A novel integration of Open3D and two-dimensional LiDAR for educational robotics

Matthew Eaton*, Devin Grace*, Ricardo Ramirez*, Sky Papendorp*
*Department of Robotics & Mechatronics Engineering, Kennesaw State University, Atlanta, Georgia
Email: meaton12@students.kennesaw.edu, dgrace6@students.kennesaw.edu, rramir18@students.kennesaw.edu, spapendo@students.kennesaw.edu

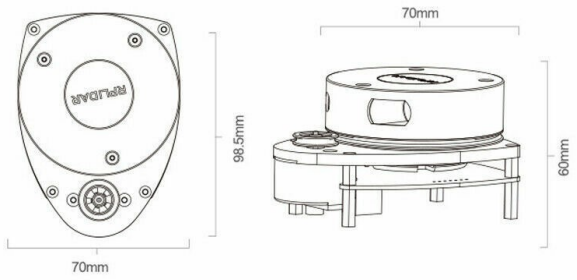

Fig. 1: RPLIDAR A1M8

***Abstract*—With the recent development of affordable LiDAR devices, the state of robotics education finds itself at a unique inflection point in which low-cost LiDAR devices, powerful single-board computers, and versatile open-source point cloud libraries have met, and present an exciting opportunity to integrate all three into the undergraduate engineering classroom. This paper aims to explore the various combinations of computer hardware available to educational institutions who wish to explore affordable LiDAR data labs, compare popular computing platforms and their performance in collecting and manipulating point cloud data for robotics education, explore methods for integrating affordable LiDAR hardware with increasingly popular point cloud manipulation libraries using the Python programming language, as well as provide educational institutions with an open source robotics platform for conducting LiDAR-based classroom lab activities.**

***Index Terms*—Robotics, LiDAR, Open3D, open-source, point cloud data**

I. Introduction

With the increased popularity, and perceived eventual ubiquity of technology such as self-driving cars, personal service robots, and other automated assistive devices, the need for accurate, low-cost, widely available LiDAR devices is readily apparent. Furthermore, the need for up-and-coming engineers to develop hands-on experience with these devices is crucial. Unfortunately, the historically high cost associated with industry-grade LiDARs means that purchasing such devices in quantities that would allow large groups of students the necessary experience is often out of reach for most undergraduate electrical, computer, robotics, or mechatronics engineering programs. Thankfully, several such low-cost devices have recently been developed, and their rapid increase in popularity has had a tremendous impact on the number of students exploring important engineering and robotics concepts such as the integration of LiDAR data with popular robotics programming languages and frameworks such as Python, C++, Matlab, as well as the ROS and ROS2 robotics programming frameworks. One such device is the RPLIDAR A1M8 developed by SLAMTec. [1] The A1, pictured in Figure 1, is a two-dimensional LiDAR with a class one eye-safe laser, and the ability to read values from 360 degrees at a rate of 8000 samples per second with a range of 12 meters. With a typical price tag of just under $100, it has quickly become the go-to LiDAR device for graduate and undergraduate robotics projects.

While the development of devices such as the RPLIDAR A1 is significant in its own right, its emergence onto the global robotics scene has also coincided with an exciting single-board computer renaissance. The global pandemic and resulting global parts shortage most certainly had a tremendously negative effect on engineering education. In fact, during the pandemic, the affordable and widely available "king" of single-board computers, the Raspberry Pi [2], was virtually unattainable. While the Raspberry Pi might have single-handedly inspired a physical computing revolution, the single-board computer industry was in trouble. Thankfully, since then the global parts market has slowly begun to return to normal, and while those periods of scarcity were unfortunate, there was an unexpected silver lining. During that period of time, the international engineering community witnessed a tremendous need to fill the gap in the single-board computer market, and thankfully several companies stepped up and developed impressive alternatives to the Raspberry Pi, including the NVIDIA Jetson Nano [3], the LattePanda Delta [4], and many more. Engineers and engineering students today have never before been presented with so many options for developing low-cost, Linux-based educational robotics platforms.

This resurgence of exciting single-board computers also referred to as SBCs, affordable LiDAR devices such as the RPLIDAR A1, as well as powerful open-source software, means that higher education institutions interested in preparing their students for careers in self-driving cars, robotics, and automation have a tremendous opportunity to offer their students hands-on experience with LiDAR technology in the

classroom. This paper aims to serve as a road map for institutions hoping to develop such hands-on experiences. With the development of low-cost LiDAR devices, a wide range of powerful SBCs, as well as the development of powerful open-source point cloud libraries, the combination of these three engineering tools presents undergraduate engineering programs with a unique opportunity to pursue low-cost, industry-relevant robotics projects on a scale previously unavailable to the traditional undergraduate engineering program.

II. Literature Review

The use of devices such as the very popular RPLIDAR A1 is well documented, especially within contexts such as its integration with the ROS and ROS2 framework, otherwise known as "The Robot Operating System", a popular robotics programming framework [5]. While the use of the RPLIDAR A1 to complete actions such as HECTOR SLAM [6], and other localization and mapping algorithms is an incredibly useful tool, there is also a seemingly so far unexplored benefit in combining the RPLIDAR's popular RPLIDAR_SDK [7] with the increasingly popular Open3D point cloud library [8]. To date, there has without a doubt been a tremendous amount of research conducted concerning the integration of the RPLIDAR A1 with the ROS, there is however a lack of research regarding the use of low-cost LiDAR devices such as the RPLIDAR A1 with other point cloud manipulation frameworks.

One of the most popular areas of research with regard to the RPLIDAR A1 is its use with the HECTOR SLAM algorithm. First developed by a team of researchers from Technische Universität Darmstadt; Stefan Kohlbrecher, Oskar von Stryk, Johannes Meyer, and Uwe Klingauf, HECTOR SLAM was one of the first SLAM algorithms developed specifically for the ROS framework [9]. While the origins of ROS itself can be traced back to 2007, it was in 2011 that the project saw a dramatic increase in popularity with the release of its first hardware product specifically designed for ROS, the Turtlebot. It was in that same year, 2011, that HECTOR SLAM was developed as a full-fledged ROS package that was then published open-source and made available to the ROS community [6]. With affordable robot hardware available in devices such as the Turtlebot, and useful localization and mapping software available through the implementation of HECTOR SLAM, the stage was set for the continued development of affordable LiDAR-based educational robotics.

The first implementation of the RPLIDAR A1 and ROS can be traced back to research conducted at Shanghai Jiao Tong University. In 2016 a research team working in the university's innovation lab known as RoboPeak Geek RD team published the first GitHub repository for the RPLIDAR ROS Package. [10] After a series of initial successes, the RoboPeak team then founded the organization SLAMTec which sells and manufactures the entire range of RPLIDAR devices to this day. With the groundwork laid, and an open-source ROS package for the RPLIDAR published, it was in 2017 that a GitHub user known as NickL77 developed the first integration between the RPLIDAR ROS package, and the HECTOR SLAM ROS package, the aptly named RPLidar_Hector_Slam ROS package [11].

While ROS remains an incredibly popular framework with which to read and evaluate point cloud data generated from the RPLIDAR A1, some researchers have explored the use of Matlab, another incredibly popular engineering-based computing suite, to visualize and interpret the data generated by the RPLIDAR A1. One such group of researchers from the Universidad de Las Américas in Quito, Ecuador published a paper entitled A MATLAB-Based Tool for 3D Reconstruction Technologies for Indoor and Outdoor Environments [12]. The team, comprised of researchers Kevin Jaramillo, David Ponce, David Pozo, and Luis Morales, found that the 3D plotting capabilities of Matlab were well suited to representing live point cloud data generated by the RPLIDAR A1. The research team went as far as creating an interactive GUI for obtaining the data from the RPLIDAR A1, as well as three popular stereo depth cameras. The data was then segmented, combined, and filtered to allow the user to reconstruct a three-dimensional point cloud based on the input data.

III. Proposed Work and Implementation

Fig. 2: Open3D Visualization Window

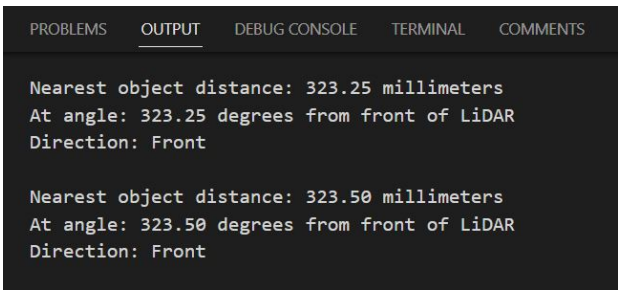

Fig. 3: Angle and Distance of the Nearest Point

While significant research has been conducted in regards to the RPLIDAR A1 and ROS, as well as MATLAB, there remains a significant opportunity in the development of software to integrate the RPLIDAR with newer open-source software frameworks designed specifically for gathering and manipulating point cloud data. Traditionally, the PCL library (or the Point Cloud Library) [13] has held the title of the most popular programming tool for manipulating point cloud data, however, in recent years an open-source alternative, Open3D, has seen a tremendous increase in popularity. Open3D is incredibly feature-rich, and given its unique mix of power and simplicity, it's easy to see why this Python package has seen such a rapid increase in its user base. There are many benefits to exposing students to software environments such as Open3D. Not only is it a modern and well-documented Python library, but its ability to represent and manipulate both 2D and 3D point clouds means that it is a tool that students can grow into, and continue to utilize as they are exposed to more advanced three-dimensional LiDAR systems. Additionally, by relying on Python alone, useful LiDAR data can be introduced into a classroom environment much sooner, potentially making introductory programming courses more active and engaging, as well as having the benefit of not needing to engage in the laborious process of installing ROS in order to access the data being produced by LiDAR devices. Not to mention the added benefit of being hardware agnostic, having the ability to run on a wide range of hardware and operating systems.

What this paper aims to show is that through the use of a simple Python script, a user is able to obtain a visual representation of a 2D point cloud from a live LiDAR device, as seen in Figure 2, as well as process the point cloud data in order to complete a whole host of useful actions such as identifying the nearest point in the point cloud to the origin, the angle of that point with respect to the origin, as well as measure the distance of the nearest point to the origin point of the LiDAR as seen in Figure 3. While this presents the user with a tremendous amount of useful data, it also barely scratches the surface of what is possible by utilizing the Open3D point cloud library. Not only that, but by utilizing other popular Python packages, such as pyserial [14], the LiDAR data can then be easily sent via serial to a microcontroller to complete actions such as motor control, or any other wide range of tasks useful within the field of robotics, or any other engineering discipline. During the testing portion of this research, the software written was tested on three different single-board computers, the LattePanda Delta 3, the NVIDIA Jetson Nano, and the Raspberry Pi 4B 8 GB, as well as a more traditional "Mini PC".

The software developed during the research portion of this project has three key components, the RPLIDAR SDK, numpy [15], and the Open3D library [16]. The first step in the process is to establish serial communication between the SBC and the LiDAR. The RPLIDAR A1 utilizes the CP2102 USB to UART converter, a very popular serial communication IC. The task of establishing a serial connection is handled by the RPLIDAR SDK. Once serial communication is established, the next task is establishing an Open3D visualization window. With the visualization window open, the scan data is then populated into a blank array by looping through the LiDAR data using the RPLidar SDK's `iter_scans()` method. This method generates an array of points in polar coordinates with respect to the origin point of the LiDAR. These polar coordinates are then converted to Cartesian coordinates using the `process_lidar_measurement()` function which utilizes numpy to complete the conversion. With the conversion complete, the array `lidar_data` is then fed into Open3D for visualization and displayed as a point cloud by utilizing the `o3d.utility.Vector3dVector()` method. The final step in the process is finding the closest point in the point cloud with respect to the origin point representing the LiDAR device's location, as well as the angle of that point with respect to the front of the LiDAR device using numpy's linear algebra function `np.linalg.norm()`, and degrees function `np.degrees()`. The code in its entirety can be found in the GitHub repository for this project.

https://github.com/ACBRrobotics/RPLIDAR_A1_open3d

In addition to developing example code for acquiring data, during the research portion of this project, an open-source robotics platform was developed to facilitate the further exploration of LiDAR data using an affordable and capable robotics platform. This robot chassis is meant to serve as a platform to explore the application of LiDAR data to more robotics-specific tasks such as navigation and obstacle avoidance. Measuring approximately 275 mm by 250 mm, this robot was designed to be small enough to support a classroom environment, but capable enough to be used outdoors. While this platform was developed to explore the concepts discussed in the scope of this paper, the hardware utilized in the chassis was chosen due to its ability to grow with the user, and can easily be adapted to more advanced robotics topics such as facilitating a ROS or ROS2 environment.

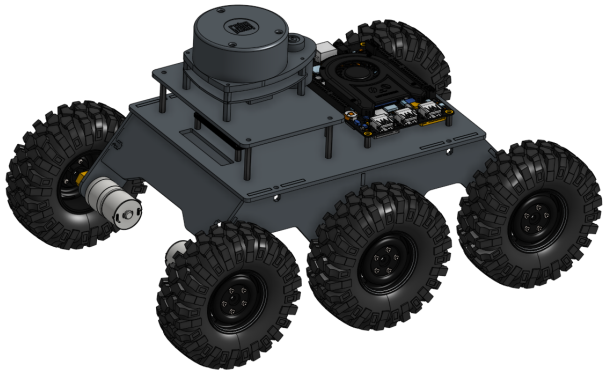

Fig. 4: open-source robot chassis for LiDAR data collection

The chassis was designed to be built from 3mm acrylic sheets and can be fabricated using any common laser cutter or CNC router. This can easily be adapted to accommodate more robust materials as well, such as aluminum or ABS.

While the chassis is pictured with 6 wheels, using only the 4 outermost wheels is more than capable of propelling the chassis. By manufacturing the chassis from individual laser-cut flat pieces of material, that can be bolted together without the need for additional brackets, the chassis was designed to achieve maximum strength, while maintaining simplicity and lower production costs. In the example pictured above, the LattePanda Delta 3 is pictured but the hole pattern can be easily modified to support any of the other SBCs tested in this paper. The motor driver chosen is the CYTRON MDD10A [17], which is a powerful 10 amp motor driver that can not only easily support the 6 dc motors in this design, but can also feasibly support larger dc motors as well. Much like the SBC pictured in this design, the hole pattern for the motor driver can also easily be altered to support a wide variety of popular dc motor drivers. This chassis is meant to be modified to meet a wide variety of needs. The fabrication files, BOM (Bill of Materials), and all software developed for the project can be found on the GitHub page for the project.

IV. Results and Analysis

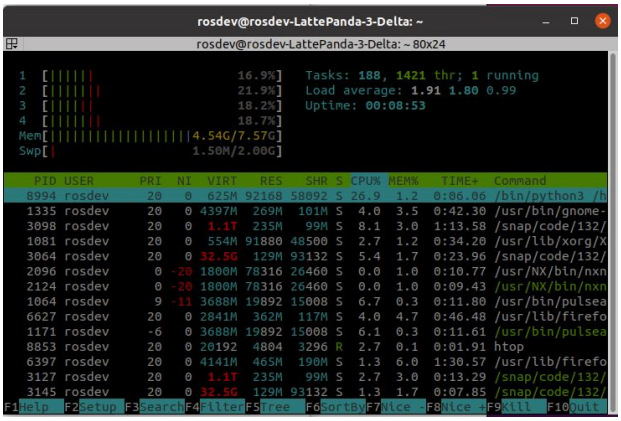

Fig. 5: resource utilization during LiDAR data collection on the LattePanda Delta 3

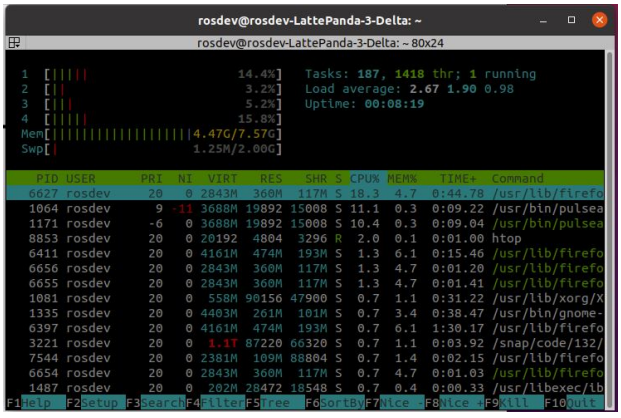

Fig. 6: resource utilization without LiDAR data collection on the LattePanda Delta 3

The first single-board computer tested was the LattePanda Delta 3. This SBC is powered by an x86-based 11th generation Intel N5105 Processor, with a base CPU frequency of 2.9GHz, and 8 GB of RAM. This is a particularly powerful SBC and with a typical retail price of $289, it is seen as more of a premium entry into the single-board computer market. The operating system installed for this test was Ubuntu 22.04.2 LTS. This particular operating system version was installed on each additional computing system tested. Ubuntu 22.04.2 LTS was chosen due to its long-term support as well as its status as the operating system of choice for ROS2 development. One of the most useful features of the LattePanda Delta 3 is its onboard ATMEGA32u4 microcontroller co-processor. The addition of this secondary microcontroller means that the communication between CPU-typical tasks, such as processing our Python script, and microcontroller-typical tasks, such as processing inputs and controlling outputs including light, sound, and motion through motor control, is seamless and incredibly simple to integrate. This allows for CPU-heavy perception tasks and lower-level functions to work together extremely well. Additionally, it can be powered by a 12v Battery, making it an ideal choice for a powerful long-range Linux-based robot platform. As seen in Figure 5, the LiDAR data acquisition performance was incredibly encouraging. Acquiring the LiDAR data, as well as performing the task of finding the distance and angle of the nearest point in the point cloud to the origin, accounted for a minuscule 70 MB of RAM, with a nominal impact of an 8.6% increase in the CPU utilization. While the LattePanda Delta 3 represents the most expensive SBC tested, it was also the highest-performing by far. This modest usage of computer resources means that other common robotics tasks, such as computer vision, additional sensor usage, or wireless communication, could easily be conducted simultaneously.

The second SBC tested was the Raspberry Pi 4 Model B. The Raspberry Pi 4 is powered by a Broadcom BCM2711, Quad-core Cortex-A72 (ARM v8) 64-bit processor with a base CPU frequency of 1.8GHz. This particular model tested was configured with 8 GB of RAM. With retail prices typically between $130 to $150, the Raspberry Pi 4B represents a good mix of affordability and performance. One of the main benefits of utilizing a Raspberry Pi for a project such as this in the classroom is its massive community. The immense amount of documentation surrounding the Pi 4 lends itself well to its integration into the engineering classroom. Additionally, having its standard 40-pin GPIO header readily accessible in software means that much like the LattePanda Delta 3, going from simply collecting the LiDAR data to conducting other tasks such as additional sensor readings or motor control, is relatively straightforward, and a feasible leap for undergraduate engineering students. That being said, the slightly slower ARM-based CPU does not allow for quite as much multi-tasking as some other more powerful x86-based SBCs. During the testing procedure, LiDAR data was collected and analyzed using the same Python script as all other SBCs tested. The LiDAR data was acquired from the RPLIDAR

A1, then processed and visualized using the Open3D point cloud library. The Raspberry Pi 4B's results were equally as encouraging as our other tests and provided us with incredibly promising results. While collecting LiDAR data, the Python script once again accounted for a modest amount of RAM, 106 MB, as well as an acceptable level of CPU utilization, with an increase of 35.6%. While the percentage of CPU utilization was approximately 4x higher than the LattePanda Delta 3, the data seems to suggest that despite its lower-power CPU, the Raspberry Pi 4B could also feasibly serve as a suitable platform for LiDAR data acquisition in an undergraduate engineering classroom setting.

The next SBC tested was the NVIDIA Jetson Nano. The Jetson Nano is powered by a Quad-core ARM Cortex-A57 Tegra X1 processor with a base CPU frequency of 1.6 GHz and is configured with 4 GB of RAM. With a typical retail price of $150, its cost is on par with the Raspberry Pi 4. During the testing process, the goal was to find a wide range of SBCs with discrepancies in RAM and CPU frequencies in order to provide the broadest amount of data possible. While the Jetson Nano is one of the newer SBCs in this group, NVIDIA has worked hard to create a robust robotics-centric community around the Jetson Nano. Much like the Raspberry Pi, it has a great deal of documentation and community support. The Jetson Nano hardware is relatively similar to that of the Raspberry Pi 4, with the exception of its GPU which is much more AI, ML, and computer vision focused. Despite having half of the RAM as the Raspberry Pi 4B tested, and a slightly slower CPU frequency, the Jetson Nano performed well during the LiDAR data acquisition and manipulation process. While running the Python script tested, there was an increase in RAM usage of 170 MB and an increase in CPU utilization of 25.6%. The data seems to suggest that the NVIDIA Jetson Nano could also feasibly serve as a suitable platform for leading undergraduate engineering students through a series of LiDAR data acquisition and manipulation labs

The final piece of hardware tested was the closest one might find to a standard PC. For institutions looking for a more traditional computing experience, while still keeping costs low, there are equally as many affordable small form factor PCs as there are SBCs available for engineering projects. The "Mini PC" selected for this experiment was chosen due to its low cost, as well as its low-power x86-based CPU. This particular model is powered by a 9th generation Quad-core Intel Celeron j3455 processor with a base CPU frequency of 1.5 GHz, which is configured with 8 GB of RAM and is typically priced close to $150, which is on par with the cost of the Jetson Nano and Raspberry Pi 4B, but less than the LattePanda Delta 3. With a width and length of only 130mm x 130mm, as well as a height of 50mm, while it may be quite a bit larger than the SBCs tested, it still maintains a relatively small footprint when compared to most modern PCs, and could easily be integrated into a mobile robotics platform. Despite the CPU's older architecture and a relatively low base frequency of just 1.5GHz, its performance was nonetheless acceptable. During the point cloud data acquisition and manipulation process, the CPU utilization percentage saw an increase of 40.9% and an increase in RAM usage of 159MB. This data suggests that almost any traditional "Mini PC" could feasibly be used to support a point cloud data lab such as what is discussed within the scope of this paper.

V. Conclusion

In conclusion, despite the wide range of processing power, and differences in CPU architecture, all computing systems tested performed very well and could serve as an excellent platform for introducing point cloud data to undergraduate robotics and mechatronics engineering education. This hardware can easily be integrated into a robot chassis to facilitate further application of collected LiDAR data. Additionally, the software utilized to visualize the RPLIDAR A1 data in Open3D was successful in serving as a lightweight interface between the platforms for acquiring, visualizing, and manipulating live point cloud data from a LiDAR device. Future work will include the expansion of applications for the collected LiDAR data to facilitate a series of labs that may be useful for a wide range of undergraduate engineering courses. Given the growing need to prepare up-and-coming engineering students for careers that will utilize a wide range of point cloud producing devices, institutions who wish to expose their engineering students to modern methods of point cloud data manipulation in a hands-on environment have a unique opportunity to do so by integrating affordable LiDAR systems such as the RPLIDAR A1 with modern and powerful point cloud libraries such as Open3D.

References

[1] "home - slamtec," SLAMTEC INC., Copywrite 2022-2023, accessed on June 28th, 2023. [Online]. Available: https://www.slamtec.ai/home

[2] "home - raspberrypi," The Raspberry Pi Foundation, Copywrite 2023, accessed on June 28th, 2023. [Online]. Available: https://www.raspberrypi.com/

[3] "home - nvidia developer portal," Nvidia Corporation, Copywrite 2023, accessed on July 7th, 2023. [Online]. Available: https://developer.nvidia.com/embedded/jetson-nano-developer-kit

[4] "home - lattepanda," LattePanda, Copywrite 2023, accessed on July 7th, 2023. [Online]. Available: https://www.lattepanda.com/lattepanda-3-delta

[5] "Ros wiki - documentation," Open Source Robotics Foundation, Copywrite 2023, accessed on July 7th, 2023. [Online]. Available: http://wiki.ros.org/Documentation

[6] tu_darmstadt, "hector_slam," GitHub, 2011. [Online]. Available: https://github.com/tu-darmstadt-ros-pkg/hector_slam

[7] Slamtec, "Slamtec/rplidar_sdk." GitHub, 2018. [Online]. Available: https://github.com/Slamtec/rplidar_sdk

[8] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3d: A modern library for 3d data processing," *arXiv preprint arXiv:1801.09847*, 2018.

[9] S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf, "A flexible and scalable slam system with full 3d motion estimation," in *Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. IEEE, November 2011.

[10] robopeak, "rplidar_ros," GitHub, 2016. [Online]. Available: https://github.com/robopeak/rplidar_ros

[11] NickL77, "Rplidar_hector_slam." GitHub, 2017. [Online]. Available: https://github.com/NickL77/RPLidar_Hector_SLAM

[12] K. Jaramillo, D. Ponce, D. Pozo, and L. Morales, "A matlab-based tool for 3d reconstruction technologies for indoor and outdoor environments," in *Advances in Emerging Trends and Technologies*, M. Botto-

Tobar, J. León-Acurio, A. Díaz Cadena, and P. Montiel Díaz, Eds. Cham: Springer International Publishing, 2020, pp. 280–290.
[13] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 1–4.
[14] pyserial, "pyserial/pyserial." GitHub, 2015. [Online]. Available: https://github.com/pyserial/pyserial
[15] numpy, "numpy/numpy." GitHub, 2011. [Online]. Available: https://github.com/numpy/numpy
[16] I. S. L. Org, "isl-org/open3d." GitHub, 2018. [Online]. Available: https://github.com/isl-org/Open3D
[17] "Cytron," CYTRON Corporation, Copywrite 2023, accessed on July 7th, 2023. [Online]. Available: https://www.cytron.io/p-10amp-5v-30v-dc-motor-driver-2-channels