the output side-by-side with the original

pcl_viewer -multiview 1 ../data/table_scene_lms400.pcd table_scene_lms400_first_plane.pcd table_scene_lms400_second_plane.pcd

- perform euclidean cluster extraction after removing the dominant planes in the scene

./euclidean_cluster_extraction

- Visualize the output with all clusters in the same viewport

pcl_viewer cloud_cluster_0.pcd cloud_cluster_1.pcd cloud_cluster_2.pcd cloud_cluster_3.pcd cloud_cluster_4.pcd

Registration:

- perform iterative closest point to align two point clouds

./icp ../data/robot1.pcd ../data/robot2.pcd

- visualize aligned and combined point cloud beside originals

pcl_viewer -multiview 1 ../data/robot1.pcd ../data/robot2.pcd icp_aligned.pcd

- attempt to fit several point cloud templates to the target point cloud, output the best match

./template_matching ../data/object_templates.txt ../data/person.pcd

- visualize the matched and aligned template against the target PC

pc_viewer ../data/person.pcd template_aligned.pcd

you may need to press ’1′ several times to get a good color scheme for the two point clouds to be visible

These demos are meant for demo purposes. Should you need advanced features, you can find a complete documentation of the available tool in the PCL’s website.

Extra credits to J. Delmerico for the demos.

↑