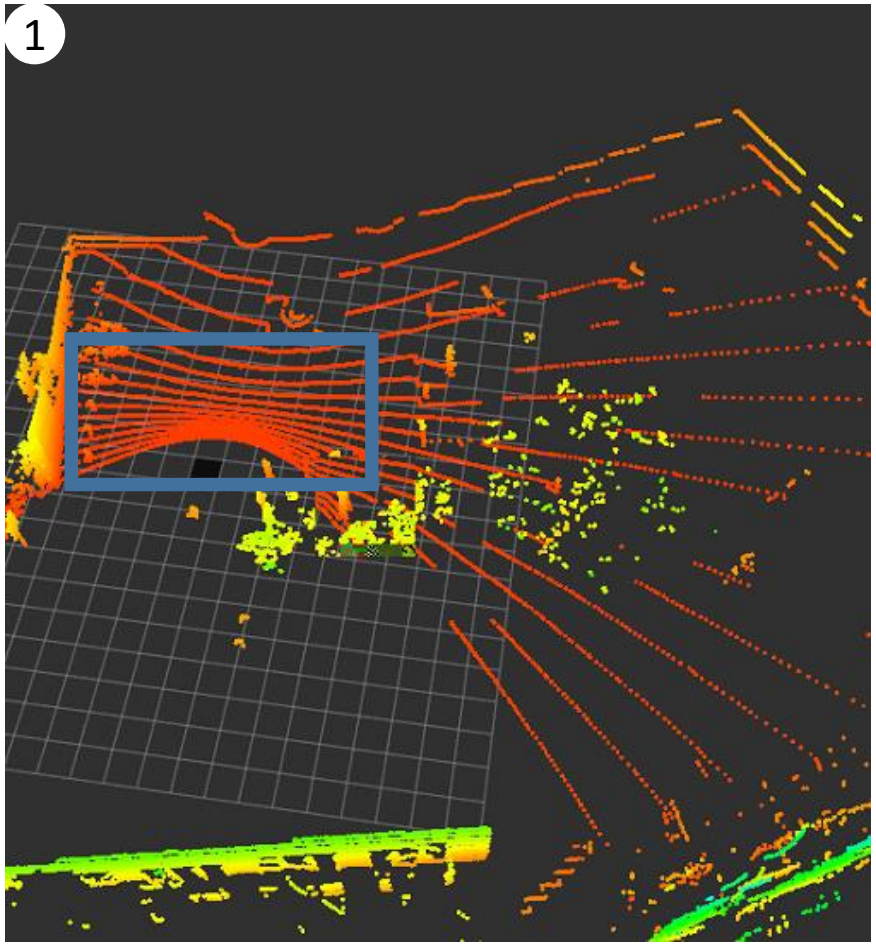
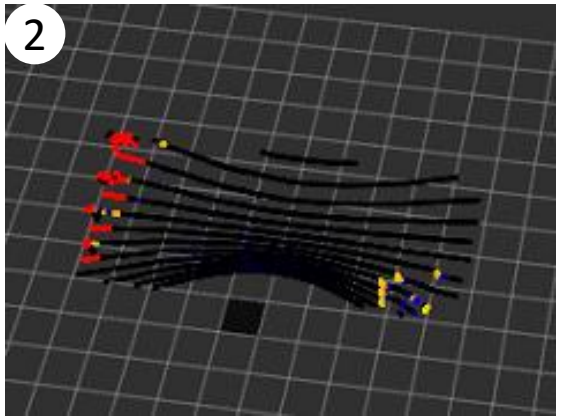
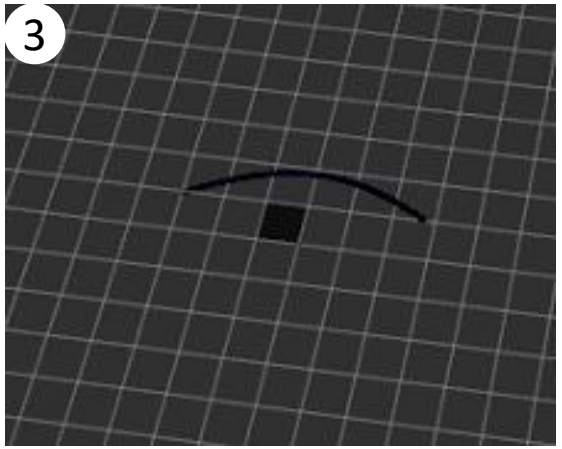






Velodyne VLP-16 has 16 laser beams that rotate 360 degree. (Pic 1)

When the distance gets far, the laser points become sparse. In the feature calculation, I only use the points 5 meters in front and -/+5 meters left/right to the robot. (Pic 2)

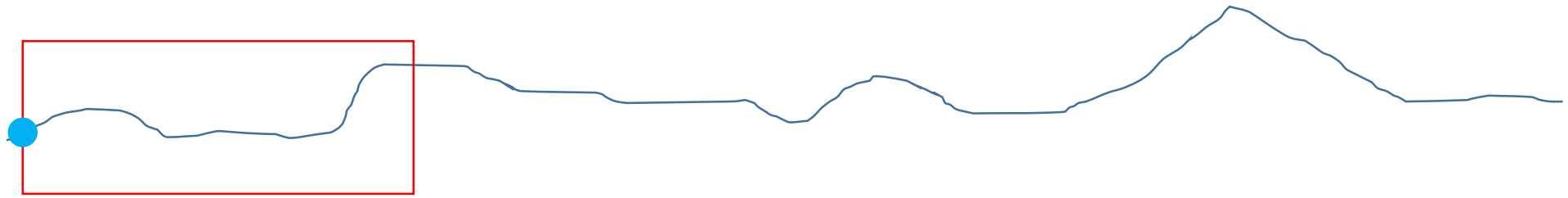
To minimize the height error between points, I calculate the terrain feature on each beam separately. (Pic 3)

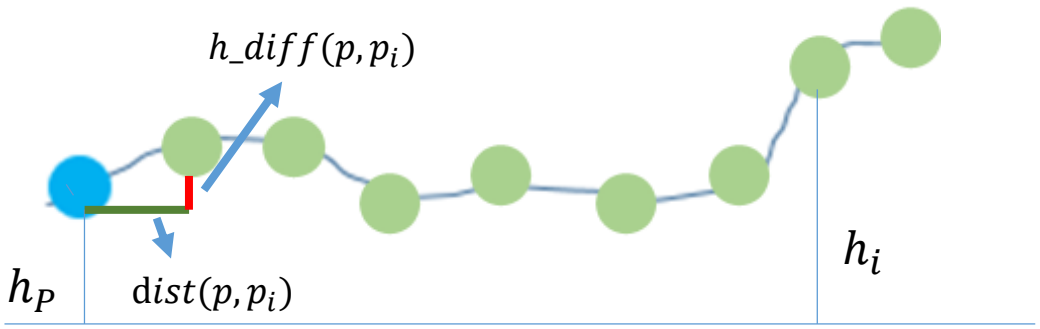
（When the new head with rotating laser is ready, we will get a much denser and accurate laser point cloud, then I will change the single laser beam process to point cloud process）

Terrain feature for every laser beam

All feature calculation is done inside the function “compute_terrain_feature” in the file “continuity_filter.h”

A window is defined for every point P in the laser beam, the window is defined as “cell_size” in the launch file. The unit is meter.





h_i: height of the ith point in the window

h_P: height of the point P

$h\_diff(p,p_i)$: height difference of the point P and p_i

$dist(p,p_i)$: horizontal distance of the point P and p_i

$$P_{mean\_height} = \frac{1}{N}\sum h_i$$

$$P_{variance\_height} = \left(\frac{1}{N}\sum\left(h_i - P_{mean\_height}\right)^2\right)^{\frac{1}{2}}$$

$$P_{\max\_height\_difference} = max\sum(h_i - h_p)$$

$$slope_i = \frac{h\_diff(P,p_i)}{dist(P,p_i)}$$

$$P_{mean\_slope} = \frac{1}{N}\sum slope_i$$

$$roughness_i = 9.8 * \frac{{slope_i}^2}{1 + {slope_i}^2}$$

$$P_{roughness} = \left(\frac{1}{N}\sum {roughness_i}^2\right)^{\frac{1}{2}}$$

Interface:

The entrance is the function “callback_velodyne” in “pointshape_based_processor.cpp”
Two filters are applied on the point cloud called: filter_crosssection and filter_continuity

filter_crosssection identify big obstacles like wall, human, tree… The obstacle points will be marked as red (there are still some bug here)
To see the result of filter_crosssection only, you can commend out the two lines of filter_continuity .

filter_continuity calculate all the terrain features of the points which are now obstacles (non-red points).
You can easily check the result of a single feature by changing “continuity_prob” value type and the color coding method in function “color_one_set”. I am using the roughness as the final feature.
The roughness value is separated into three part: from 0 to 0.4, colored in blue represent the flat ground; 0.4 to 3.5, colored in orange, represent the steps; greater than 3.5 is the obstacle.

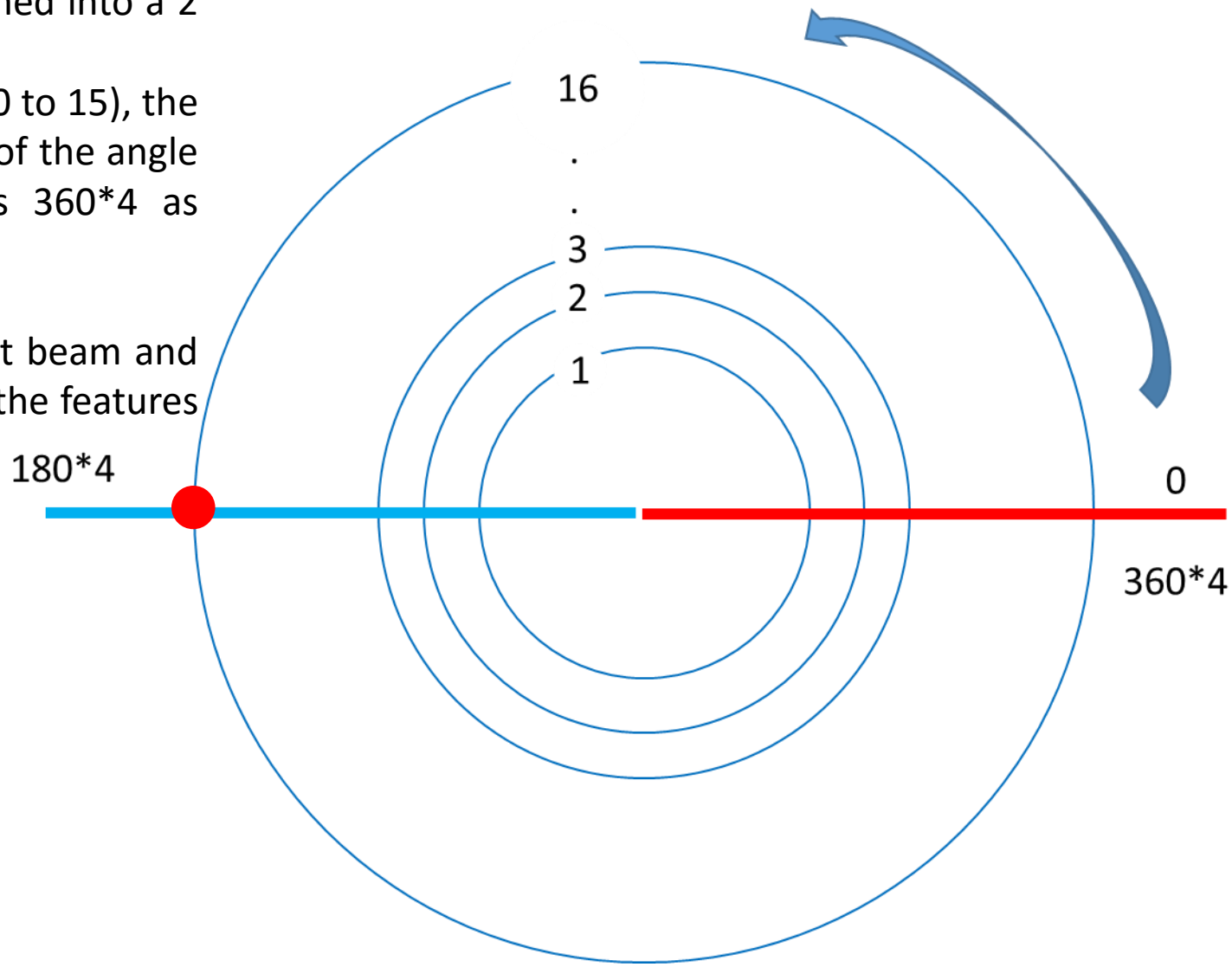
The output topic is “**gournd_obstacle**”

You can calculate your cost value and assign it to “continuity_prob” to quickly check the result without changing other parts.

Feature structure:

The feature structure is defined in “terrain_function_structure.h”

For every frame the points and its features are reformed into a 2 dimension array. (velodyne_sets and feature_sets)
The first dimension is the id for the laser beam (from 0 to 15), the second dimension is index for angles. (the resolution of the angle is defined in parameter “point_num_h” which is 360*4 as default)
For example:
Velodyne_sets[1][720] is the point located on the first beam and 720/4 = 180 degree. Feature_sets[1][720] represents the features for the point.



The “filtering_all_sets” function will update the feature_sets and “color_all_sets” function will color the points based on the features and reform the points to a single point cloud.

You can also access the feature you this way after the two filter.