# I. Structure of the Repository

In this repository you will find a car localization simulation using the KITTI dataset. There are also examples from the Point Cloud Library tutorials that can be found here: `http://pointclouds.org/documentation/tutorials/`. The code for the localization simulation is all contain in the folder MAIN.

This is a brief explanation of what you will find in this repository. You do not need to go through all of it, that would be a wasted of time. Please refer to the PCL documentation to make this programs run. The folder structure is as follows :

- **CmakeFiles**: No need to look in here. Files needed for compilation.
- **Data**: This folder contains the data needed to run the code in the **src** folder.
  - **Calibration**: this folder contains transformation matrices needed. Much more information can be found on the KITTI page: `http://www.cvlibs.net/datasets/kitti/`. Please take a look at the âĂIJOdometryâĂİ and âĂIJRaw dataâĂİ tabs in the link above. You'll find plenty of information there, here you have two links to papers explaining how the KITTI dataset is organized:
    - **Odometry**: `http://www.cvlibs.net/publications/Geiger2012CVPR.pdf`
    - **Raw data**: `http://www.cvlibs.net/publications/Geiger2013IJRR.pdf`
  - **pcd-files**: this folder contain LIDAR data which is stored in .pcd format. We can find two folders inside this directory:
    - **KITTI**: the LIDAR data of the KITTI dataset. You will load this files one by one when running the code.
    - **Test**: some other pcd files used in other scripts outside MAIN.
  - **poses**: stores the ground truth poses of the car, which is a combination of GPS and IMU (inertial measure system). Each of the 10 files contained in this folder specifies a few thousands of poses. At this point we are just working with âĂIJ00.txtâĂİ. Again, more information about how the poses are stored can be found in the papers and the line documentation of the KITTI dataset.
- **build**: This folder contains the executable files specified by CmakeLists.txt which is located in the pcl-pr0j3ct directory.
- **src**: inside this folder is the source code we have written. Except for the MAIN folder, the rest of them are either examples from the tutorials in the point cloud library webpage or modifications of those. They are included here now, but they will be removed at some point.
  - **MAIN**: This folder contains the core algorithm.
    - **src**: Contains C++ files. The MAIN.cpp is the code for cylinder extraction from KITTI dataset. The other cpp files in this folder are functions that are called by MAIN.cpp. Feel free to get back to me in case IâĂŹve missed comments for certain functions.
      - **MAIN**: main script that calls the rest of the functions.
      - **cluster_extraction.cpp**: You can find the explanation here. `http://www.pointclouds.org/documentation/tutorials/cluster_extraction.php`

- **cylinder_segmentation.cpp**: `http://pointclouds.org/documentation/tutorials/cylinder_segmentation.php`
- **extracting_voxel_grid.cpp**: We use a leaf size of 0.75cm to down-sample the point cloud `http://pointclouds.org/documentation/tutorials/voxel_grid.php`
- **plane_from_cluster.cpp**: `http://www.pointclouds.org/documentation/tutorials/planar_segmentation.php`
- **read_matrices_pose.cpp**: code to read the pose of the car from the poses file.
- **visualize.cpp**: Code to visualize the point clouds. `http://pointclouds.org/documentation/tutorials/pcl_visualizer.php`

* **headers**: Contains the header files of the src folder.

## II. Explanation of the MAIN Script

The process is as follows:

1. Read the PCD files.
2. Remove far away points.
3. Down-sample using Voxel grid with a predefined leaf size.
4. Remove the ground points.
5. Divide the laser scan into clusters.
6. Remove clusters with low density of points. Sparse clusters can not form good features (cylinders).
7. Extract planes from the clusters and remove them. This makes it easier to find the cylinders.
8. Remove clusters with low number of points after removing the planes. This means that those clusters where planes.
9. Run the cylinder segmentation algorithm to extract cylinders out of each cluster remaining.
10. Save the extracted cylinders.