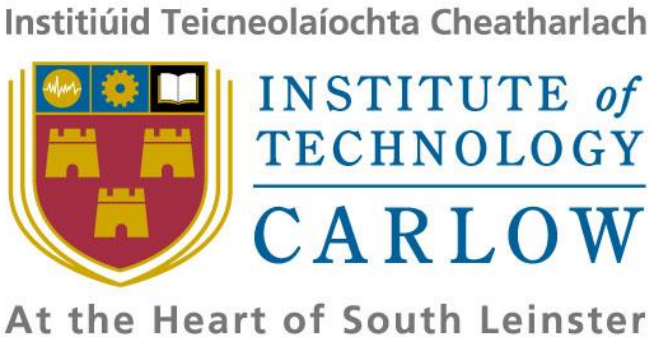


An exploration of Simultaneous Localisation and Mapping using Lidar

Paul Mahoney – C00227807

Research Document

November 2020

ABSTRACT

The aim of this research paper to decide on which technologies I will be utilising in my project. I have learned that the Robotic Operating System (ROS) will be a key aspect in my project. Ultimately, I will be utilising ROS in conjunction with SLAM, Raspberry PI OS, and a Picar V. A challenge I will be faced with is adapting to Debian coming from little Linux experience. Additionally, I have not had experience with creating a functioning system integrated with hardware.

Contents

1. Introduction

In this paper, I will be discussing many fundamental aspects of my envisaged project. I will detail any relevant technologies under broad headings such as hardware, software, etc below and discuss my chosen technologies and their potential alternatives if available.

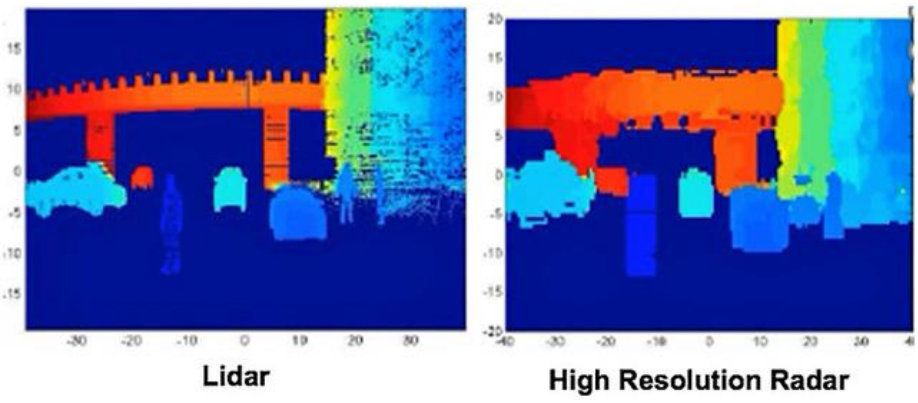
My project "autonomous mapping via lidar scanner" involves the creation of an unmanned vehicle that utilizes a lidar scanner as its sole method of traversal. The vehicle will create comprehensive 2D maps of its surrounding space for obstacle avoidance, navigation, and mapping. I have been supplied with Slamtech's RPLidar, Sunfounder's Picar-V, and a Raspberry Pi model 4B. These are the core hardware components of my project. I plan to use the Raspberry Pi and a control hub to allow the lidar scanner to interface with the Picar. I will create software that will read the data coming in from the lidar scanner and use it to navigate a space.

2. Overview of areas/technologies researched

In this section, I will outline my chosen technologies. I will give an in-depth analysis of these technologies and I will give a brief description of any potential alternatives I found during my research.

2.1 Lidar Scanner

The lidar scanner is the most pivotal element of my project. Major aspects of my envisioned system such as obstacle avoidance and autonomous 2D mapping rely on this component. Therefore, a strong understanding of how it works is essential to succeed. LiDAR stands for Light Detection and Ranging. Although it's not necessarily a new technology it has become more relevant in recent years. As opposed to similar technologies such as radar which emits radio waves, lidar utilizes infrared light. Infrared light provides a more accurate reading due to the nature of it being a beam, this allows pinpoint data to be returned vs a wave. An example to contrast how lidar can be more accurate can be seen below:



Lidar vs Radar (Neal, 2018)

In the case of my project, I have been given Slamtech's RPLiDAR A2M8. This specific lidar scanner returns 2D data by rotating on a 360° plane. This rotation is achieved through a in built motor. The A2 utilizes a laser triangulation system that allows for the high-speed collection of data. This device ranges 4000 times per second (Slamtech 2017). The data being read in is referred to as a point cloud (Rodríguez, 2019). A point cloud refers to a set of points in a plane.

RPLiDAR A2M8

2.2 Other Hardware

I plan to use a Raspberry Pi as the control hub of my project. A Raspberry Pi is a relatively low-cost microcomputer. It is capable of everything a regular desktop can do such as internet browsing, media consumption, and computation processing. It is widely used and there is an abundant amount of open source libraries and documentation to enable ease of development. Python is the most used language within the raspberry pi ecosystem, but it can compile any language.

An alternative to the raspberry pi is the Arduino. Broadly speaking the above description of a raspberry pi would fit the Arduino. Strictly speaking, an Arduino is a microcontroller where a Raspberry Pi is essentially a miniature personal computer. Additionally, there are fundamental differences between the two which caused me to choose the Raspberry Pi. The first of these differences is the architecture behind these devices. An Arduino comes with an 8bit microcontroller. In contrast, the Raspberry Pi has a 64bit controller which would be considerably faster. Similarly, the Arduino possesses two kilobytes of ram where a Raspberry Pi has one gigabyte. In terms of input/output (I/O) the Arduino is also less sophisticated with only one USB port for data transfer and a power connection. In comparison, a Raspberry Pi possesses' HDMI, an SD card port, 4 USB 2.0 ports, gigabit ethernet jack, and a wireless LAN port. Taking these factors into consideration, combined with the fact that the Arduino is best used for compiling C++ I felt as though the Raspberry Pi was by far the better choice for this project.

I have been provided with a Sun-founders Picar V. This is an unmanned vehicle that is designed to be used with a raspberry pi. I plan to use the data from the lidar scanner to influence this robot's navigation. Although the robot comes with a camera I only plan to use it to let the end-user manually control the vehicle. An autonomous mode will turn on the lidar and cause the robot to drive on its own. A user can then view maps being created dynamically and suggest routes through the software I intend to devlop.

2.3 SLAM

Accurate data is essential for drones to navigate safely. Moving an inch in the wrong direction could lead to falling off a step and damaging the device. A solution to this is Simultaneous Localization and Mapping (SLAM). SLAM is a subset of algorithms that take in data from

various sensors (in this case the lidar scanner) to create a 2D or 3D map of a space. This map allows the robot to know where they are in a space making obstacle avoidance possible (Martin, 2019). The combination of technologies such a SLAM and GPS power the self-driving car revolution (Standard, 2020).

The front-end collection of data in SLAM can be broken into two types, Visual SLAM and LiDAR SLAM. Visual SLAM collects data from cameras to triangulate its 3D position. This data allows the system to understand its current pose (estimation of current location relative to a landmark). As previously mentioned, LiDAR slam takes in a point cloud and converts the data into a map. An example of a completed 2D map can be seen below:

Completed lidar scan (Sadowski, 2020)

Various algorithms convert this point cloud into a map. From my research, I have found two main SLAM packages: Gmapping and Hector SLAM. Both of these SLAM packages achieve the same purpose. Gmapping is a C++ based package whereas Hector SLAM is a python-based package. From my preliminary research, it seems as though Gmapping operates with higher precision. As of the time of writing, I have not received my Raspberry Pi and I am having issues with connecting the Lidar so this will be tested further when I have the raspberry pi.

2.4 Software

I plan to keep the default operating system (Raspberry Pi OS otherwise known as Raspbian) which is a 32-bit Debian-based operating system optimized for the pi. There is a plethora of alternative OS' such as Ubuntu Mate, but I have decided to stick with the default system due to its support and compatibility with the device. Additionally, Raspberry OS was designed to be user-friendly for those who are new to the Raspberry Pi. With these factors in mind, I feel as though Raspberry OS best suits the current use case.

As for the implementation of my chosen algorithm (Not yet decided), it will be necessary to use the Robotic Operating System or ROS. ROS is a collection of frameworks and packages designed for the creation of robotic software development (ROS, 2020). ROS is a 'middleware' which is "software that provides common services and capabilities to applications outside of what's offered by the operating system" (Red hat, 2020).

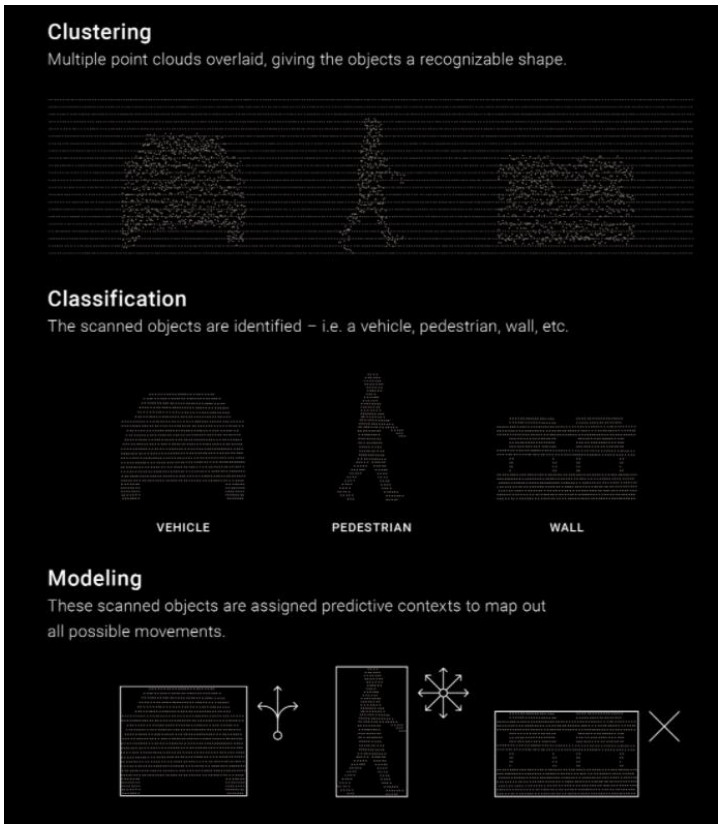
One of the most appealing features of ROS is the method in which it handles libraries. This allows high-level software to be designed without an in-depth knowledge of how the hardware works. If the hardware is compatible with ROS, a comprehensive library likely exists for it. This makes development in this system modular. The creation of complex objects from simpler ones is the definition of composition in object-oriented software development and I am excited to deploy this principle in my project.

Development in ROS is accomplished using Catkin workspaces. Catkin is ROS's build system or compiler. Typically, the building of code is hidden behind an integrated development environment (IDE). The building of code can be complex so having it withheld from the user is convenient. The problem is it is very useful knowledge that is typically 'swept under the rug' by software developers. Seeing as ROS is not an IDE but an OS(middleware), a build system is necessary to compile code. Catkin is ROS's solution to this. This is a file directory where packages can be edited and built. A workspace is created through a terminal. Each time you want to run Catkin in a terminal however it will have to be sourced to ROS and created using the catkin_make command in the right directory.

Doxygen is a tool I will be using to write software reference documentation. Doxygen generates an online HTML and offline reference manual for linked code. It supports most object-oriented languages such as C++, Python, and C#. This tool allows for documentation to remain consistent among multiple sources (Doxygen, 2020).

2.5 Existing Projects

As I touched on earlier a major application of lidar scanners is self-driving cars. In essence, this technology works by taking in a clustered 3D point cloud and matching the data to pre-existing shapes, such as a pedestrian, car, or wall. This is the classification of the point cloud. If a match is made here then it tries to predict its movement based on preexisting patterns. For example, a car can only swerve left or right, brake, or keep going straight (FF Team, 2020).



Classification of artifacts in self-driving cars (FF Team, 2020).

Additionally, this technology is utilized by Google with it's google streetcar. I could not isolate the exact use of this data due to it not being public knowledge but from my research, I could summarise is being used in the creation of 3D models of a space.

Another common application of this technology is a Roomba. A Roomba is a robotic vacuum that autonomously navigates a room to clean the floor. A lidar scanner among other sensors power this cleaner. Some higher-end models create maps of the rooms to increase their coverage of the entire floor. Although the design may differ from model to model the Lidar scanner is typically integrated into the front bumper. This is used to measure the distance of objects directly in front of them. This allows it to edge clean a room without constantly hitting the walls.

Eufy RobVac 30c

A multitude of existing beginner projects for ROS touch on this subject in a basic sense. Robots such as the Turtlebot aim to teach users about autonomous navigation. This project differs in its use of solely using the Lidar scanner as its sensor. Additionally, the Turtlebot's functionality in these tutorials is simplistic in relation to this project. Despite this, I aim to utilize these tutorials to help me gain a better understanding of ROS and similar technologies.

3. Summary and Conclusions

In conclusion, I have researched Lidar scanners, hardware, SLAM, and software. By researching various elements of my project, I have been able to discover what tools I would need to create an autonomous mapping system via the Lidar scanner.

Although I have been provided with specific hardware for my project, I have gained a greater understanding of each component and its alternatives. I now understand how the Lidar scanner works which will be pivotal knowledge moving forward into my project. I understand how it generates data in the form of a point cloud how it is converted into a map. I determined that a Raspberry Pi is better suited to this specific use case based on its greater specifications. Additionally, the specific robot I will be using is intended for use with the Raspberry Pi.

I have decided to keep the default OS for the Raspberry Pi, Raspberry Pi OS. I have chosen this over the alternatives due to its continued support. A potential challenge I face with this is having no experience with a Debian based Linux distribution. Fortunately, it was created to be friendly to new users.

I gained a greater understanding of how SLAM works, and I have narrowed down two SLAM packages to test, Gmapping and Hector SLAM. Both packages exist within ROS and will allow

for easy integration with my Raspberry Pi. Fortunately, both packages are similar making them interchangeable in the grand scheme of the project. I would like to test efficiency and precision as early as possible.

The SLAM packages exist within ROS. It is a 'middleware' that will act as a bridge between these packages and my machine. The system is open-source and was designed for robotic building and coding. Seeing as ROS is not an IDE but an OS a build system is necessary to compile code. Catkin is ROS's build system. It requires its file directory and to be sourced to ROS each time it needs to be compiled.

From my research, I believe I am ready to move onto the next stages. Next, I will be developing a functional specification, and following that, I will create a Design manual.

4. Bibliography

Neal, A., 2018. *Lidar Vs. RADAR*. [online] FierceElectronics. Available at: <https://www.fierceelectronics.com/components/lidar-vs-radar>.

Slamtech Cdn.sparkfun.com. 2017. [online] Available at: <https://cdn.sparkfun.com/assets/e/a/f/9/8/LD208_SLAMTEC_rplidar_datasheet_A2M8_v1.0_en.pdf>

Rodríguez,, A., 2019. *Point Cloud - An Overview | Sciencedirect Topics*. [online] Sciencedirect.com. Available at: <https://www.sciencedirect.com/topics/engineering/point-cloud>.

Martin, S., 2019. *What Is Simultaneous Localization And Mapping? What Is Simultaneous Localization And Mapping? | NVIDIA Blog*. [online] The Official NVIDIA Blog. Available at: <https://blogs.nvidia.com/blog/2019/07/25/what-is-simultaneous-localization-and-mapping-nvidia-jetson-isaac-sdk/> [Accessed 11 November 2020].

Standard, B., 2020. *From GPS To Lidar, The Technologies That Power Autonomous Vehicles*. [online] Business-standard.com. Available at: <https://www.business-standard.com/article/technology/from-gps-to-lidar-the-technologies-that-power-autonomous-vehicles-120071000034_1.html#:~:text=These%20vehicles%20use%20combinations%20of,on%20and%20along%20the%20roadway> [Accessed 11 November 2020].

Sadowski, M., 2020. *Hands On With Slam_Toolbox*. [online] msadowski blog. Available at: <https://msadowski.github.io/hands-on-with-slam_toolbox/> [Accessed 11 November 2020].

ROS, 2020. *Documentation - ROS Wiki*. [online] Wiki.ros.org. Available at: <http://wiki.ros.org/Documentation> [Accessed 11 November 2020].

Redhat, 2020. *What Is Middleware?*. [online] RedHat. Available at: <https://www.redhat.com/en/topics/middleware/what-is-middleware#:~:text=Middleware%20is%20software%20that%20provides,all%20commonly%20handled%20by%20middleware> [Accessed 11 November 2020].

Doxygen, 2020. *Doxygen: Main Page*. [online] Doxygen.nl. Available at: <https://www.doxygen.nl/index.html> [Accessed 11 November 2020].

FF Team, 2020. *What Is Lidar & How Is It Making Self-Driving Cars Safer? | Futuresight Blog*. [online] Ff.com. Available at: <https://www.ff.com/us/futuresight/what-is-lidar/> [Accessed 12 November 2020].