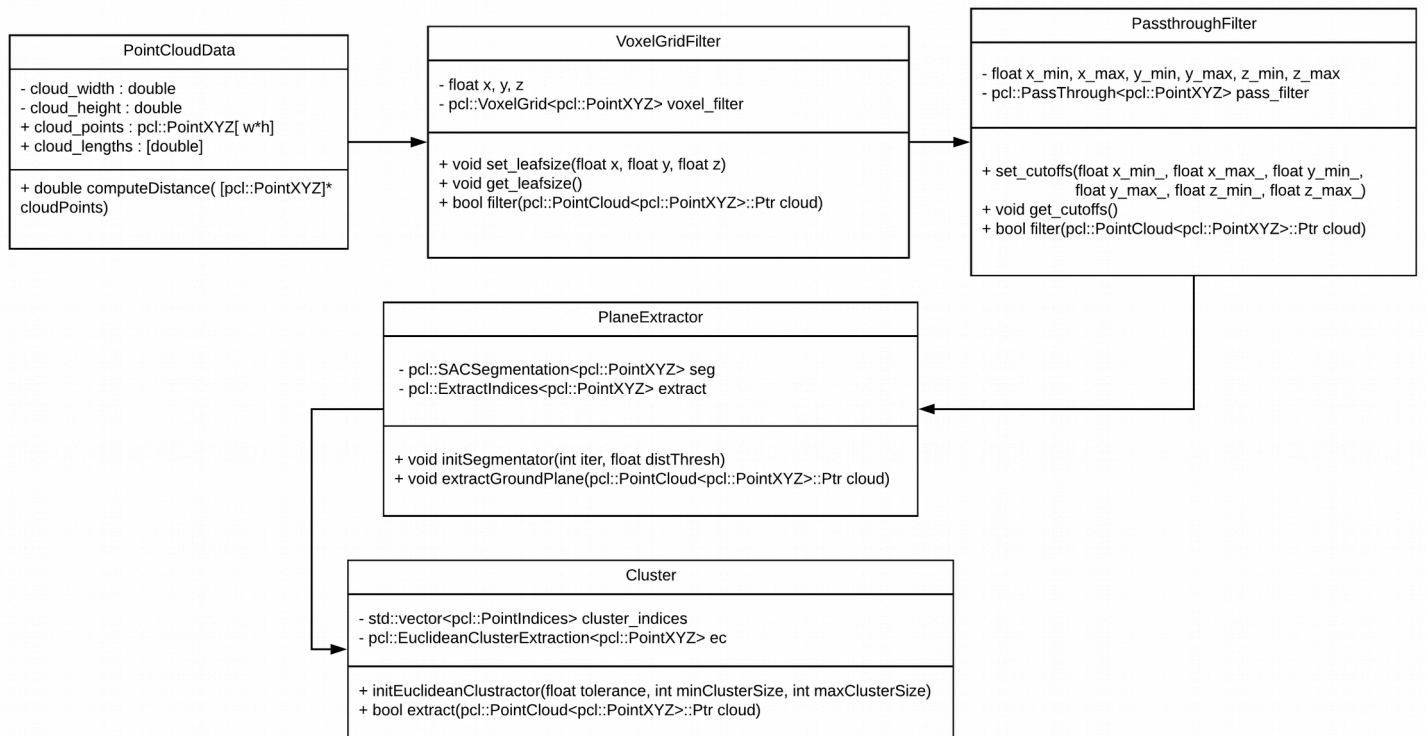# 3D Lidar Human Detection Module

Class Diagram

PointCloudData

- cloud_width : double
- cloud_height : double
+ cloud_points : pcl::PointXYZ[w*h]
+ cloud_lengths : [double]

+ double computeDistance([pcl::PointXYZ]* cloudPoints)

VoxelGridFilter

- float x, y, z
- pcl::VoxelGrid<pcl::PointXYZ> voxel_filter

+ void set_leafsize(float x, float y, float z)
+ void get_leafsize()
+ bool filter(pcl::PointCloud<pcl::PointXYZ>::Ptr cloud)

PassthroughFilter

- float x_min, x_max, y_min, y_max, z_min, z_max
- pcl::PassThrough<pcl::PointXYZ> pass_filter

+ set_cutoffs(float x_min_, float x_max_, float y_min_,
              float y_max_, float z_min_, float z_max_)
+ void get_cutoffs()
+ bool filter(pcl::PointCloud<pcl::PointXYZ>::Ptr cloud)

PlaneExtractor

- pcl::SACSegmentation<pcl::PointXYZ> seg
- pcl::ExtractIndices<pcl::PointXYZ> extract

+ void initSegmentator(int iter, float distThresh)
+ void extractGroundPlane(pcl::PointCloud<pcl::PointXYZ>::Ptr cloud)

Cluster

- std::vector<pcl::PointIndices> cluster_indices
- pcl::EuclideanClusterExtraction<pcl::PointXYZ> ec

+ initEuclideanClustractor(float tolerance, int minClusterSize, int maxClusterSize)
+ bool extract(pcl::PointCloud<pcl::PointXYZ>::Ptr cloud)

Activity Diagram

Recieve PCD(Point Cloud Data)

Apply Voxel Filter

Apply Passthrough Filter

Extract Ground Plane

Cluster regions

Extract Centroid of Cluster

Add Centroid to Human_Location array

Any remaining clusters

Process next cluster

Yes

No

Return Human_Locatiion array