Proof of Concept

Map environmental structures with LIDAR

Thomas Schenk

Contents

[Context 3](#_Toc131109547)

[Problem 3](#_Toc131109548)

[Goal 3](#_Toc131109549)

[Available Algorithms 4](#_Toc131109550)

[KISS-ICP 5](#_Toc131109551)

[Robot 5](#_Toc131109552)

[External machine 5](#_Toc131109553)

[Comparison 8](#_Toc131109554)

[Conclusion 9](#_Toc131109555)

### Context

The JACKAL is aware of its surroundings. A key part of this is being able to recognize and see certain obstacles. This requires detailed sensors to provide the JACKAL with qualitative data to make these recognitions. But not only the JACKAL needs qualitative data to understand its environment, it also needs this data to be useful enough to serve to the end user so that it can understand the environment as well. The end-user could make use of this data in the form of a map or a 3d visualization.

### Problem

Currently the data provided by the JACKAL is inferior to the data which could be provided if the hardware would be upgraded. The current data which is provided by the robot is not good enough. The 2D map, although understandable, would benefit massively from a new LIDAR which is a direct hardware upgrade. And the 3D mapping, currently done by a camera, is almost too detailed, showing way too much data of the visuals around it; creating a very fuzzy and illogical image.

### Goal

Create a proof of concept which shows that a 3D LIDAR is a viable method to upgrade not only the 2D but also the 3D map by providing more accurate and quantitative data to construct these maps. The end-user benefits massively by having a map based on more and more accurate data to gain environmental awareness, this proof of concept aims to provide an answer to the question whether it is possible to create this data with the use of a new 3D LIDAR.

## Available Algorithms

From research we determined a few algorithms that were worth pursuing for this application. The two most notable were:

* FLOAM

<https://github.com/wh200720041/floam>

* KISS-ICP

<https://github.com/PRBonn/kiss-icp>

*Floam is still in its testing phase, therefore there isn’t any concrete data yet.*

## KISS-ICP

The KISS-ICP algorithm is a very straightforward and easy to use one. Yet it needed some parameter tuning to achieve a sensible result. More on this later.

We run the KISS-ICP algorithm in the following configuration:

### Robot

On the robot we directly interface with the LIDAR to make a pointcloud node which publishes its pointcloud to the topic: /velodyne\_pointcloud.

### External machine

We link the external machine to the robot and run the algorithm. In future versions this could potentially be done all on the robot; but for debugging and testing a separate machine is desirable. Not only for speed, but mostly for visualization. We can check the map being created realtime. We can even see ourselves moving.

With the parameters unchanged, we don’t get a map. This sounds super weird because it is. The reason for this is that the algorithm determines for itself what the appropriate value for the drawn voxels. The drawn voxels are the graphical dots the user sees after a while of mapping. The default value for these is so incredibly small that they’re not visible!

We tuned this parameter to be ‘1’ and ran the program again. We got the following output:  
A screenshot of a map

Description automatically generated with medium confidence

Arguably, this map is near useless. We cannot make out anything.

Mainly due to the fuzz of the previous map, we decided to also limit the range. This range works a bit weird, so it needs some explanation. The max-range variable decides at which range the robot starts drawing its voxels. By making this a bit smaller, the robot wont go drawing more inaccurate voxels all the way down the hallway. This will decrease the fuzz in the image above.

We decreased the voxel size to 0.5 and the max range distance to 70. This produced the following map.

A screenshot of a computer

Description automatically generated with medium confidence

We really liked that this map had a lot less fuzz, but now we felt like the voxel size was too small again.

We decreased the max range distance to 50 and the voxel size to 0.7. This produced the following map.

A screenshot of a computer

Description automatically generated with medium confidence

We felt quite happy with this map. To limit ourselves to the size of the proof of concept we stopped pursuing this for now.

## Comparison

To compare this work to the last group, we went ahead and looked at their media files. These files contain videos and screenshots of their maps. A great source to look at how noisy the previous lidar was when trying to detect objects, is this video: <https://projects.fhict.nl/t6-rb/fall-22/autonomous-exploration-of-a-building/-/blob/main/doc/media/jackal_navigation_mapping.mp4>

We can also look at some of the pictures provided in: <https://projects.fhict.nl/t6-rb/fall-22/autonomous-exploration-of-a-building/-/blob/main/doc/research/robot_SLAM_package_comparison.docx>   
where the grey picture depicts a room scanned with their lidar and the currently used algorithm.

We can conclude from both that they contain less data and are noisier and more prone to error compared to our results (especially the first one shows this quite well).

## Conclusion

Concluding we can state that while running very short test runs, the LIDAR can already make out quite intricate shapes and structures in a very respectable amount of time. All simulations which were ran weren’t specifically aimed at giving the LIDAR time to think, it was “just” driving around. When we compare this to the work of the last group, we can clearly see that even while driving around, their map is not only lesser in quality; it also reaches less far.

A 3D LIDAR is a very viable tool to provide more and more accurate data to the algorithm(s) to construct a map. Even in its rudimentary form, the data clearly shows improvement over the actions of the last group.