Adaptive Speed Optimization for SLAM Using KISS-ICP on a Rover Robot

Andres Castrillon, David Seong, Dylan Leong

***Abstract*— This paper presents the implementation of a state-of-the-art Simultaneous Localization and Mapping (SLAM) model, KISS-ICP, on a wheeled robot developed in the Robomechanics Lab at Carnegie Mellon University [1]. In this paper, we introduce an adaptive speed system designed to optimize both mapping efficiency and accuracy in previously unexplored environments. The focus of this project is to assess mapping accuracy at various robot speeds in a test environment, where generated maps are compared against a ground truth map. Future efforts will leverage these results to develop an adaptive speed algorithm to enhance mapping accuracy without sacrificing accuracy through a developed adaptive speed algorithm. Additionally, plans include deploying the robot in new, unstructured environments to validate its performance.**

I. INTRODUCTION

Simultaneous Localization and Mapping (SLAM) is a fundamental tool in robotics that addresses critical tasks such as autonomous navigation, environmental monitoring, and exploration. Despite its potential and wide range of applications, challenges persist in improving its robustness, computational efficiency, and adaptability in dynamic and unstructured environments while maintaining accuracy.

This paper presents a novel approach to SLAM that evaluates the concept of adaptive speed control based on environmental complexity. Through the implementation of the state-of-the-art KISS-ICP (Keep it Simple and Straightforward Iterative Closest Point) algorithm on a wheeled robot platform, we aim to demonstrate that robotics tasks that leverage SLAM can be significantly optimized through the use of the adaptive speed method [2].

Our key contributions include:

1. Implementation of KISS-ICP on a wheeled robot platform.
2. Development of a basis for adaptive speed control by comparing point cloud density at 8 different speeds.
3. Formulate future research methodology for a robust adaptive speed control autonomy stack.

II. BACKGROUND

With recent advancements in SLAM, the efficiency and robustness of mapping and surveillance of previously unknown areas have improved significantly. Among these, the KISS-ICP algorithm, introduced in [2], represents a minimalistic LiDAR-based approach that leverages the iterative closest point (ICP) algorithm and an effective downsampling strategy to achieve accurate robot pose estimation with minimal parameter tuning [3].

While adaptive SLAM techniques have been explored in diverse contexts, direct application of adaptive speed control, leveraging point cloud density and mapping accuracy, is a relatively unexplored area. Cosgun introduced the concept of speed maps to set static and dynamic speed limits for mobile robots based on environmental context and human presence [8]. The speed maps allowed robots to adjust speed dynamically to ensure safe and optimized navigation through environments. Specifically, potential collisions were reduced by moving faster in open areas and slower in confined or human-populated areas on the map. Building on this foundation, researchers aim to refine adaptive speed control strategies by integrating more nuanced environmental data and dynamic real-time feedback, opening up further possibilities for improving efficiency and safety. To improve upon the existing speed maps, the adaptive speed control method initialized with training information about the environment. This approach has the potential to bring significant improvements in field robotics, enabling more effective and reliable autonomous navigation in dynamic and challenging environments.

III. METHODS

1. Project Methodology

Within the scope of the project, the methodology provided proof of concept for adaptive speed control. The motivation behind the proof of concept was to provide a metric to indicate the correlation between point cloud accuracy and robot speed. The proceeding sub-sections described the execution of the methodology.

1A. Proof of Concept Environment Setup

The testing environment was a large, enclosed space with obstacles including stacked foam rectangles and cardboard boxes, shown in Fig. 1. The large obstacles were utilized to display clear, static objects within the 3D reconstructed point cloud. The robot moved along a straight path with the starting and end points defined by green mats. This setup ensured a controlled environment where the effects of varying speeds and obstacle density on the LiDAR data could be effectively observed and analyzed.

Figure 1. Proof of concept environment setup.

1B. Varying Speed Experiments

Before experimentation, the hardware of the rover robot with an attached LiDAR was verified for any possible issues. The path of the robot was hard-coded to travel at a constant speed for a period of time in seconds. The robot was set to 8 different speeds for a total of 9 trials. The additional trial was included for evaluation of the robot at the highest speed. The LiDAR data was recorded into rosbags with the Ouster SDK source code to be processed by KISS-ICP. This ensured consistency across all trials and provided high-quality data for subsequent analysis and validation of the algorithm's performance.

1C. CloudCompare Software

After each trial, the map generated for the room using KISS-ICP was exported as a .ply file. Maps were then compared two at a time using CloudCompare [7]. In this software, the two point-clouds were imported and registered to ensure the two maps were aligned. Then, a point-to-point comparison was run to calculate the mean distance and standard deviation between the two clouds, which provides insight into how different the two maps are. A visual example of how two point-cloud maps are aligned on CloudCompare is shown in Fig. 2.

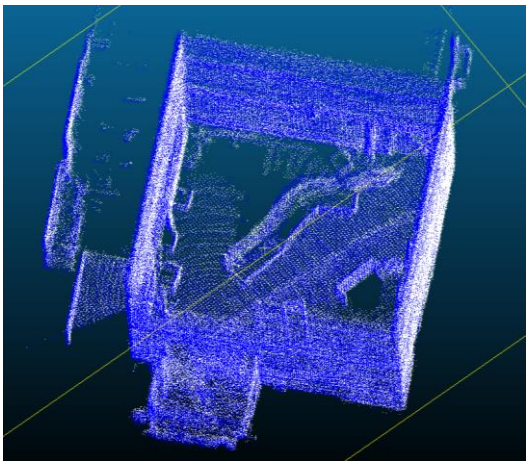

Figure 2. Point-to-point comparison between two maps using CloudCompare.

2. Research Methodology

The end goal of the research is to develop an autonomous stack for adaptive speed control. However, the project primarily focused on the proof of concept to verify the necessity of the method. The future methodology consists of a training phase, where speed parameters are tuned, and a testing phase, used to validate the performance of the model. These phases are further discussed in detail in the following sub-sections.

2A. Environment Setup

The testing environment will consist of an open space enclosing 5 obstacles (e.g. traffic cones). The mapping path for the robot will be designed to start at an origin mark, go around the obstacles, and then return to the origin. The robot will use GPS data to ensure it remains on track. Fig. 3 illustrates the intended testing environment and path of the robot. As can be seen, the obstacles are clumped together to create a region in the robot's path that is more densely packed than the rest.

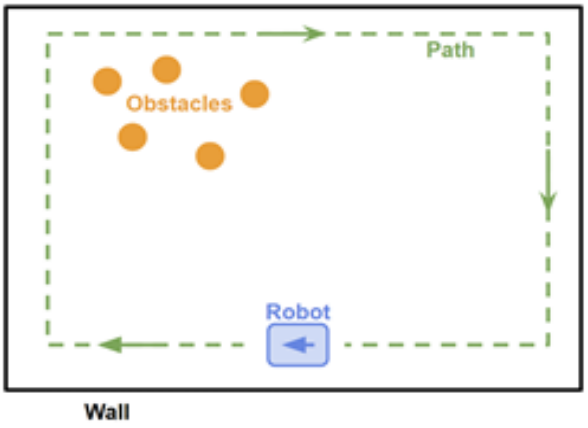

Figure 3. Environment setup and intended path of the robot.

2B. Initial Ground Truth Map

In the first run, the rover will follow its path at 50% of its maximum speed to produce a "ground truth" map of its environment. Given the precision of the Ouster sensor and the reliability of KISS-ICP, it is assumed the mapping results at this speed will be representative of the ground truth [2][4].

2C. Adaptive Speed Training

At this stage, the rover will complete the path again in the same environment a series of times, however, now, the speed at which the robot is moving will vary throughout the path based on the density of obstacles detected around the robot. When the amount of occupied space around the robot exceeds a set threshold, the robot's speed will be adjusted to a 'slow speed', to ensure detailed mapping of dense areas. Meanwhile, if the occupancy around the robot is below the threshold, the robot's speed will be set to a 'fast speed', to reduce mapping time.

At the end of each run, the map for the trial will be compared to the ground truth map of the environment. Based on this comparison, the 'slow speed' and 'fast speed' velocities will be tuned to improve the mapping accuracy or/and speed for the next trial. This tuning process will be repeated until the robot reaches the desired accuracy under the expected time. Then, the 'slow speed' and 'fast speed' values that lead to the best performance will be saved as the parameters for the trained model.

2D. Adaptive Speed Testing

For the testing phase, the obstacles in the environment are rearranged to new positions. Then, a new ground truth map is developed following the methodology described in Section 2.B.

Now, the trained model will be tested in the new environment by following the same trajectory as before, but this time, varying the speed throughout the path according to the trained 'slow speed' and 'fast speed'. An example of what the updated environment and the expected robot path could look like with the adaptive speed is shown in Fig. 4.

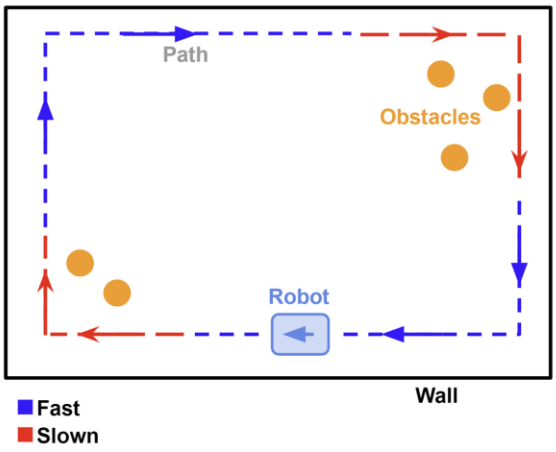

Figure 4. Updated environment used for validation.

The map generated from this test trial will be compared to the ground truth that was created for the updated environment. The overall performance of the model will depend on the accuracy of the generated map and time reduction. Once validation of the method has been completed in a controlled environment, the adaptive speed control pipeline will be experimented in outdoor environments.

2C. Summary of Research Methodology

Once again, the research methodology presented in Section 2 is the plan for implementing adaptive SLAM in future experiments based on the results derived from this paper. A high level overview of the training and testing steps proposed is shown in Fig. 5.

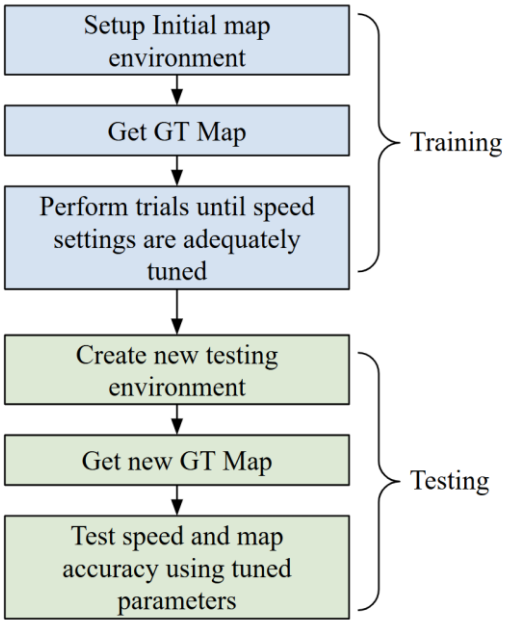

Figure 5. Flow chart overview of training and testing steps for the end goal research.

Currently, the plan is to approach the tuning process manually while the trials are performed. However, depending on the complexity of the tuning process or how many trials are needed to achieve adequate performance from the algorithm, it may be beneficial to automate the tuning process through simulation and optimization.

IV. GITHUB

The implementation code, including the KISS-ICP integration, adaptive speed control algorithm, and data processing scripts, is available at [5].

V. REVIEW OF THE COLLECTED DATA

Before integrating KISS-ICP into the complete pipeline, the validity of the method was confirmed with visualization of datasets. The Ouster OS0 collected data in the format of rosbags. Specifically, the data collected in the rosbags were IMU, LiDAR, and metadata packets. For experimentation, a

simple path from ANSYS Hall to the gazebo in Schenley Park and back were recorded into two separate datasets. Figure 6 displayed a single time step from the LiDAR data around the location of Scaife Hall. The datasets will be processed into our pipeline in future steps.

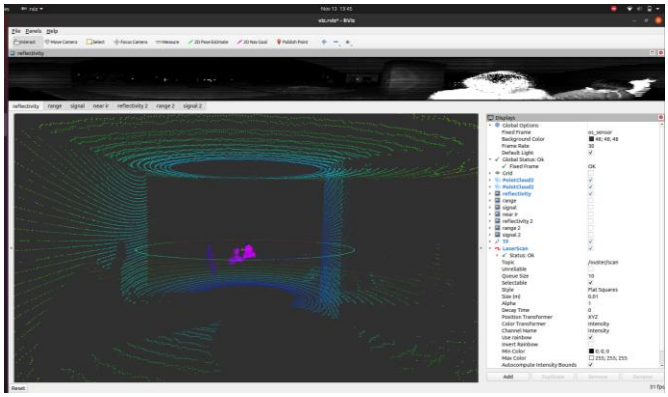

Figure 6. Screenshot of a single time step from LiDAR data.

VI. EXPLANATION OF EXPERIMENTS, RELEVANCE AND DISCUSSION OF ANY IMPORTANT IMPLEMENTATION DETAILS

As stated previously, validation of the KISS-ICP method was necessary before integration into the complete pipeline. The MulRan parking lot dataset was visualized with KISS-ICP and results were recorded [6]. The point cloud visualization of the MulRan dataset after a substantial number of timesteps was shown in Figure 7.

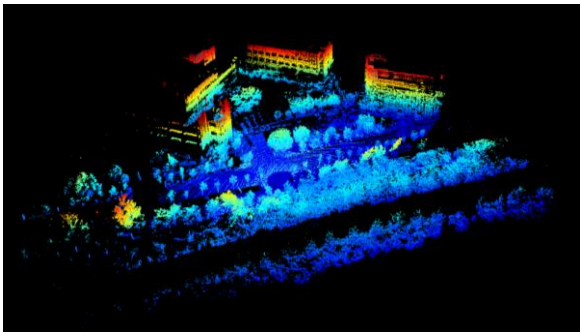

Figure 7. Screenshot of the MulRan dataset visualized with KISS-ICP

Next, KISS-ICP will be tested on collected rosbags. This proceeding step began the utilization of ROS1. The eventual goal is the integration of KISS-ICP into a ROS1 autonomy stack. Specifically, a rosbag will be processed with the ouster-ros package and published to the point2 topic. The KISS-ICP package will then subscribe to the point2 topic to collect results and formulate a point cloud visualization. Figure 8 gives the rqt graph of the ouster-ros package for context.

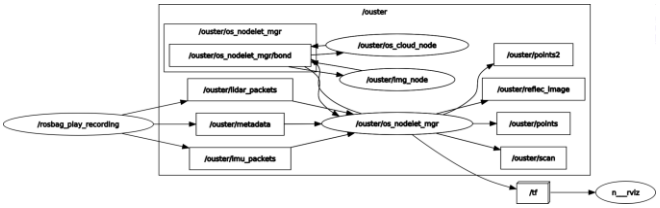

Figure 8. Screenshot of a single time step from LiDAR data.

VII. EXPERIMENT RESULTS AND SIGNIFICANCE

A. LiDAR Validation

Data collection experiments were conducted near Scaife Hall to demonstrate LiDAR's capability to capture point cloud data in structured, natural and dynamic environments. These initial experiments focused on assessing the LiDAR's performance and the reliability of the supporting setup.

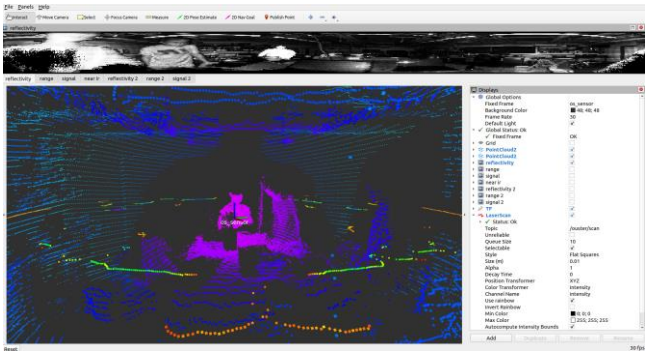

Figure 9. Point Cloud Generated Using Rviz.

Key Observations:

1. Depth Data Collection: The LiDAR successfully captures both static and dynamic objects and generates detailed point clouds. While the initial results are promising, additional experiments are required to explore its performance across various resolutions and settings to optimize the data collection process

2. Hardware: A 29.6V 4500 mAh Lithium-ion battery was used to power the LiDAR during the experiments. The battery provided sufficient power through the three experiments, ensuring stable operation. This reliability ensures a smooth execution in our future experiments.

B. Cloud Compare Results from Trials

In total, 9 trials were performed ranging from speed settings of 0.4 to 1.8. These values represent the speed setting on the robot, additional testing is required to determine how these values exactly translate to units of speed (e.g. meters per second). Two trials were performed at the 1.8 setting, with one of them (Trial 8) being labeled as an outlier due to the unexpectedly high difference in mean distance error.

Trial 1 was treated as the growth-truth of the map as it was performed at the slowest speed and is, thus,

assumed to have had the highest accuracy. The data collected from the remaining trials were compared to the map generated in Trial 1 using Cloud Compare. The computed mean distance error and standard deviation between a map of a given trial and Trial 1's map are displayed in Table I.

TABLE I. Key results collected for trials compared using CloudCompare Softtware

Trial	Speed Setting	Mean Distance	Standard Deviation
1	0.4	N/A	N/A
2	0.6	0.036	0.509
3	0.8	0.036	0.059
4	1.0	0.037	0.259
5	1.2	0.040	0.177
6	1.4	0.0438	0.445
7	1.6	0.0451	0.302
8	1.8	0.1528	0.547
9	1.8	0.0472	0.054

Mean Distance and Standard deviation are compared to trial 1, which was conducted at the lowest tested speed and treated as the ground-truth map.

The mean distance error values presented in the above table were plotted against the robot speed in Fig. 9. The outlier, Trial 8, was excluded from the graph. A linear trendline, with an R^2 value of 0.946, was plotted behind the data points for a clearer visualization of the observed trend.

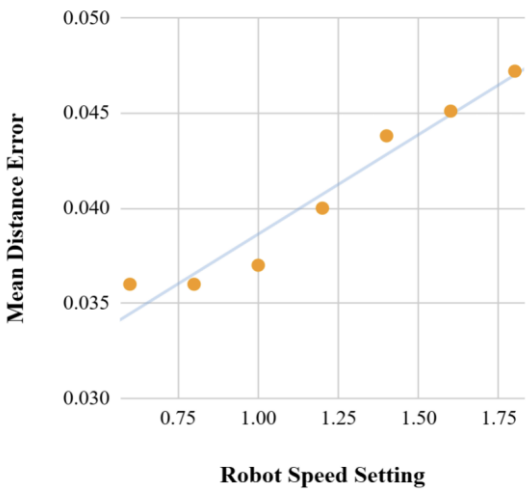

Figure 9. Robot Speed vs. Mean Distance Error.

VIII. DISCUSSION OF FACED CHALLENGES

For the training step, where the 'slow speed' and 'fast speed' parameters are being set, the initial proposal suggested conducting only one trial in which the mapping accuracy of the robot would be checked real-time every few timesteps as the robot is moving. This process would have required performing KISS-ICP realtime at a high frequency, which could be time consuming and computationally expensive.

Instead, the implementation simplified this process by comparing maps offline after the robot has completed the entire path. This change reduced the overall number of times KISS-ICP needed to be performed, and reduced the coding complexity of controlling the robot's speed.

During the implementation of the training step, hardware interfacing between the LiDAR and the rover hardware and transferring LiDAR specific code to the rover's NUC served as significant challenges.

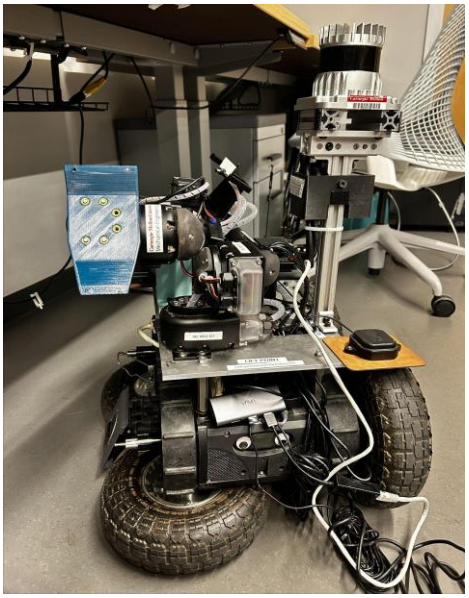

Figure 10. Patrick Under Repair after NUC Disconnect Issue.

There was an issue where the NUC was unable to receive LiDAR data through its ethernet cable, therefore 2 separate laptops had to be used in the data collection process, one for controls and one for data collection. This forced one of us to walk alongside the robot, introducing false positive points in the dataset.

On the hardware side, the laptop used for the rover's control was often unable to connect to the rover's network in the testing environment. To establish a connection, the rover setup often had to be brought outside of the lab, then brought back in. Furthermore, when stopping at high speeds, the cable that connects the NUC to its motor drivers frequently got disconnected. This required a restart of the whole system, delaying our data collection process.

From the software configuration issues and miscellaneous hardware challenges we encountered, we learned that in physical robot implementation projects, it is essential to allocate additional time for debugging and setup to account for unexpected errors that may impede overall progress.

IX. CONCLUSION AND FUTURE WORK

A. Interpretation of Results

Trial 8 being an outlier could be attributed due to human error when setting up the experiment. For example, if the robot was oriented incorrectly at the beginning, the generated would have larger discrepancies. This is further supported by the fact that the second trial using the 1.8 speed setting (Trial 9) had a mean distance error that met the initial expectations. Performing additional trials and at a greater number of speed settings would help verify the results in this study.

Overall, all the trials performed led to roughly accurate maps of the room, based on visual inspection, regardless of the speed setting used. This speaks to KISS-ICPs robustness when it comes to mapping at different speeds. However, when the maps collected from the different trials were compared to the 'ground-truth' map (Trial 1's map), the mean distance error increased as the robot speed setting increased. This trend can be observed in the Fig. 9, where the trendline has an R^2 value close to 1, indicating the relationship appears to be linear.

The trend of map accuracy declining as robot speed increases matches the expected trend for the results. One possible explanation for the increased error could be related to the nature of ICP SLAM algorithms. ICP compares the closest points between two frames to keep track of where the robot is. A larger speed means there is a greater physical transformation between the frames the LiDAR sensor picks up. This increased delta means the "closest points" in ICP are now further apart compared to a slow moving robot, which could lead to a more inaccurate representation of the transformation.

The results from this experiment serves as validation that the field of robotics would benefit from an adaptive-speed SLAM algorithm that is capable of balancing the advantages (reduced mapping time) and the tradeoffs (lower map accuracy) associated with increasing speed during mapping.

B. Future Work

With Experiment results clearly demonstrating a relationship between speed of the rover and the accuracy of the generated, our future work will focus on the following areas:

- GPS based path planning: Currently, rover's path is hard coded and tailored to a specific testing environment. A GPS based path planning system will enable more flexible and efficient testing of the algorithm in different environments.
- Adaptive speed algorithm: An adaptive speed speed control algorithm will be implemented where the rover adaptively changes its speed according to its surrounding point cloud density to optimize its mapping process.
- Field testing: Once an adaptive speed algorithm has been developed, it will be tested in various outdoor environments to validate its performance.
- Sensor Fusion: The KISS-ICP algorithm presented on this paper relied only on LiDAR data to generate the map of the environment. Using additional sensors, such as an Inertial Measuring Unit (IMU) or wheel encoders, could help provide a more accurate estimate of how the robot is moving during the mapping process. Therefore, combining the LiDAR data with other sensor readings may allow for more accurate maps being generated even at high speeds.

X. REFERENCES

[1] A. Johnson. "Robomechanics Lab." cmu.edu. Accessed: Oct. 25, 2024. [Online.] Available: https://www.cmu.edu/me/robomechanicslab/

[2] I. Vizzo et al. "KISS-ICP: In Defense of Point-to-Point ICP -- Simple, Accurate, and Robust Registration If Done the Right Way," IEEE Robotics and Automation Letters, vol. 8, no.2, pp. 1029-1036, Jul. 2023, doi: 10.48550/arXiv.2209.15397

[3] P. Besl and N. McKay. A Method for Registration of 3D Shapes. IEEE Trans. on Pattern Analalysis and Machine Intelligence (TPAMI), 14(2):239–256, 1992.

[4] Ouster. OS0 Ultra-Wide View High-Resolution Imaging Lidar. (2021). Accessed: 11/15/2024. [Online]. Available: https://data.ouster.io/downloads/datasheets/datasheet-rev05-v2p1-os0.pdf

[5] D. Seong. "david_seong8914/SLAM_project". Github. 11/15/2024. [Online]. Available: https://github.com/davidseong8914/SLAM_project

[6] J. Jeong, Y. Cho, Y. Shin, H. Roh, and A. Kim. Complex urban lidar data set. In Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA), 2018.

[7] Girardeau-Montaut, Daniel. *CloudCompare*. [Online]. Available: https://www.danielgm.net/cc/

[8] A. Cosgun. “Speed Maps: An Application to Guide Robots in Human Environments,” in *14th International Workshop on Human-Friendly Robotics (HFR)*, Nov. 2021 [Online]. Available: https://arxiv.org/abs/2111.02659