CS-7644 Machine Learning for Robotics

Homework 4 – Hough and RANSAC

Lucas Schwing – Chloe Potherat

1 Hough Transform

We want to use Hough Transform to find likely plane parameters from the point cloud and in particular the ground plane. The Hough Transform relies on an ND accumulator, which is implemented as a 3D OpenCV matrix.

Accumulator

The accumulator is setup with 3 dimensions because we need 3 random points to make our plan. For each dimension, here is the range and discretization we choose:

```
<launch>
  <node pkg="floor_plane_hough" type="floor_plane_hough" name="floor_plane_hough" output="screen">
      launch-prefix="xterm -e gdb --args"
      <param name="base_frame" value="bubbleRob" />
      <param name="max_range" value="2.0" />
      <param name="n_a" value="30" />
      <param name="a_min" value="-1.0" />
      <param name="a_max" value="1.0" />
      <param name="n_b" value="30" />
      <param name="b_min" value="-2.0" />
      <param name="b_max" value="2.0" />
      <param name="n_c" value="33" />
      <param name="c_min" value="-2.0" />
      <param name="c_max" value="2.0" />

      <remap from="/floor_plane_hough/scans" to="/depth_registered/points"/>
  </node>
</launch>
```

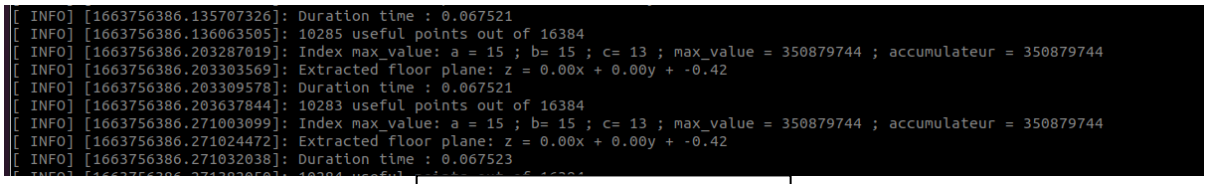
Launch file with the chosen parameters

Parameters; trade-off between precision and computation

To choose those parameters, we had to test multiple values based on the fact that:

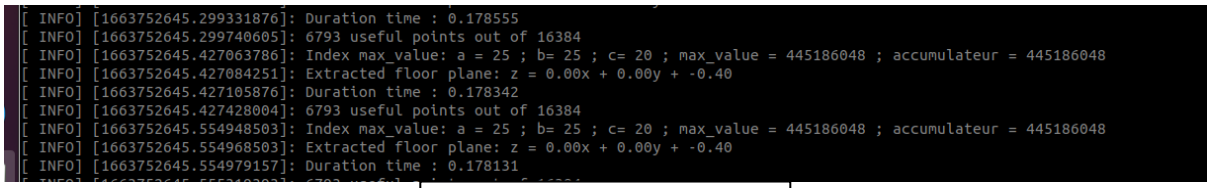
- If we increase the value of discretization without changing the value of the range, it increases the precision but also the computation time.

For n_a=n_b=30 and n_c=33:



computation time = 0.067s

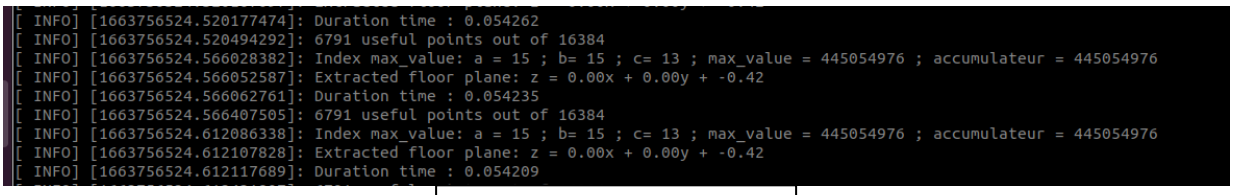
For n_a=n_b=n_c=50:



computation time = 0.17s

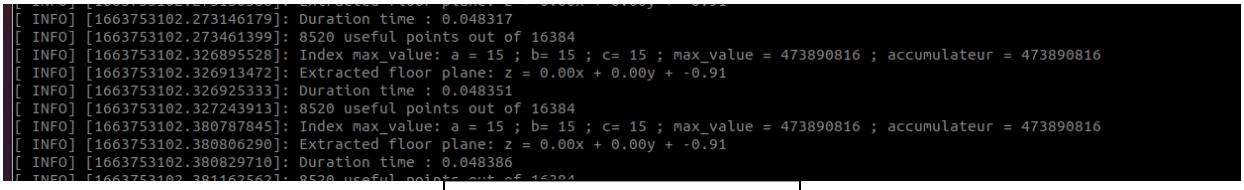
- If we broaden the range but we do not increase the discretization, we lose precision. But the computation time is better.

For a=[-1;1] ; b=[-2;2] ; c=[-2;2]:



computation time = 0.054s

Here a=b=c=[-10;10]:



computation time = 0.048s

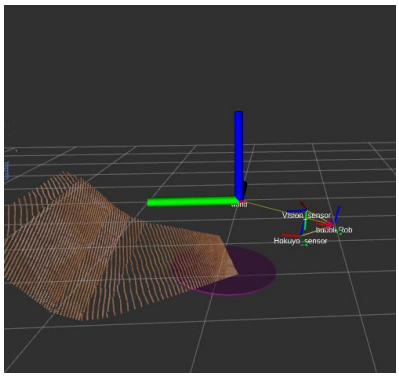
- We do not need the same discretization on all parameters.

Problem of aliasing?

Discretization implies aliasing. To avoid aliasing, we could limit the parameter plane to a certain band (with a filter for example) and then we could sample with Nyquist-Shannon. This method combined with the Hough Transform could enable us to have a precise estimator while still being resilient to noise and outliers.

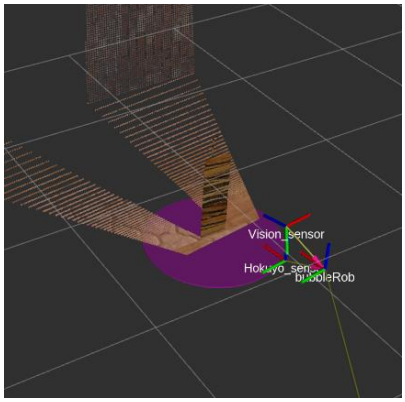
Sensitivity to environment

Hough Transform behaves pretty good next to slope:



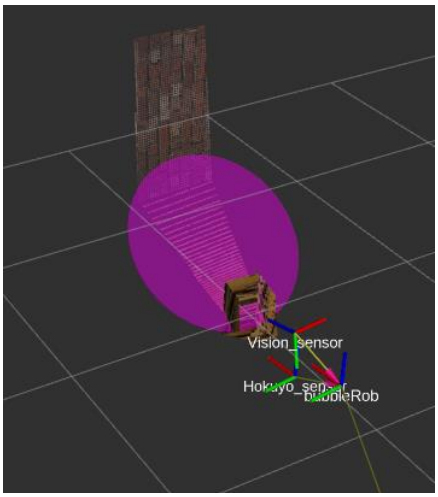
Ground plane on a slope

Also, it is doing well next to obstacles. But obviously, if we are too close to the obstacle, his point cloud is focused on the obstacle and the ground plane is not very great.



Ground plane next to an obstacle

.



Ground plane too close to an obstacle

Overall, we can assume that Hough is a robust detector because the sensitivity to noise and outliers is pretty low. However, Hough works for a low number of parameters. We had a lot of trouble with memory that prevented Hough to work...

2 RANSAC

Number of samples

If we use this equation to find the number of iterations: $h \geq \left\lceil \frac{\log \epsilon}{\log(1-P)} \right\rceil$ with e = the desired probability of not having sampled at least one good subset after h iterations, we want e to be 0 so log(e) will tend to infinite. Therefore, he needs to be the biggest that it can be. However, the computational time will be huge.

Thus, we tested a big value for the number of samples and lowered it until we have a correct computation time.

n_samples=1500

Tolerance

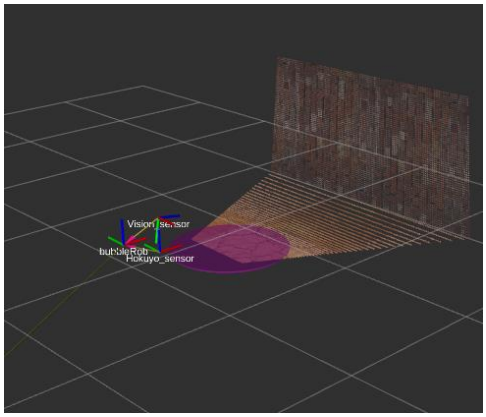
tolerance=1

```
fpr.launch
<launch>

  <node pkg="floor_plane_ransac" type="floor_plane_ransac" name="floor_plane_ransac" output="screen">
      launch-prefix="xterm -e gdb --args"
      <param name="base_frame" value="bubbleRob" />
      <param name="max_range" value="2.0" />
      <param name="n_samples" value="1500" />
      <param name="tolerance" value="1" />

      <remap from="/floor_plane_ransac/scans" to="/depth_registered/points"/>
  </node>
</launch>
```

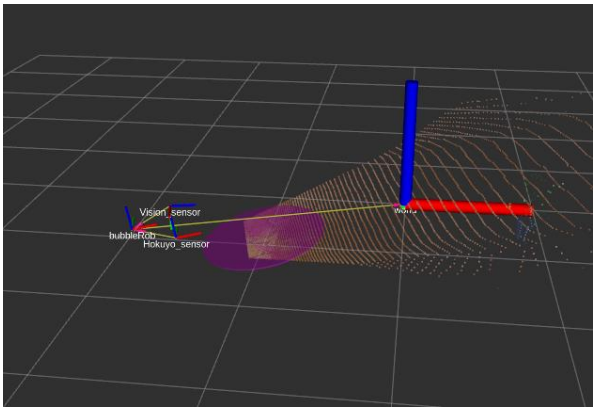
Launch file with the chosen parameters

Sensitivity to environment

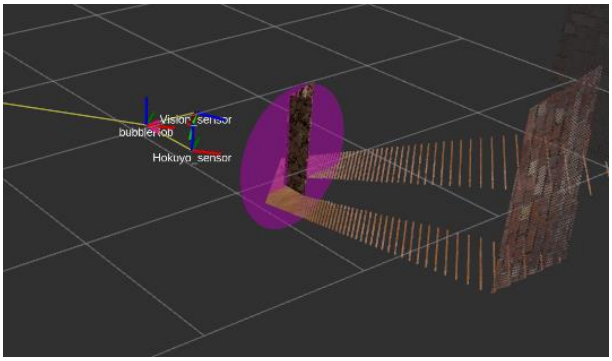
RANSAC reacts well to the environment but not when we are too close to an obstacle. The computational time stay good for 1500 samples but become a little bit too long when we choose 2000+ samples.



RANSAC without obstacles



RANSAC on a slope



RANSAC in front of an obstacle

Overall, RANSAC is also a great estimator.

RANSAC and least-squares method

We could use a method where RANSAC would be applied as a first step for deleting the outliers. And then we use the least-square method to fit to ground plane.