EFFECTIVE NAVIGATION AND MAPPING OF A CLUTTERED ENVIRONMENT USING A MOBILE ROBOT

A Dissertation
Presented to
The Academic Faculty

by

Pileun Kim

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Civil and Environmental Engineering

Georgia Institute of Technology
December 2020

EFFECTIVE NAVIGATION AND MAPPING OF A CLUTTERED ENVIRONMENT USING A MOBILE ROBOT

Approved by:

Dr. Yong K. Cho, Advisor
School of Civil and Environmental Engineering
Georgia Institute of Technology

Dr. Eric D. Marks
School of Civil and Environmental Engineering
Georgia Institute of Technology

Dr. Yi-Chang James Tsai
School of Civil and Environmental Engineering
Georgia Institute of Technology

Dr. Chao Wang
Department of Construction Management
Louisiana State University

Dr. Jun Ueda
School of Mechanical Engineering
Georgia Institute of Technology

Date Approved: December 2, 2020

ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to my advisor, Dr. Yong K. Cho since he has provided me encouragement and valuable advice during times I have spent with him. I would also like to extend my sincere thanks to the members of my committee: Dr. Yi-Chang James Tsai, Dr. Jun Ueda, Dr. Eric D. Marks, and Dr. Chao Wang. Moreover, many thanks to current and former lab members: Dr. Kyungki Kim, Dr. Inbae Jeong, Dr. Youjin Jang, Jingdao Chen, Jisoo Park, Kinam Kim, and Jitae Kim. I would like to thank all of the faculty and staff of Georgia Tech who helped me in my work, directly and indirectly.

Special thanks to my family: Daehwi Kim (father), Soonmi Jeong (mother), Jangok Eom (father-in-law), Sookja Jeon (mother-in-law), Pilho Kim (brother), Soojin Eom (sister-in-law), Yangjoon Park (brother-in-law), Ah-in Park (nephew), and Ah-yul Park (nephew). Lastly, I'm extremely grateful to my lovely wife, Hyebin Eom and cute babies, Chloe J. Kim and Aaron J. Kim. Everything I have, I owe to your prayers, support, and great love.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

SLAM	Simultaneous Localization and Mapping
LiDAR	Light Detection And Ranging
IMU	Inertial Measurement Unit
ICP	Iterative Closest Point
LOAM	Laser Odometry And Mapping
ROS	Robot Operating System
BIM	Building Information Models
NBV	Next Best View
NBS	Next Best Scan
LoC	Lack of Completeness
DoF	Degree of Freedom
RRT	Rapidly-exploring Random Tree
VFH	Vector Field Histogram
PCA	Principal Component Analysis
MAV	Micro Aerial Vehicles
UAV	Unmanned Aerial Vehicles
LM	Levenberg-Marquardt
RMSE	Root Mean Square Error

SUMMARY

Today, the as-is three-dimensional point cloud acquisition process for understanding scenes of interest, monitoring construction progress, and detecting safety hazards uses a laser scanning system mounted on mobile robots, which enables it faster and more automated, but there is still room for improvement. The main disadvantage of data collection using laser scanners is that point cloud data is only collected in a scanner's line of sight, so regions in three-dimensional space that are occluded by objects are not observable. To solve this problem and obtain a complete reconstruction of sites without information loss, scans must be taken from multiple viewpoints. This thesis describes how such a solution can be integrated into a fully autonomous mobile robot capable of generating a high-resolution three-dimensional point cloud of a cluttered and unknown environment without a prior map. First, the mobile platform estimates unevenness of terrain and surrounding environment. Second, it finds the occluded region in the currently built map and determines the effective next scan location. Then, it moves to that location by using grid-based path planner and unevenness estimation results. Finally, it performs the high-resolution scanning that area to fill out the point cloud map. This process repeats until the designated scan region filled up with scanned point cloud. The mobile platform also keeps scanning for navigation and obstacle avoidance purposes, calculates its relative location, and builds the surrounding map while moving and scanning, a process known as simultaneous localization and mapping. The proposed approaches and the system were tested and validated in an outdoor construction site and a simulated disaster environment with promising results.

Keywords: Laser scanning; Point cloud registration; Mobile robot navigation; Simultaneous localization and mapping; Scan and path planning; Obstacle avoidance

CHAPTER 1. INTRODUCTION

1.1 Background and Motivation

Accurate as-is three-dimensional (3D) models of cluttered environments, such as construction or disaster sites, using laser scanning devices offer many advantages. For example, they can provide virtual views and enable users to walk through the site and monitor construction progress from thousands of miles away. Engineers can use these models to analyze the stability of a structure and examine possible corrections without causing destruction. Furthermore, construction sites change and evolve over time with design changes, so an as-designed model may not match a site's current status. As-is 3D models are, therefore, useful when as-designed models are either not available or inaccurate. Methods of obtaining these models are becoming faster, more accurate, and more automated, but decision making by a human operator is still required. For example, routes and the locations of each individual scan must be determined, for which conventional methods normally rely on human intuition and experience. The requirement for multiple scans is due to the occlusion problem.

An interesting problem with the Light Detection And Ranging (LiDAR) method is that the depth information is generated from the first object on the robot's line-of-sight path, which can cause complications with line-of-sight occlusions, as shown in Figure 1. Depending on the environment, critical data may be missed due to line-of-sight obstructions, or the invisible region on the map may be treated as impassible. Moreover, these issues can cause serious concerns if the LiDAR-scanned point clouds are converted into other data forms such as voxel maps or heightmaps.

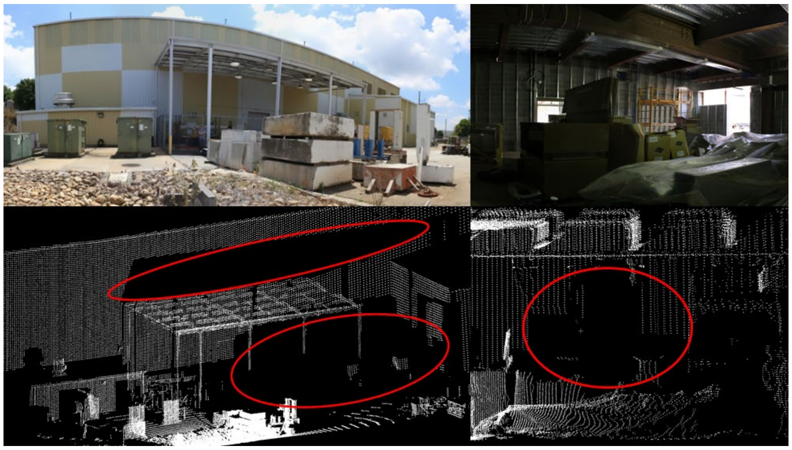


Figure 1 – Point cloud line-of-sight obstructions

To perform interactive tasks in a physical world, a robot needs a basic set of capabilities. This includes not only mobility but also the ability to be aware of the environment enough to perform the task. Therefore, most robot systems should be able to create an environment map by measuring with sensors and estimating its current position. Problems related to mapping, localization, and planning are at the heart of the mobile robot field and are mainly addressed in two-dimensional (2D) environments. In order to widely used to real-world robots, they need to be able to navigate not only simple environments but also large 3D and cluttered environments. The construction environment is dynamic, hazardous, cluttered, and irregular. One cannot know whether to expect large equipment, holes, a flat terrain, or a random pile of rubble. Therefore, a robust and reliable approach is required to enable an autonomous robot to drive safely and build an accurate 3D map. To clarify this task, it is necessary to carefully define the problems of mapping, localization, and planning.

Mapping is to collect multiple sensor measurements and correlating them into a single map. The map has to deal with the uncertainty of pose estimation and sensor noise in order to accurately correlate sensor measurements as the robot moves. Localization is to estimate the pose of a robot based on environmental maps and sensor measurements. It also has to deal with sensor noise and uncertainty in the map. In some cases, it may be impossible to identify a similar position within a map. Planning is to determine where to go next for the robot. One of the important problems for a mobile robot is path planning, which is finding a safe and fast path to a specific target location.

Figure 2 demonstrates that these three issues are interdependent. The interdependence between mapping and localization is a well-known problem that is commonly referred to as simultaneous localization and mapping (SLAM). This is because the localization estimation is required for mapping and the map is needed for localization. The path planner selects the next action to the direction of reducing uncertainty, which means that both mapping and localization rely on planning. The process of determining the next goal position for a mobile robot that reduces uncertainty in both mapping and localization is called active SLAM.[1].

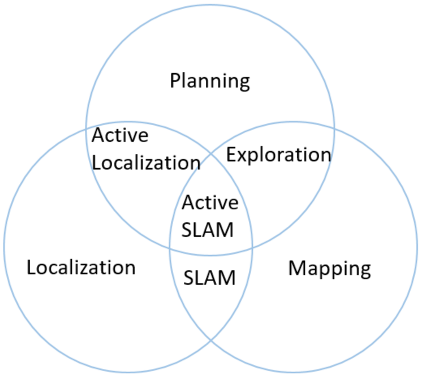


Figure 2 – Relationships between localization, mapping, and planning

To be a real application for construction or disaster sites, the obstacle detection and avoidance should be fast, robust, and not rely on the as-designed condition. There are several ways for robot navigation that work by using a priori map and planning the robot's action in advance [2,3]. However, the conventional approaches are not suitable for construction or disaster sites due to the unique characteristics of construction and disaster environments.

This thesis introduces an effective navigation and mapping method for autonomous as-is 3D geometric data acqusition by a mobile robot at a cluttered environment with many spatial uncertainties. The proposed architecture was implemented and tested with a custom-designed mobile robot platform, Ground Robot for Mapping Infrastructure (GRoMI), which uses multiple LiDARs and sensors to sense and build a 3D environment map. To evaluate the overall navigation and mapping performance of GRoMI and the proposed software architecture on construction and disaster sites, experiments were conducted on several simulated environments and two real environments.

1.2 Research Objectives and Contributions

1.2.1 Objectives

The main objective of this research is to identify an automated and intelligent 3D reconstruction method of the built environment for a ground laser scanning robot in an unknown cluttered environment by finding self-determined scan locations without a prior map.

1.2.1.1 Sub-objective #1: Identify and remove the ground and moving objects in the point cloud

The first research question to be solved is how a mobile robot chooses where to go to move safely and improves localization accuracy. For a mobile robot to explore an unknown environment, it must be able to localize and map of surrounding. Especially, it is important to identify the ground near the robot is traversable or not and the objects near the robot are moving or stationary. Because it is related to safe navigation without collision and SLAM accuracy. To achieve this objective, the mobile robot must be integrated with a laser scanning system and must be able to solve the kinematic problem to acquire highly accurate, dense 3D point cloud data using rotating 2D LiDAR scanners. When it can build a 3D point cloud, it must extract edges in the point cloud using range differences between adjacent points. If there is no edge in a region of the point cloud, this means that the region is continuous, and there are no obstacles. Based on this approach, a robot can estimate the unevenness score and segment out ground points. Then, non-ground points can be divided into independent several clusters by using Euclidean distance between them and the cluster correspondence of consecutive point clouds is performed. The centroid of each cluster is extracted and tracked the location to specify moving objects. Finally, the points identified as a moving object are removed from the extracted features which will be used for odometry calculation. The tasks for this sub-objective is as follows:

a) Solve the kinematic and odometry model for the 3D scanning system

b) Calculate terrain unevenness to identify ground points

c) Extract moving objects and remove it from odometry calculation

The expected outcomes are point clouds which have not include both ground and moving objects points.

1.2.1.2 Sub-objective #2: Estimate a navigation goal and plan path to the goal

The second research question to be solved is how a mobile robot can avoid both static and dynamic obstacles in a cluttered environment. Path planning is to determine an collision-free path to reach the goal position. The information that obtained in sub-objective #1 is stored in a 2D surface map, called a robot navigation map and the path planning is based this 2D map. Before plan the path to the goal, the navigation goal should be determined. The frontier-based approach is used to select a temporary navigation goal in this research. The reason the term temporary goal is used that it is not the final goal or the stationary scan location goal. The stationary scan location is determined in the sub-objective #3 using scan view planning. Once the navigation goal is selected, the global planner estimates the shortest path to the goal and D* Lite algorithm is used as global planner in this sub-objective. While it moves along the estimated path, it keeps scanning and segmenting obstacles, ground, wall, and occlusion area. If the mobile robot faces unexpected dynamic obstacles on the planned path, the global planner replaced to the local planner and it estimate the velocity of moving obstacles. For this study, potential field method is used as the local planner. The global and local planner generates the waypoints to each goal location. Finally, the path tracking controller, running at a higher frequency than the both global and local planner, continuously generate control inputs which is right and left wheel velocities for a differential drive robot to ensure stable navigation from the current state to the next waypoint. The tasks for this sub-objective is as follows:

a) Determine a navigation goal location and its path for global planner

b) Update the path in real-time to avoid dynamic obstacles for local planner

c) Guide the mobile robot to waypoints with tracking control

The expected outcomes are waypoints to both planner, and wheel motor speed for both side.

1.2.1.3 Sub-objective #3: Estimate the most effective scan view

The third research question to be solved is how a mobile robot can determine scan locations in an unknown environment without a priori information. To generate highly accurate dense 3D point cloud data, this thesis uses a "stop and scan" strategy rather than a "continuous incremental scan" as the latter generates noise and inaccuracies in the point cloud data. Finding efficient scan locations for the target site is one of the key contributions of this thesis. While the mobile robot moves to the determined exploration position, it keeps scanning and builds a map for navigation and obstacle avoidance. It can also generate a voxel map, which is a downsampled map that uses small cubes to reduce the computational load. It then performs scan planning in the currently built map to select the scan location that is able to identify locations that will yield new information as many as possible when visited, which means to find the occluded area that can be scanned to complete the target area map. The tasks for this sub-objective is as follows:

a) Generate a voxel map using the currently built point cloud

b) Determine the scan locations through real-time scan view planning

The expected outcome is a voxel map containing the scan locations

1.2.2 Contributions

This thesis first presents a detailed literature review of existing autonomous mobile mapping robotic systems and identifies the major limitations and remaining challenges. To address the shortcomings of existing state-of-the-art robotic systems, this thesis develops a novel framework for effective navigation and automated, accurate 3D point cloud mapping of a cluttered environment using a mobile laser scanning system.

- It presents traversability estimations for point clouds with unevenness scores using range differences between adjacent points.
- It provides moving objects detection and removal from SLAM calculation to improve localization accuracy.
- It introduces an autonomous navigation approach that combines a grid map based on traversable path planning and a voxel map based on scan view planning.
- It offers a real-time active SLAM approach that integrates 3D LiDAR SLAM, planning algorithms, tracking control, and dynamic obstacle avoidance.

In summary, the proposed approach eliminates the need for physical visits to the site, thereby improving construction site monitoring capabilities and saving time and money. This may result in changes to the decision-making process and the ways in which modern construction projects are monitored and controlled, which, in turn, may affect future construction.

1.3 Thesis Structure

The rest of the thesis is organized as follows:

Chapter 2 presents a detailed review of existing state-of-the-art autonomous mobile scanning robotic systems. Existing state-of-the-art mapping robotic systems are discussed in detail, with a focus on their capabilities and shortcomings. The need for a novel framework or system for autonomous mobile mapping is presented at the end of the chapter.

Chapter 3 presents a novel software architecture for autonomous laser scanning and the design of a robotic system. It also provides validation of each architecture module with simple simulated environments.

Chapter 4 presents the results of simulations, real construction and simulated disaster environment conducted to validate the overall proposed architecture. It then explains how the experimental results using the techniques described in Chapter 3 improved the planning, control, and error detection of autonomous robots.

Chapter 5 concludes the thesis by providing a summary of the presented work and a discussion about potential directions for future research.

CHAPTER 2. LITERATURE REVIEW

2.1 Objective Detection and Traversability

Most researchers tried to use 2.5-dimensional representation which is trade-off between the completeness of 3D and the simplicity of 2D representation for traversability analysis and was used successfully in the defense advanced research agency challenge [4]. Identifying obstacles is achieved by height difference or elevation map [5]. This approach proves difficult when dealing with overhanging objects, such as tree branches. It could be dealt with by employing a safe height [5] or by using extended elevation maps [6], in which the obstacles above a free space are safely discarded. Chang et al. [7] tried to use the slope for detecting obstacles. The height difference threshold was 1 m to detect large objects faraway so the smaller obstacles could not be detected. Morton and Olson [8] utilized a height–length–density terrain classifier which could even deal with partial observability for detecting obstacles. Kuthirummal et al. [9] utilized graphs to connect traversable cells containing height histograms. Guo et al. [10] developed gradient cues of road geometry to classify points into four classes: reachable, drivable, obstacle, and unknown regions by using Markov random fields.

Principal component analysis (PCA) can analyze the spatial distribution of points with eigenvectors. Lalonde et al. [11] classified areas into scattering, linear, and surface with PCA on individual voxels. Pellenz et al. [12] applied PCA recursively on non-flat regions to overcome the problem of the sparseness of data in a grid. Sinha and Papadakis [13] assessed traversability based on features extracted through PCA. Anguelov et al. [14] also used PCA to get point features for further classification. However, the PCA approaches are either computationally expensive or require training in classification methods, such as neighborhood search, model fitting, machine learning, Gaussian Regression, and PCA.

2.2 Autonomous Navigation and Path Planning

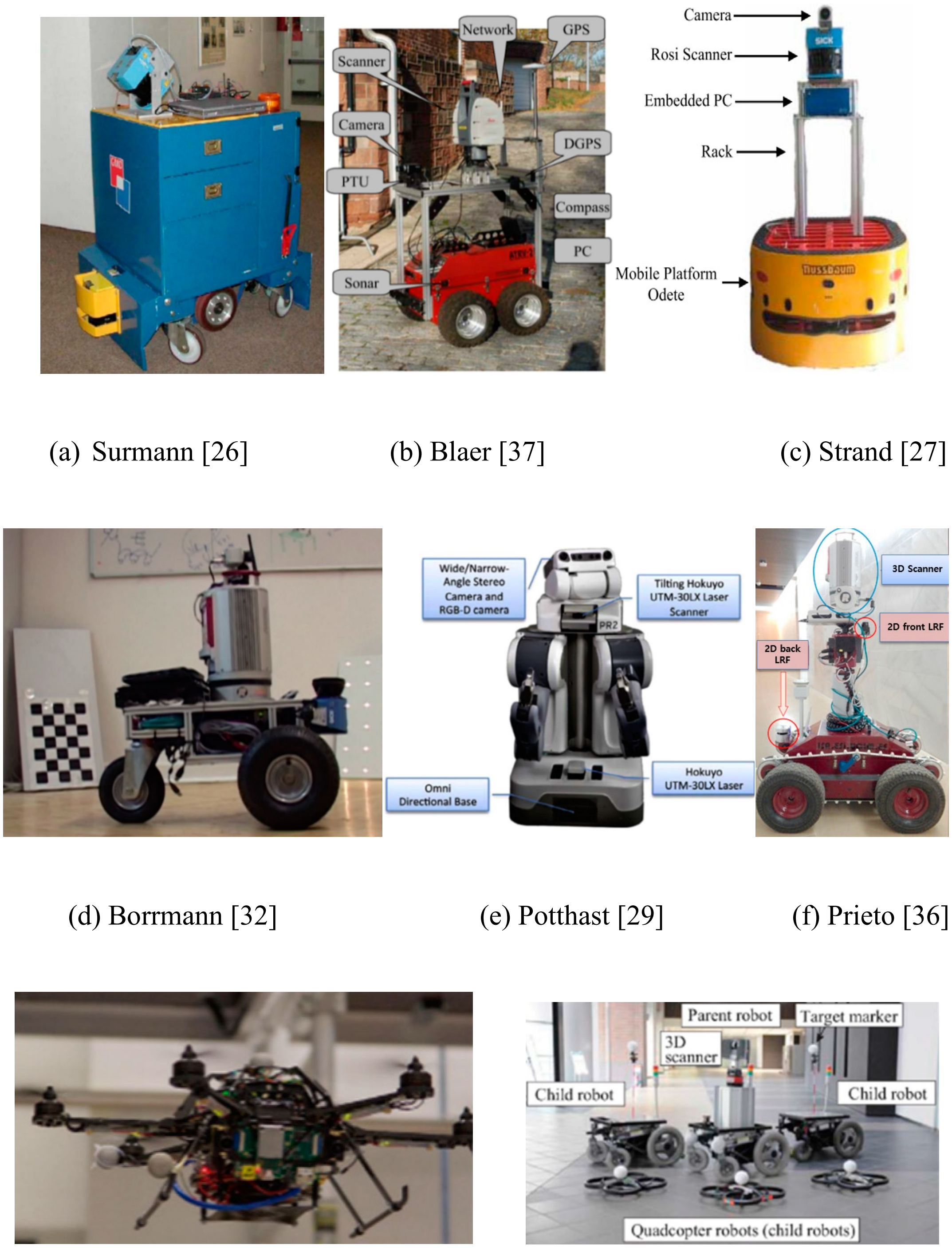
The optimal or feasible path between the start point and end point can be found using motion planner algorithms such as Bug algorithm [15], A* [16], Rapidly-exploring Random Tree (RRT) [17], and RRT* [18]. Bug algorithm is a simple planner applicable in 2D and practical for objects with two degrees of freedoms. RRT is a sampling-based method that can find a path with more than two degree of freedoms by constructing a space filling tree. RRT creates a path by sampling random seeds. RRT* is more optimized than RRT which provides a smooth path [19]. Unlike A* and Bug, RRT and RRT* can be used in cases with many degree of freedoms. These algorithms aim primarily to find a feasible, collision-free path, but without guaranteeing high visibility during inspection [20].

One of the widely used algorithms for mobile robot and autonomous vehicle navigation is D*, which was evoled from A*. The advantage of D* is adding new information to its map, when it enters in an unknown environment or encounters obstacles, to find a new shortest path from current location to the goal. Another path planning algorithm is the Bug algorithm, where the robot moves in a straight line toward goal till it senses an obstacle and the robot avoids obstacles by deviating from the line while updating new important information. Bug1 and Bug2 [21] are most commonly used path planning algorithm but it has a weaknesses that it does not garantee to find the optimal and shortest path in general. It also is not a good solution in complex areas due to its simplicity, especially in a construction site having many unpredictable objects in the environment. VFH was designed to avoid unexpected and moving objects while simultaneously steering a robot toward the goal position [22]. However, VFH does not effective especially in construction environment which needs to scan a large area because the occupancy grid map requires a huge memory to be able to keep count of the values in each cell. Furthermore, it is hard to avoid the local minima problem [23]. The primary idea of Dynamic Window Approach method is generating a valid search space, and choosing an optimal path from the search space [24]. The optimization goal is to minimize the time by moving fast as much as possible in the optimal direction.

2.3 Mobile Scanning Platforms

Mobile scanning systems are increasingly being used in industrial, construction and artificial intelligence applications. In particular, autonomous scanning plays an essential role in the field of automatically generating 3D models of buildings. However, creating a 3D model of a building from 3D data is still a semi-manual task. A fully autonomous system is a system that can perform exploration, 3D data collection, and 3D data processing without human interaction without initial knowledge of the scene. A representative example of a mobile scanning robot is shown in Figure 3. Sequeira et al. [25] present a simple autonomous robot that partially digitizes a single room with a Time-of-Flight laser range finder. Surmann et al. developed the Ariadne robot [26], a mobile platform with a 3D LiDAR which can digitalize a large indoor environment. The Rosete platform is proposed by Strand et al. in [27]. They developed a rotating laser scanner mounted on a mobile robot and scaned simple indoor environment to overcome the small field of view. ATRV-2 AVENUE is designed to collect data for large outdoor environment. The system had a high degree of autonomy, but it is needed a priori knowledge about the target site. The scanning robot proposed by Blodow et al. [28] explored indoor environments autonomously and generated a map from colored point clouds. The same robot is used in [29], but it tested the Next Best View (NBV) algorithm in small environments. Charrow et al. [30] presented a ground robot equipped with a 2D LiDAR and an RGB-D camera to build 3D map of indoor environments. Iocchi et al. [31] got 3D maps of buildings with a 2D LiDAR, stereo-vision, and IMU on a mobile platform. Borrmann et al. [32] proposed Irma3D, a robotic platform that automatically generates 3D thermal models of indoor environments.

The usage of micro aerial vehicles (MAV) is increasing as an autonomous platforms to capture 3D information from indoor and outdoor environments. Bircher et al. [33] presented a new path-planning method to explore autonomously for finding an unknown volume using a hexacopter and a stereo camera. Rusu et al. [34] tried to acquire kitchen 3D maps with recognized nearby objects. The output is a coarse 3D model composed of cuboids and planes that represent relevant objects, such as containers or tables. Kurazume et al. [35] developed a multiple robot team that scans indoor and outdoor environment cooperatively. The system was consist of a parent platform equipped with an 3D LiDAR system and several child robots, including both ground and aerial robots. The parent robot generated 3D model and child robots used to help a precise localization and registration. Finally, the platform MoPAD [36] was able to build detailed 3D models of the buildings interior. Table 1 presents a summary of representative autonomous mobile scanning platforms, including the tested environment and limitations.



(a) Surmann [26] (b) Blaer [37] (c) Strand [27]

(d) Borrmann [32] (e) Potthast [29] (f) Prieto [36]

(g) Bircher [33] (h) Kurazume [35]

Figure 3 – Example of autonomous scanning system

Table 1 – Summary of mobile scanning platforms.

Method	Year	Tested Environment	Limitations
Surmann [26]	2003	Corridor	Too many scans for simple trajectory
Blaer [37]	2007	Interior and exterior scenes	Previous 2D map needed Localization based on GPS
Iocchi [31]	2007	Hallway and some rooms	Previous 2D map needed
Strand [27]	2008	Hallway and some rooms	Low 3D model resolution
Blodow [28]	2011	Single room	Based on 2D NBV
Borrmann [32]	2014	Corridor and several rooms	Only walls are obstacles
Potthast [29]	2014	Two rooms and corridor	Restricted on planned scenarios. Accumulative registration error
Charrow [30]	2015	Long corridor	Only 2D exploration and map
Bircher [33]	2016	Indoors and outdoors	Low resolution of 3D model
Prieto [36]	2017	Complex configuration of adjacent rooms and corridors	Too long for preprocessing stage
Kurazume [35]	2017	Indoors and outdoors	Planning in 2D plane. High complexity of the system
Meng [38]	2018	Indoors	Low resolution of 3D model

2.4 Next Best Scan Planning

The most important things to automatic scanning with occlusion is that of a good selection of the next scannig location. This is known in the literature as the NBV problem [39], but in our context, this could be renamed as Next Best Scan (NBS). Most of autonomous scanning platforms utilized the 2D map of the target site and estimate the scan lcoation based on the future visibility, but sometimes it faced to high levels of occlusion [40]. It also employed the frontier-based approach which is 2D map based as a starting point [41]. However, 2D map provide incomplete information in terms of occlusion of a 3D real world, so it brings to erroneous or non-optimum locations.

The 3D-based NBV algorithms was frequently used for exploring volumes and inspecting surfaces of relatively small objects not a large-scale sites. Blaer et al. [37] presented a four step planning method. At first step, a coarse 3D model is collected with a 2D map. It nominates several randomly selected scan positions for second step. Then, it determines an optimal positions which covers the boundaries of the free space. For last step, the obtained coarse 3D model is refined. Surmann et al. [26] proposed a 2D-3D combined approach that computes several NBVs and selects the best option. Borrmann et al. [32] also presented a similar 2D-3D mix algorithm. It enables to move the robot with 2D NBV until it recognizes an enclosed space. After then, it tries to use a 3D NBV algorithm, which is similar to Blaer et al. [37]. Potthast et al. [29] propsed a probabilistic NBV with Markov Random Fields. The method allocates the probability to be seen in the next scan location to each voxel. Meng et al. [38] generated an octree structure and defined the NBV location by using the volumetric information gain approach. They presentd a two-stage planner, consisting of a frontier-based coverage strategy and a fixed start open traveling salesman scheme. Charrow et al. [30] suggested a combined planning approach with global planning and local motion primitives.

2.5 Point of Departure

Autonomous scanning became more and more important as the sensors are improved. Even though some autonomous scanning platforms can scan specific environments and simple scenarios autonomously, there are still a number of challenges which are related to the achievements and limitations of autonomous scanning methods. As the above literature review shows, the current systems still have gaps and serious weaknesses that need to be addressed to enable the creation of autonomous systems that work in realistic environments.

- Most researchers have used pitching or rolling scan mechanisms because they are effective in obtaining information about the front region. However, the advantage of a yawing scan is that its field of view can be widened to 360°. The laser scanning system used in this proposal uses four 2D LiDAR scanners within a continuously rotating mechanism (yawing) to build a 3D point cloud. The reason for using four LiDAR scanners is to reduce the time required to achieve for 360° scanning.

- The state-of-the-art technology for registering large-area scanning is continuous incremental mapping. However, this approach results in noise and cannot easily obtain a high-resolution point cloud. Therefore, the laser scanning system used in this proposal uses a combination of stop-and-scan and continuous methods to reconstruct an environment.

- There are many algorithms for detecting obstacles and computing traversability. However, most approaches use computationally expensive methods or require a machine learning process for classification. The proposed method uses simple point-level segmentation to compute a unevenness by using the ranges and scan geometry of LiDAR data.

- Most researchers who have attempted to develop autonomous systems have used a previously built 2D map and NBV planning on a 2D plane in the indoor environment. However, the approach used here combines 3D SLAM, effective scan view planning with 3D NBV, and path planning considering terrain unevenness to build a high-resolution point cloud with no prior information about the environment.

CHAPTER 3. METHODOLOGY

3.1 Problem Statement

Exploration in mobile robotics is an iterative process that involves navigation. During exploration, the robot autonomously moves through an unknown environment with the aim of creating a map of the environment that can be used for subsequent navigation. Exploration is closely related to the other essential tasks of mobile robotics shown in Figure 1. It involves creating a map of the environment, which can then serve as a model for planning further exploration steps. The central question in autonomous exploration is: “given what you know about the world, where should you move to gain as much new information as possible?” [41]. A simplified generic exploration frame entails in the following five steps:

1. Scan and integrate: capture sensor measurements (e.g., a point cloud) and integrate them into the map.

2. Evaluate: find and evaluate appropriate goals; select the next goal.

3. Plan: create a motion plan (trajectory) to reach the next goal.

4. Move: execute the motion plan created in the previous step.

5. Repeat: unless a termination criterion has been met, continue to repeat these steps.

The termination criterion for exploration can be specified as complete or almost complete coverage of the search space.

Search problems are closely related to exploration. A generic search algorithm frame would look very similar to that presented above, but it would differ not only in the termination criterion (object searched for has been found) but also in the evaluation step. Whereas exploration strives to minimize the overall time needed to explore the whole search space, search attempts to minimize the time needed to find the object it is searching for.

A famous problem linked to exploration is the art gallery problem illustrated in Figure 4. Also called, the museum problem, this is a well-studied visibility problem in computational geometry. It originates from the real-world problem of guarding an art gallery and, specifically, of computing the minimum number of guards who together can observe the whole gallery. The task is to find the optimal placement of guards or cameras on a polygonal representation of a known map, such that the entire space is covered by their field of view.

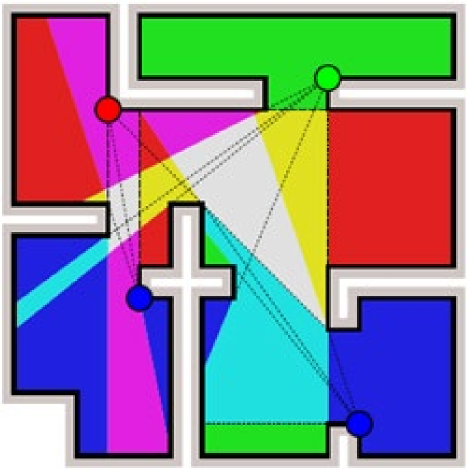


Figure 4 – Art gallery problem illustration from Wikipedia

The frontier-based class of exploration algorithms originally proposed by Yamauchi [41] represent a classic exploration approach in 2D. A frontier can be defined as the boundary between known unoccupied space and unknown territory. The frontier-based approach is based on the idea that to gain the newest information from the environment, the robot (or its sensor) should move to the boundary between free space and unknown territory. Figure 5 shows a simple exploration scenario illustrating the concept of frontiers. In this thesis, these problems will be expanded to 3D with the concept of NBV which computes a sequence of viewpoints until an entire scene has been observed by a sensor based on the above approaches.

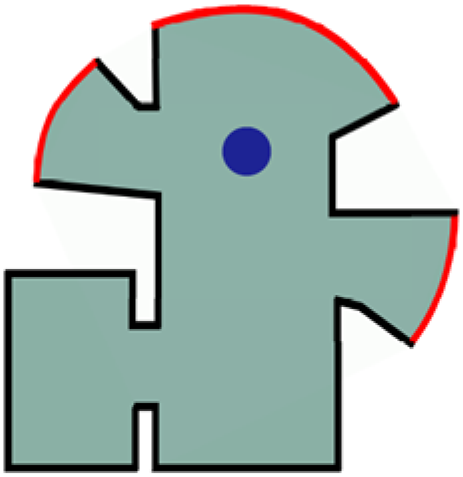


Figure 5 – Concept of frontiers from Wikipedia: obstacles shown in black, explored free areas in grey, and frontiers in red

In this thesis, the ground robot for site mapping must be suitable for deployment, ranging from construction sites to war zones, and disaster areas, therefore, it should be easy and intuitive to operate. Due to their inherent mechanical advantage, wheeled systems can pass over many obstacles. However, there are cases where the robot might roll over or get stuck. Even with recent advancements in the autonomous capabilities of robotic systems, state-of-the-art robotic systems cannot match the decision-making capabilities of a human operator especially for applications in disaster scenarios. However, as Figure 6 shows, even teleoperated systems can fail under rough terrain conditions. This thesis aims to improve the traversability of ground robots on challenging terrain conditions in both autonomous and semi-autonomous operating modes.

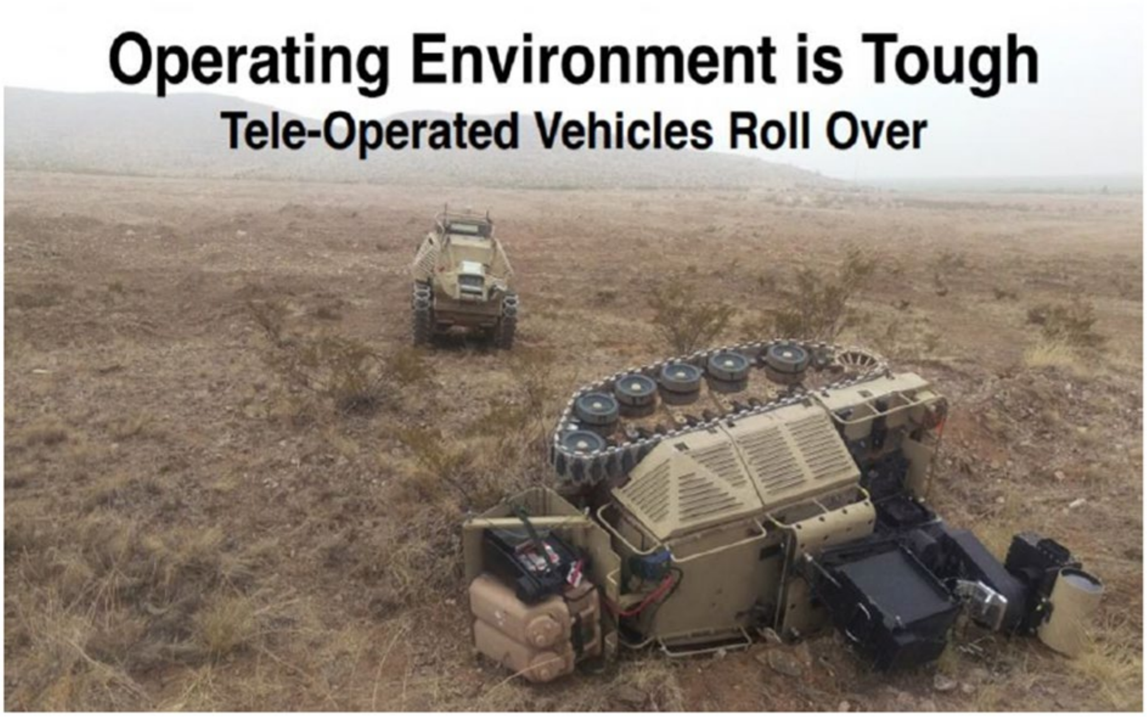

Operating Environment is Tough
Tele-Operated Vehicles Roll Over

Figure 6 – Failure of teleoperated tracked robotic systems. K. Massey, "Squad Mission Equipment Transport (SMET): Lesson Learned for Industry," 2016

3.2 Map Representation

This section describes two map representation methods used in this thesis: occupancy grid cell mapping, a fundamental mapping approach that uses range sensors, and OctoMap, another probabilistic type of mapping that uses an efficient tree-based data structure to enable 3D mapping of relatively large environments.

3.2.1 Occupancy Grid Maps

Occupancy grid cell maps, shown in Figure 7, were originally designed for sonar sensor-based mapping, which was introduced in [42]. Sonar sensors provide rather inaccurate conic measurements, so multiple readings must be used to obtain a reasonable model of the environment. Today, they are commonly used in combination with 2D LiDAR-based mapping with 3 Degree of Freedom (DoF) localization, e.g. in Hector SLAM [43], SPM-ICP [44], and gmapping [45].

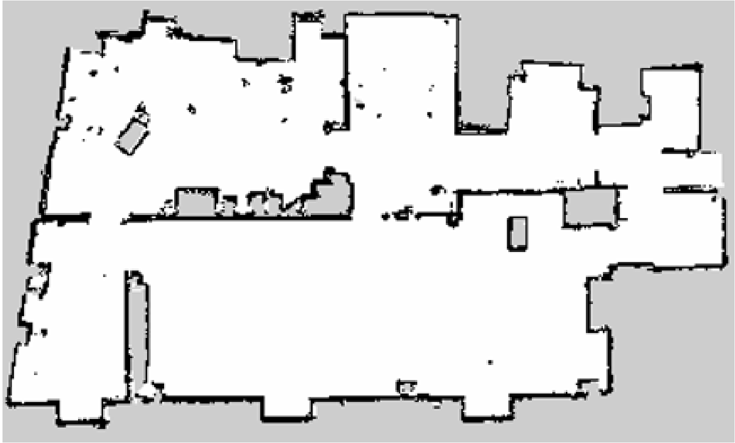


Figure 7 – An example of occupancy grid cell map [48] (Black cells are occupied, white means free, and grey represent unknown space)

3.2.2 Octree Maps (OctoMap)

OctoMap [46] is octree-based mapping [47]. It is a highly optimized framework for volumetric 3D mapping. Similar to occupancy grid approaches, OctoMap represents the environment using random binary variables. The data structure of an octomap is an octree, which is a tree structure with non-leaf nodes, each of which has eight descendants. Each node represents a cubic volume (voxel) and stores the corresponding probability of being occupied. Descendant nodes split the voxel represented by their parent node into eight sub-voxels, as shown in Figure 8 [48]. Figure 8 demonstrates how voxels are extended to sub-voxels when sensor measurements are collected. Unknown space is represented by transparent voxels, occupied space by black voxels, and free space by gray voxels. Memory can be saved by avoiding unnecessary branching when nodes represent free space.

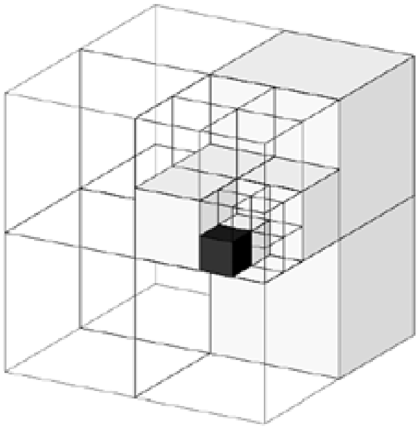


Figure 8 – Octomap map representation [48] : occupied shown in black, free areas in grey, and unknown in transparent

The downside of the octree-based data structure, compared to other structures such as that of array-based grid maps, is that direct map access is not possible. The log odds notation is used to store occupancy probabilities. This enables the map to be updated in a computationally efficient manner and direct rendering of the map in different resolutions.Figure 9 presents an example of the same map rendered in four different resolutions.

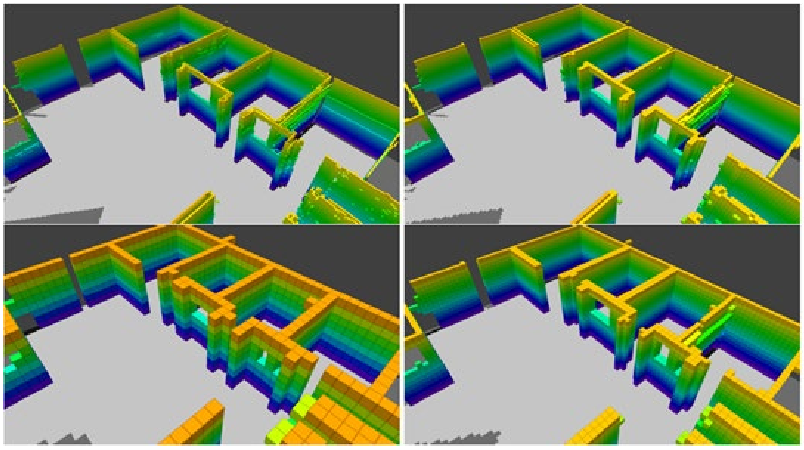


Figure 9 – Four different resolutions of the same octomap [48]: from the top left in clockwise 0.05 m, 0.1m, 0.2 m, and 0.4 m

3.3 Overall Architecture

The overall framework for the proposed research is shown in Figure 10. There are five main procedures: (i) 3D SLAM, (ii) navigation, (iii) a scan view planning, (iv) stationary scanning, and (v) point cloud registration. When the mobile robot platform with the hybrid 3D laser scanning system is moving within an environment, it calculates its current position and orientation using both current and previous maps. First, the robot takes input about the target scanning region boundary with the coordinate which has an origin at the initial location of the robot. Then, the robot turns on the 3D SLAM to estimate the current location and collect environmental information. The global planner determines a local goal position within the scanned area and plans the path to reach the local goal as a set of waypoints. These waypoints are then sent to the path tracking controller which takes a step to each waypoint. While it moves to a local goal position, the robot receives measurements from the laser scanning system, which it uses to update the surrounding map. Using these measurements, the scan view planning module tries to

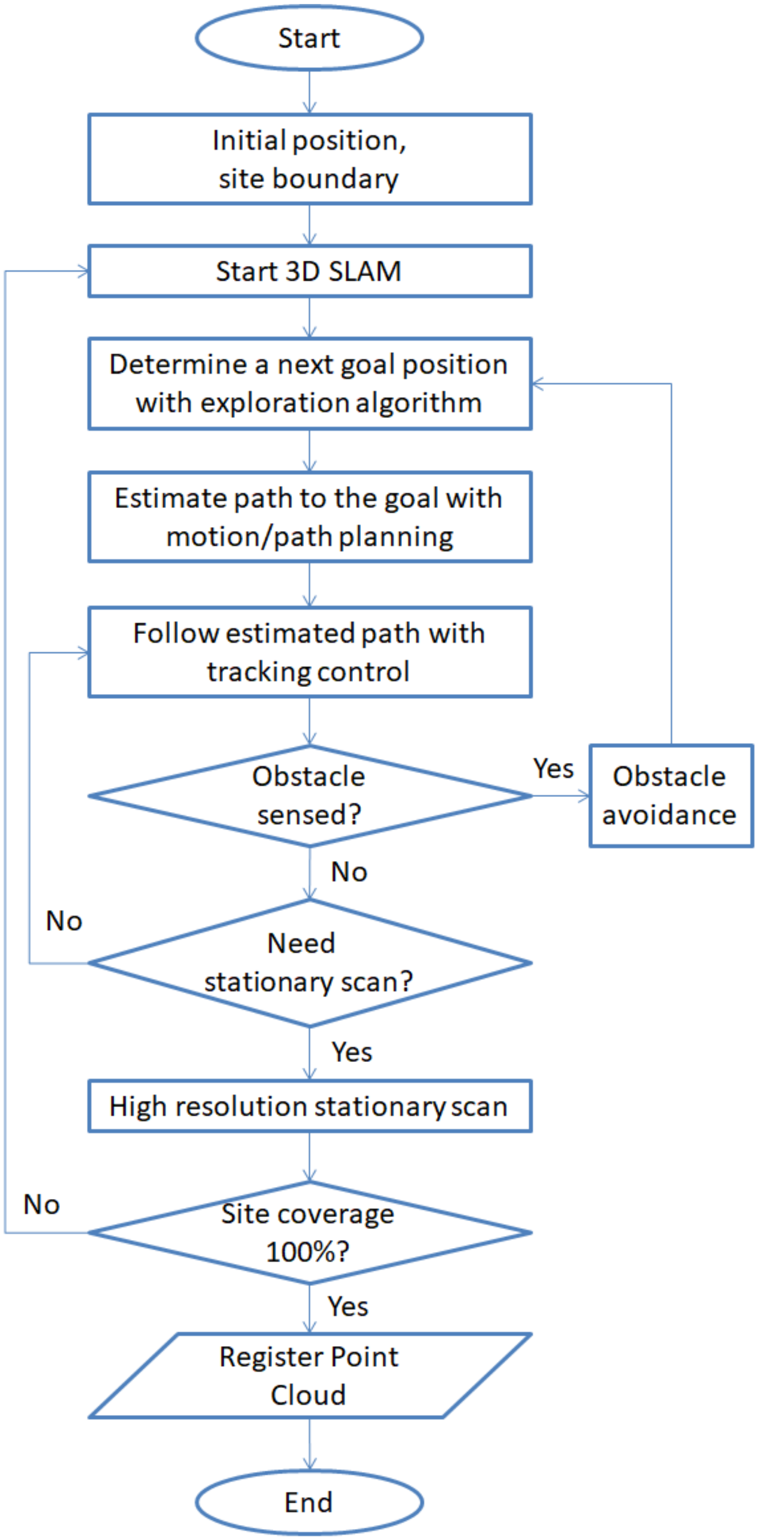

Start
Initial position, site boundary
Start 3D SLAM
Determine a next goal position with exploration algorithm
Estimate path to the goal with motion/path planning
Follow estimated path with tracking control
Obstacle sensed?
Yes
Obstacle avoidance
No
Need stationary scan?
No
Yes
High resolution stationary scan
Site coverage 100%?
No
Yes
Register Point Cloud
End

Figure 10 – Flowchart of the proposed architecture

calculate where a good scan location in the known area that measurement reached is. Once the stationary scan location is determined, the robot sets a new goal position and tries to move to that location to perform a stationary scan and generate a high-resolution point cloud. It then returns to the first step, 3D SLAM, to determine a new local goal position and finds a path between the current position and the local goal. If the robot senses obstacles while it travels on the determined path, it switched the artificial potential field module to avoid the obstacles. Once its view is clear of obstacles, the robot requests a new local goal position. This process is repeated until the target region is scanned completely.

3.4 Data Acquisition System

3.4.1 Hardware Design and Sensors

In this thesis, a custom-built ground robot, GRoMI, was used to verify the proposed method. GRoMI is composed of two major parts: a hybrid laser scanning system and an autonomous UGV platform, as shown in Figure 11 along with its subsystems. The GRoMI was designed for weighing around 135 kg (300 lbs), with an overall size of approximately 4ft x 3 ft x 6 ft (L x W x H).

The upper part is a robotic hybrid laser system, consisting of five 2D and one 3D LiDAR devices. The 3D LiDAR, commonly used for autonomous vehicles and advanced driver-assistance systems, was mounted on the top of the laser scanner system and utilized to acquire real-time laser scans of the site. It has a range of up to 100 m with a ±15° vertical field of view. As it has only 16 scan lines in the vertical direction, the scanned point cloud gets sparser as it gets farther from the scanner. Therefore, this 3D LiDAR was used only for the localization process. For building a high-resolution 3D point cloud map, the vertically-mounted 2D LiDARs were used to scan the environment at calculated scan locations. Each line laser is mounted vertically and has a range of up to 80 m with 0.1667° resolution in a vertical direction. The cylinder-shaped scanning system rotates to scan the environment, accumulates the scan data, and generates a high-resolution 3D point cloud map. It takes 200 seconds to rotate 360°, and the scan frequency of each line laser is 50 Hz. Therefore, the point cloud resolution of the laser scanning system is 0.036° in a horizontal direction. The scanning system also has a camera for mapping the texture to generate a more realistic 3D point cloud map.

On the other hand, the lower part is an autonomous platform. The most important thing for the platform is to make a division for the front and back parts of the ground robot and connect with a rotating bearing axis to stabilize when it moves in uneven terrain. The autonomous platform has four wheels and it controls as a set of left wheels and right wheels. Therefore, the robot can turn around with moving a different direction for the left and right wheels. All other sensors are attached to upper laser scanning system, but GPS/IMU sensor module for increasing the localization accuracy is mounted on lower autonomous platform. The upper laser scanning system has more sensor noise sources for IMU than a lower autonomous platform while the robot is moving and also the laser scanning system should rotate to get a high-resolution point cloud, which could be another sensor noise source.

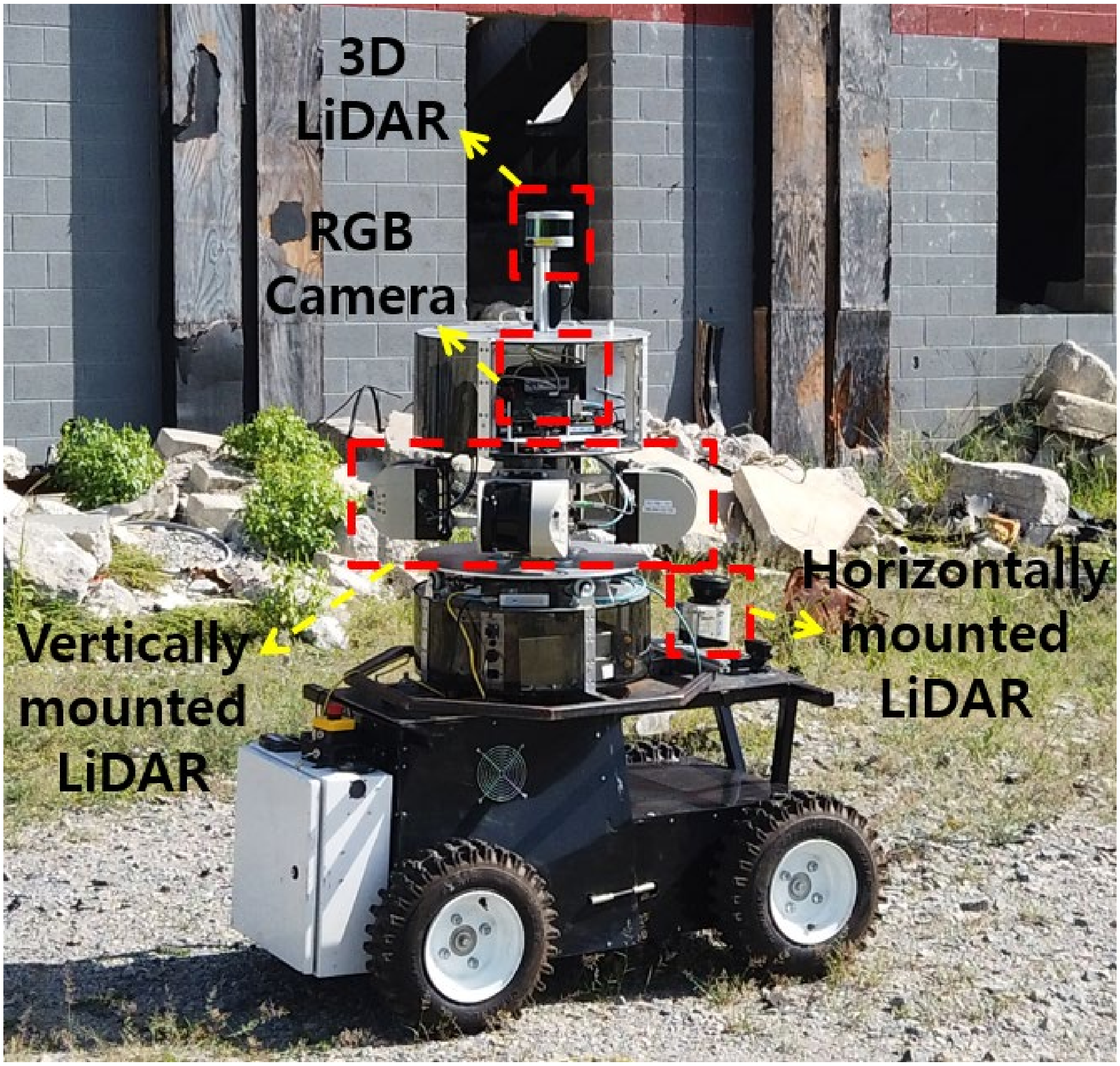


Figure 11 – Mobile robot platform with a hybrid laser canning system

3.4.2 Software and Communication Architecture

The GRoMI system is designed to be operated fully autonomous and capable of specifying a target area in construction or disaster sites. This requires the robot to be able to navigate autonomously in rugged terrain, and to overcome the challenges introduced by dynamic environmental interactions. To achieve this aim, it must play a role in a distributed communication middleware and provide an infrastructure that makes it possible to connect components easily and facilitates modular design in both simulated and real environments. This project uses the Robot Operating System (ROS), the most well-known robotics middleware, comprising a collection of software frameworks for robot software development. It also provides useful libraries (including libraries for coordinate frame transformations, point cloud manipulation, visualization, and data logging to name a few) into which it groups tools. Although ROS is not an operating system, it provides services such as hardware abstraction, low-level device control, implementation of commonly used functionality, message passing between processes, and package management, which are designed for a heterogeneous computer cluster. Running sets of ROS-based processes are represented in a graph architecture where processing takes place in nodes that can receive, post, and multiplex sensor data and messages, including those related to control, state, planning, and actuation.

A node is a process that performs computation. Nodes are combined in a graph and communicate with one another using streaming topics, remote procedure call services, and the parameter server. The nodes are meant to operate at a fine-grain scale, and a robot control system will usually comprise many such nodes. For example, a system may have one node to control a laser rangefinder, another to control the robot's wheel motors, another to perform localization, another for path planning.

Topics are named buses over which nodes exchange messages. Topics have anonymous publish-and-subscribe semantics, which decouples the production of information from its consumption. In general, nodes are not aware of who they are communicating with. Instead, nodes that are interested in data on a topic subscribe to that topic; nodes that generate data publish the data to the relevant topic. A topic might have multiple publishers and subscribers. For example, a node that controls a laser rangefinder would publish a laser scan topic message. The localization node might subscribe to the laser scan topic to estimate the current location of the robot, while the wheel motor control and path planning nodes might subscribe to the same topic to avoid obstacles and determine the path to the goal.

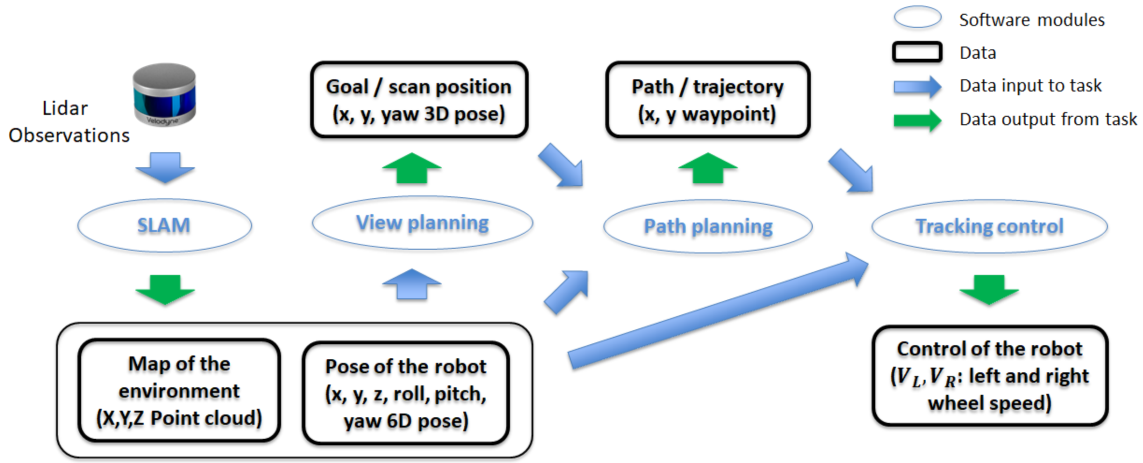

Software modules
Data
Data input to task
Data output from task
Lidar Observations
SLAM
Goal / scan position (x, y, yaw 3D pose)
View planning
Path / trajectory (x, y waypoint)
Path planning
Tracking control
Map of the environment (X,Y,Z Point cloud)
Pose of the robot (x, y, z, roll, pitch, yaw 6D pose)
Control of the robot (V_L,V_R: left and right wheel speed)

Figure 12 – Data flow diagram

Figure 12 presents a data flow diagram of the proposed software architecture in the ROS environment during navigation. The blue circles denote ROS nodes, and the black rectangles denote ROS topics. The blue arrows indicate subscription to a topic by a node and the green arrows indicate publication to a topic by a node. There are four major nodes: SLAM, scan view planning, path planning, tracking control. The SLAM node subscribes to the 3D LiDAR topic message and publish the estimated pose of the robot in 6D (x, y, z, roll, pitch, and yaw) and the map of the environment in the point cloud. These SLAM outputs are the inputs for the scan view, path planning, and tracking control modules. The scan view planning node computes the best position for the next scan on the 2D plane, which comprises the x-y plane and the yaw angle. Then, the path planning node generates waypoints to that position. Finally, the tracking control node calculates the wheel speed necessary for the mobile robot to reach the waypoints and the goal.

3.4.3 Kinematic Solution and RGB Data Fusion

The customized laser scanning system provides more flexibility in terms of hardware control and software programming compared to a commercial LiDAR scanner. For example, the real-time sensor data is stored and processed to generate a 3D point cloud integrated with RGB values by using the kinematic solution of the laser scanning system and transformed into a global coordinate. This type of concept could be applied both customized and commercial laser scanning system if the kinematic parameters are known. The schematic of the kinematics solution of the hybrid laser scanning platform used in this thesis is shown in Figure 13.

The coordinate system 0 (x_0, y_0, z_0) is fixed on the laser scanning system, which is the base reference frame for localization in the 3D environment. The local coordinate system 1 is located at the center of the laser scanner's body frame, which is rotating along the z_1 axis in terms of θ_1. The local coordinate systems 2 and 3 are fixed on the center of each of the two vertically-mounted LiDAR scanner. Each LiDAR can collect distance measurements from -95° to 95° which is named θ_3 with a 0.1667°-increment resolution. Finally, the local coordinate system 4 indicates a measured point (x_4, y_4, z_4) on an object surface. Based on the relationships identified, the kinematics problem is solved using Denavit–Hartenberg parameters, as shown in Table 2 and Equation 1. The red box in

Equation 1 is the point coordinate of a map location in terms of the body rotation angle θ_1 and laser scanner beam angle θ_3.

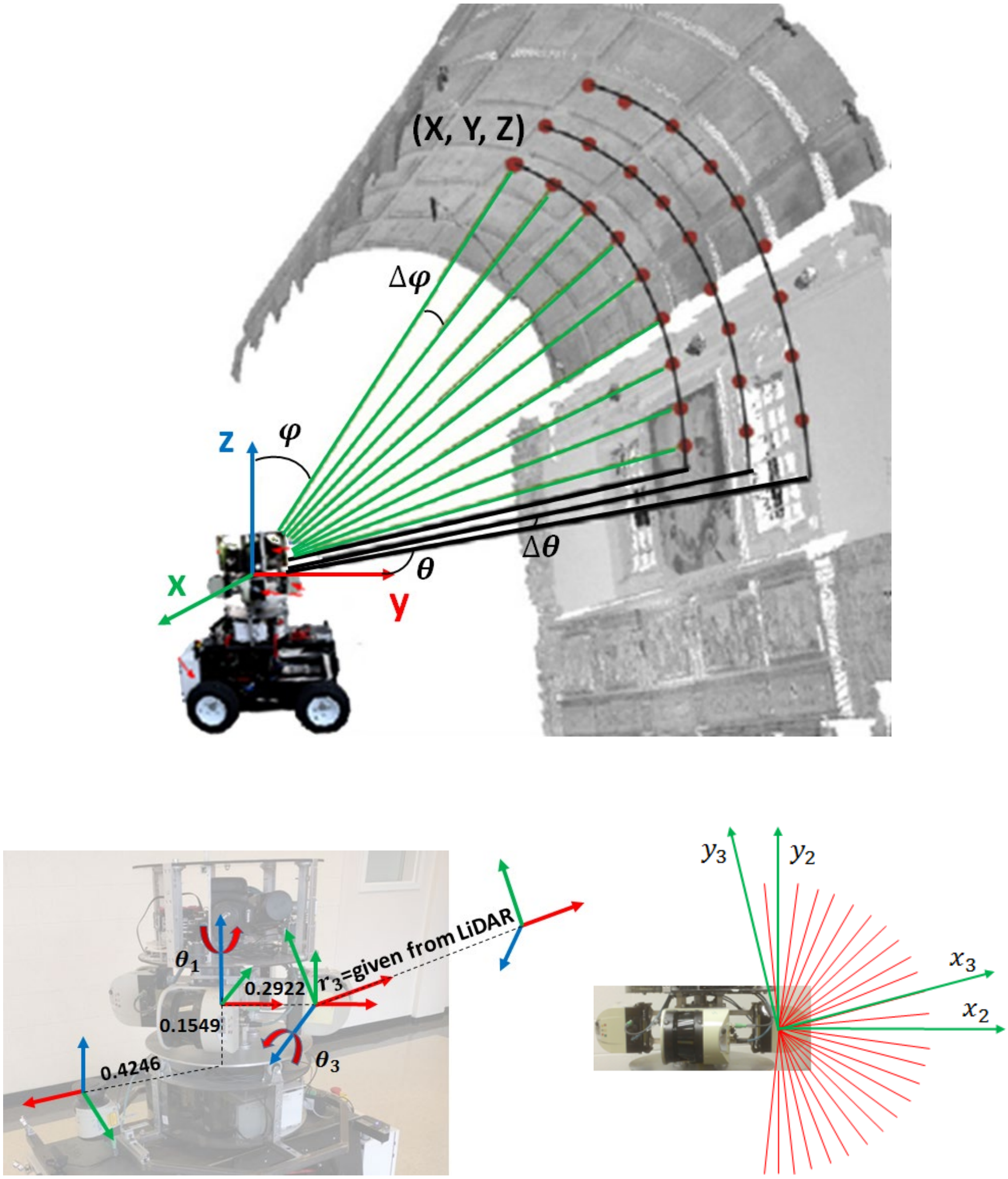


Figure 13 – The kinematics of the hybrid laser scanning system

Table 2 – D-H parameters for the hybrid laser scaning system

	θ	d	r	α
0	-	-	0.4246	0 °
1	θ_1	0.1549	0.2922	90 °
2	0 °	0	0	0 °
3	θ_3	0	r_3	0 °
4	0 °	0	-	-

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta_1)\cos(\theta_3) & -\cos(\theta_1)\sin(\theta_3) & \sin(\theta_1) & r_1\cos(\theta_1) \\ \sin(\theta_1)\cos(\theta_3) & -\sin(\theta_1)\sin(\theta_3) & -\cos(\theta_1) & r_1\sin(\theta_1) \\ \sin(\theta_3) & \cos(\theta_3) & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_3 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} \boxed{\begin{matrix} r_3\ \cos(\theta_1)\cos(\theta_3) + r_1\cos(\theta_1) \\ r_3\ \sin(\theta_1)\cos(\theta_3) + r_1\ \sin(\theta_1) \\ r_3\ \sin(\theta_3) + d_1 \end{matrix}} \\ 1 \end{bmatrix}$$

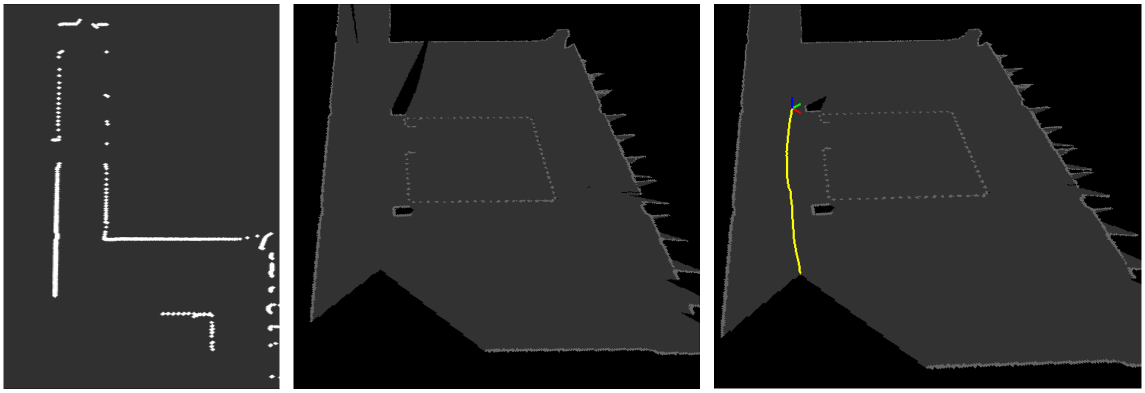
Equation 1

3.5 SLAM-based Point Cloud Registration

3.5.1 2D Pose Estimation with Hector Mapping

As described in Chapter 1, the SLAM algorithm is the process that builds the environmental map and estimates the current location at the same time, which is interdependence with each other. There are a lot of SLAM algorithms from the traditional wheel odometry-based filtering method to a camera or LiDAR-based advanced method. The target environment of this thesis is cluttered construction or disaster sites. Such environments have many possibilities to slip due to the ground condition so that the traditional wheel odometry-based filtering approaches are not appropriate in these environments because the slip is the main source of wheel odometry. The camera-based visual SLAM is the hottest topic in the robotics field. However, there is a critical limitation on visual SLAM that it is vulnerable the light conditions. If the light conditions differ with the same environment, visual SLAM could occur localization error. Therefore, visual SLAM works excellently in the indoor environment, however, the target environment of this thesis is outdoor.

Furthermore, the SLAM problem belongs to a difficult problem because the mapping should be performed while estimating the location and direction of the sensor continuously and simultaneously when the sensor is moving during mapping. Also, there are various challenging factors such as the time difference that sensor data comes in and the error while it estimates motion. Numerous LiDAR-based SLAM algorithms have been studied since the LIDAR sensor has a relatively small distance error compared to other sensors. It depends on the LiDAR device but the accuracy is approximately $\pm\boldsymbol{2.5cm}$ in 150 $\boldsymbol{m}$. The 2D LiDAR-based SLAM algorithm utilizes the horizontal LiDAR data to calculate the position and orientation of the mobile platform on 2D plane. The Hector SLAM algorithm, developed by Kohlbrecher et al. is to perform laser scan matching between the current LiDAR scan and the incrementally built map for pose estimation and to generate a horizontal map. Figure 14a demonstates the raw LiDAR data. After then, scan matching is performed using the current estimated map and the previously built map to get translation and rotation, as shown in Figure 14b. This will keep calculating the robot pose as the robot moves along the path as well as the 2D map, which is shown in Figure 14c**.**



(a) (b) (c)

Figure 14 – Scan matching with horizontal LiDAR: (a) raw LiDAR data, (b) 2D map, (c) trajectory of the mobile robot

The coordinate frames are visualized in Figure 15 to estimate 6D pose with use of Hector SLAM. Both "odom" and "map" frame is a fixed world frame. The base footprint frame does not provide height information and represents only the 2D pose of the robot in 2D plane. The base stabilized frame adds the robot height relative to the base footprint frame. The base laser link frame has origin at the horizontally-mounted LiDAR and adds the roll and pitch angles relative to the base stabilized frame. Therefore, the base stabilized and base laser link frames are equal if there is no roll/pitch motion which is calculated by IMU measurement.

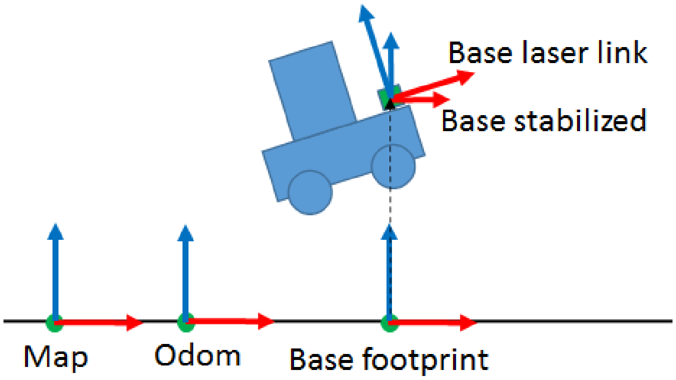


Figure 15 – The coordinate frames to estimate 6D pose

3.5.2 3D Motion Estimation with LiDAR Odometry

The problem of pose estimation with Hector SLAM has the possibility of some distortions when it moves because it is based on a rotating 2D laser scanner in spite of IMU compensation. This poses a major challenge for correct and accurate registration. Currently, most 3D reconstruction systems estimate the motion and reconstruct the environment through post-processing. They generally collect the data using the sensor and then process it on a powerful computer to obtain an accurate model of the environment. In contrast, the LiDAR Odometry And Mapping (LOAM) algorithm [49] to estimate and calculate the odometry and mapping with 6 degrees of freedom space $(x, y, z, \theta_x, \theta_y, \theta_z)$ in real-time with 3D LiDAR sensor data. It continuously estimates the motion of the sensor, removes distortion from the point cloud, and registers the local point cloud to obtain the whole map. The algorithm assumes that the movement of the sensor is continuous and smooth over time without abrupt changes, and a sweep is defined as one complete 360° scan. The right superscription k indicates the number of sweeps, and P^{k} indicates the point cloud perceived during sweep k.

The LOAM contains two major threads, "odometry" and "mapping," which run in parallel, as shown in Figure 16, which provides characteristics of low drift and low computational load. The main function of the "odometry" thread is to estimate the relative motion of the sensor between two sweeps by finding LiDAR feature points and their correspondences at a higher frame rate. It also removes distortion in the point cloud caused by the motion of the LiDAR using the estimated motion. It assumes that the angular and linear velocities of the sensor during a sweep are constant, which enables it to linearly interpolate the pose transform within a sweep for the points that are scanned at different times. The "mapping" thread takes the undistorted point cloud and incrementally builds a map. It simultaneously computes the pose of the LiDAR on the map using pose optimization at a lower frame rate, which is once per sweep. The estimation of the LiDAR state is a combination of the outputs from the two threads.

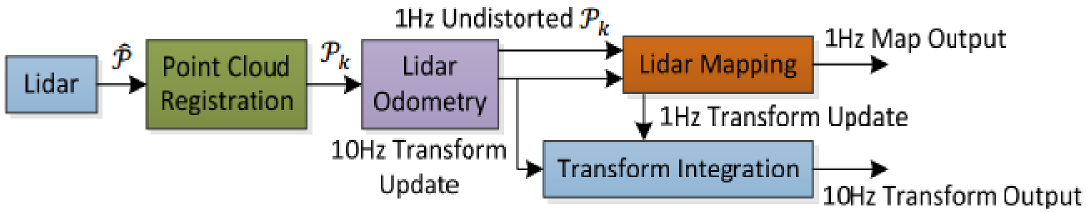


Figure 16 – Block diagram of the LiDAR odometry and mapping software [49]

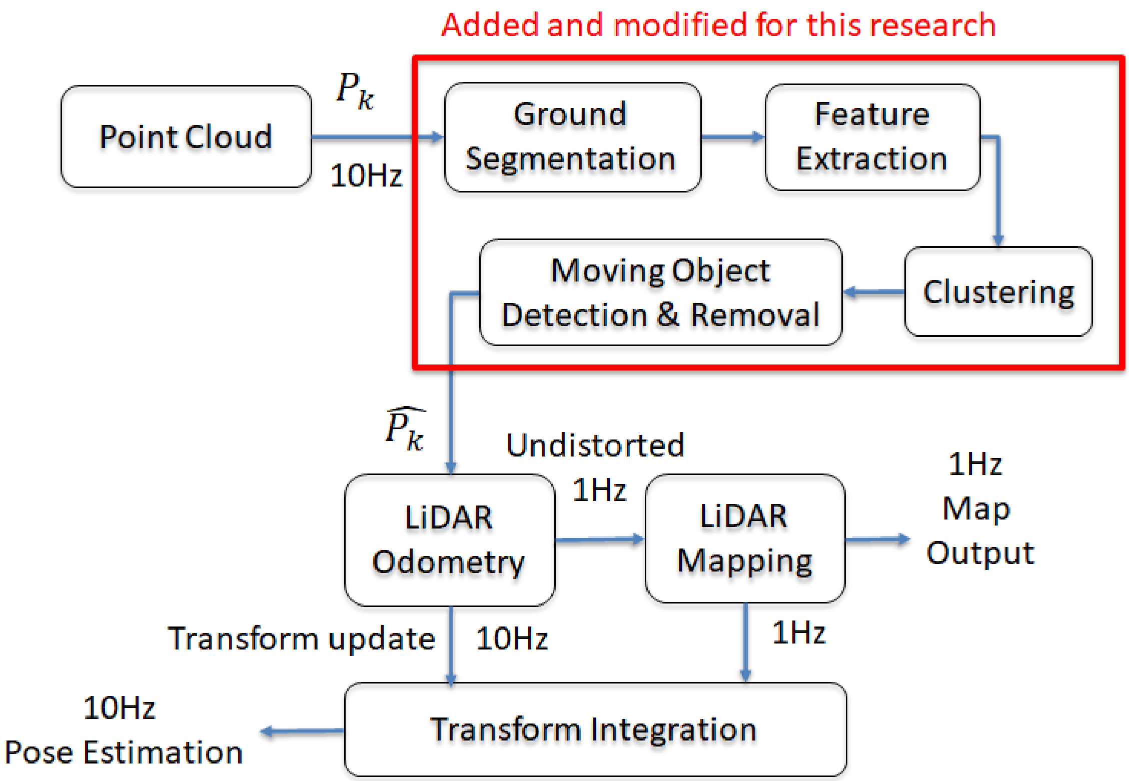


Figure 17 – Block diagram of the modified LOAM algorithm for this research

Figure 17 is the modified diagram of the LOAM algorithm used in this research. Among this diagram, the feature extraction modules revised and the ground segmentation, clustering, and moving object detection and removal parts are added. At first, the raw LiDAR data is collected and generated to point cloud. After the ground segmentation and feature extraction is done, it clusters point cloud into several objects. Then, the moving objects are detected and removed from the extracted features for better SLAM calculation. This is because the LOAM algorithm is to estimate odometry by calculating a relative pose between current and previous point cloud data. At the same time, it also computes the Lidar Mapping which is correcting the pose by matching the accumulated point cloud with the current point cloud. The detailed modification of algorithm will be described in the section 3.7.

3.6 Real-time Terrain Traversability Assessment

A LiDAR edge is a point in the point cloud that shows discontinuities or large changes. The laser scan data of flat ground without any objects should be in the form concentric circles on the ground plane. This is because each circle is generated by the ranges detected by the complete rotation of a single specified angle laser. This formation means that there are no LiDAR edges and no discontinuities or large changes in the point cloud. However, the data would deviate from a circular pattern if there were walls or objects around the laser scanning system, as shown in Figure 18. If an object was present, the range data would be less than that expected on flat ground. In contrast, the range data would be greater than that expected on flat ground if a hole was present. This demonstrates that the vertical and horizontal geometry of the scan data can be utilized to determine traversable regions and obstacles.

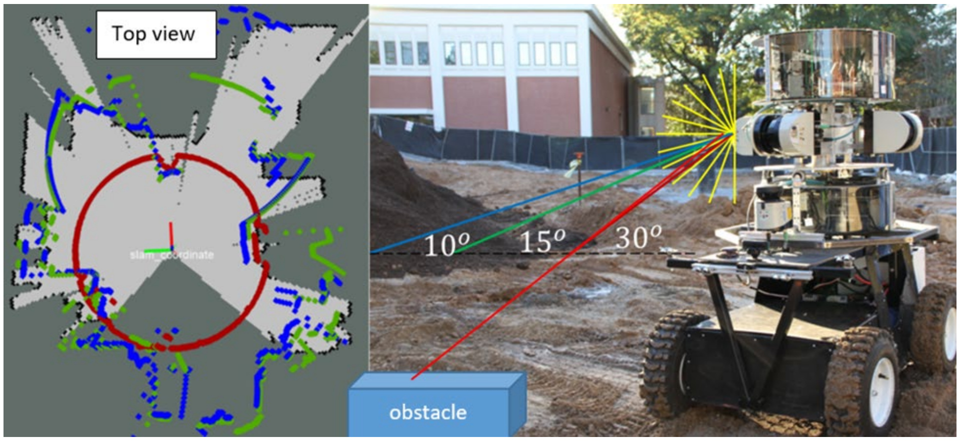


Figure 18 – Detecting obstacles using vertical laser lines

The customized laser scanning system collects 3D range data while it is rotating about an axis. While the system is rotating, it samples the surrounding environment vertically by firing lasers at several elevation angles (θ). At the same time, it collects data at a discrete azimuth angle (φ) horizontally. This approach is dependent on the difference in ranges and the angle between the neighboring laser beams and not on the azimuth angle φ. Figure 9 depicts the laser scanner system and its laser firings at a specific azimuth φ when lasers sample the flat ground at various points away from the sensor position for several elevation angles θ along a radial direction. In this paper, the measured range data is described as $R_{\theta,\varphi}$, where θ is the vertical index along the elevation angle and φ is the horizontal index along the azimuth angle. The raw scan data is also pre-processed by running a simple median filter with a three-point window in vertical to smooth the data.

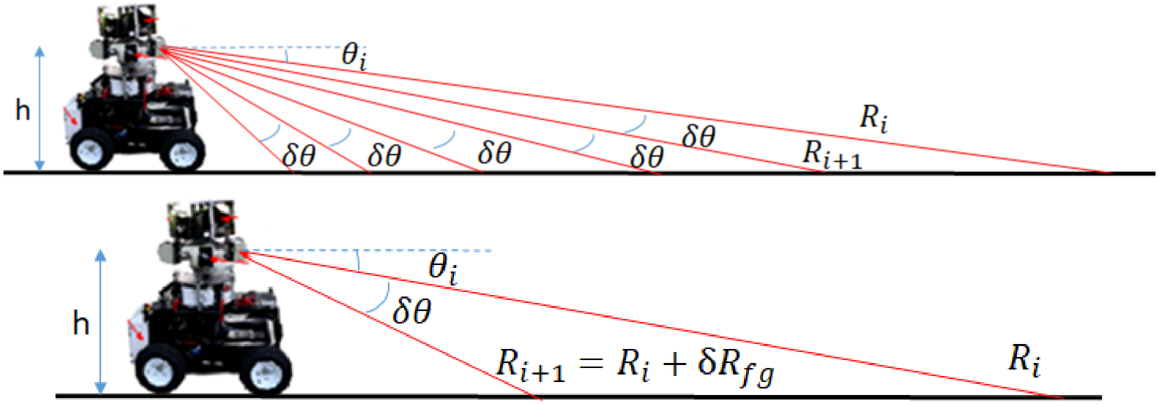


Figure 19 – Illustration of single LiDAR firings to the flat ground

The range difference between a laser at elevation angle θ and a neighboring laser at elevation angle θ+δθ is represented by δR. If the ground in the surrounding environment is perfectly flat, there is an expected δR for the elevation angles θ and θ+δθ, as described in Equation 2 and Figure 19.

$$h = R_i \sin\theta_i = (R_i + \delta R_{fg}) \sin(\theta_i + \delta\theta)$$

Equation 2

$$\delta R_{fg} = R_i \left(\frac{\sin\theta_i}{\sin(\theta_i + \delta\theta)} - 1 \right)$$

Since $\delta\theta$ is positive and $\sin\theta_i < \sin(\theta_i + \delta\theta)$, the expected range difference for flat ground δR_{fg} should be a negative value. On the other hand, δR_m can be defined as the range difference between neighbor points in Equation 3.

$$\delta R_m = R_{\theta_{i+1},\varphi_j} - R_{\theta_i,\varphi_j}$$

Equation 3

The value of δR_{fg} can vary from small to large. For example, if θ is small, then δR_{fg} will have a very large value. In contrast, if θ is large, then δR_{fg} will have a relatively small value. This is visualized in Figure 19. Therefore, it is necessary to use a ratio between the measured range difference and the expected range difference for flat ground, as shown in Equation 4.

$$\rho = 1 - \frac{\delta R_m}{\delta R_{fg}}$$

Equation 4

This represents the unevenness of the nearby terrain. If the value of ρ is close to zero, this means that the points are nearly flat. On the other hand, the further away from zero the value, the greater the degree of deviation from unevenness. Figure 20 shows four examples of ground shape calculation.

Using this geometric characteristic, it is possible to detect object locations from the mobile platform, which is helpful for navigation. However, it is also necessary to compute the slope of the ground to determine its traversability, as in Equation 5. Slopes are traversable if they are within the navigational capability of the robot; so a terrain with a slope of less than a given threshold is traversable [8,50]. Here is an example of this approach in Figure 21 which shows an example point cloud of the ground and wall.

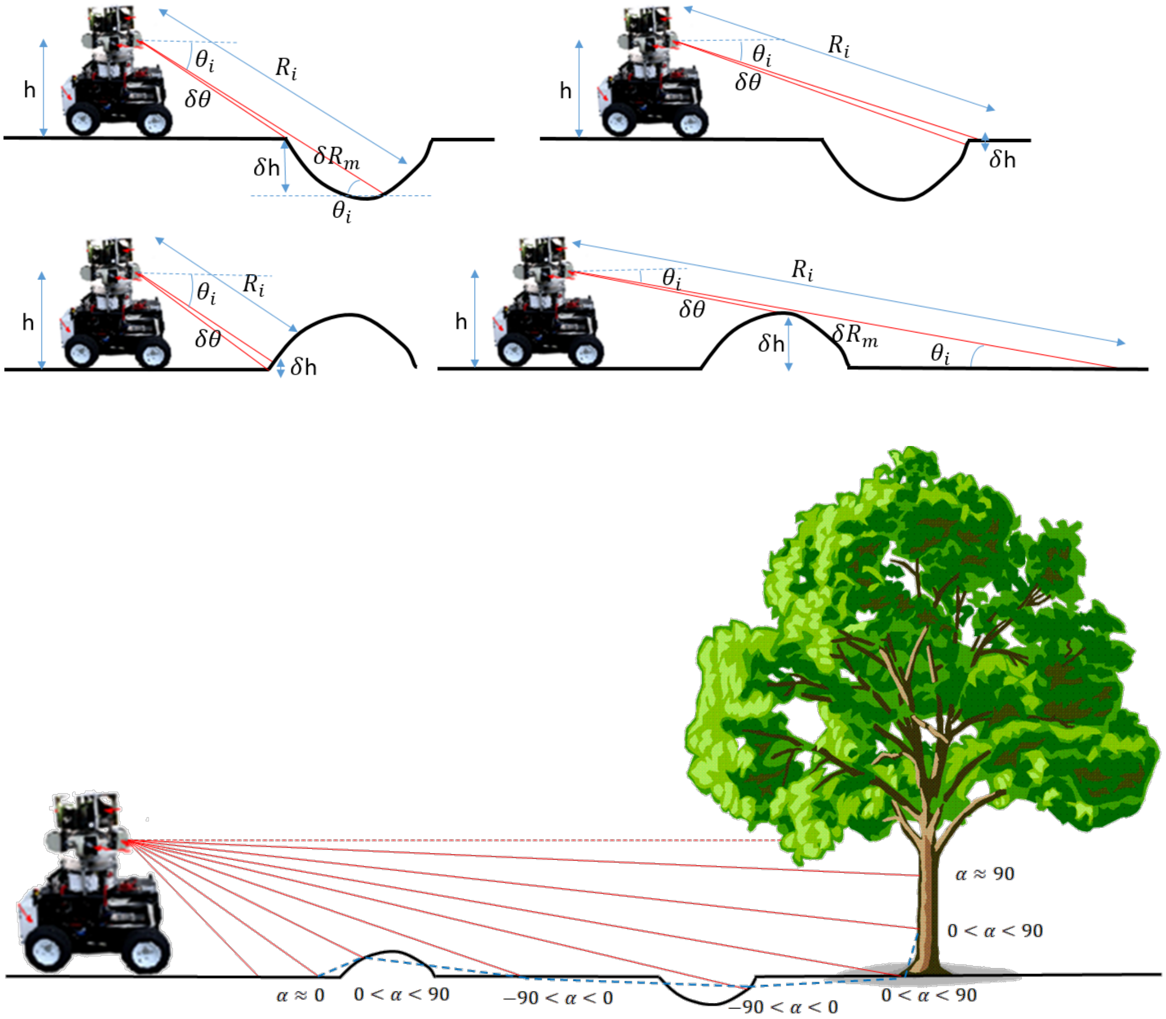
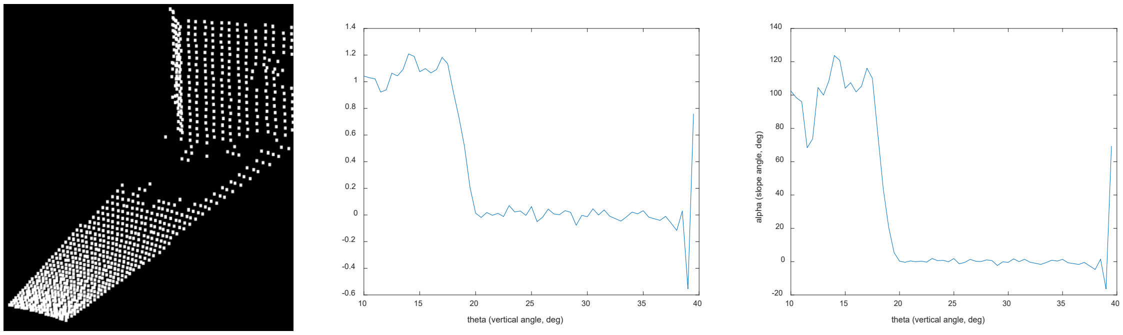


Figure 20 – Illustration of single LiDAR firings to the non-flat ground

$$slope = \frac{(h - R_i\, sin\, \theta_i) - (h - R_{i+1}\, sin\, \theta_{i+1})}{R_i\, cos\, \theta_i - R_{i+1}\, cos\, \theta_{i+1}}$$

$$= \frac{R_{i+1}\, sin\, \theta_{i+1} - R_i\, sin\, \theta_i}{R_i\, cos\, \theta_i - R_{i+1}\, cos\, \theta_{i+1}} = \tan(\alpha) \qquad \text{Equation 5}$$

$$\alpha = \tan^{-1}\left(\frac{R_{i+1}\, sin\, \theta_{i+1} - R_i\, sin\, \theta_i}{R_i\, cos\, \theta_i - R_{i+1}\, cos\, \theta_{i+1}}\right)$$



(a) (b) (c)

Figure 21 – (a) Example point cloud, (b) unevenness and elevation angle, (c) slope angle and elevation angle

The proposed method performs obstacle detection at the laser scanning point level by classifying points as belonging to the ground or to an obstacle. It performs real-time classification along a LiDAR measurement angle. It identifies the traversable region without using normal classification, plane fitting, Gaussian Regression, or PCA, which are computationally expensive or require training for classification. The fact that the traversable region can be obtained solely by the geometry and ranges of a rotating laser scanner makes this method simple and effective. The aim is not to identify or recognize obstacles but simply to detect objects and identify the traversable region for the mobile robot.

3.7 Moving Objects Detection and Removal from SLAM calculation

The approach used in this section detects and removes moving objects from the incoming point cloud frames in real-time which can later be used by any available SLAM algorithm to improve localization accuracy. The main idea is to observe relative motion between the LiDAR and the objects within its surroundings. The data far from the LiDAR does not required to detect nearby moving objects because it usually has noisy data and inaccurate. Therefore, the LiDAR raw data could be cropped to a specific range of data around the LiDAR. With the cropped point cloud considered as input for moving object detection and removal, the process has been proposed as shown in Figure 22.

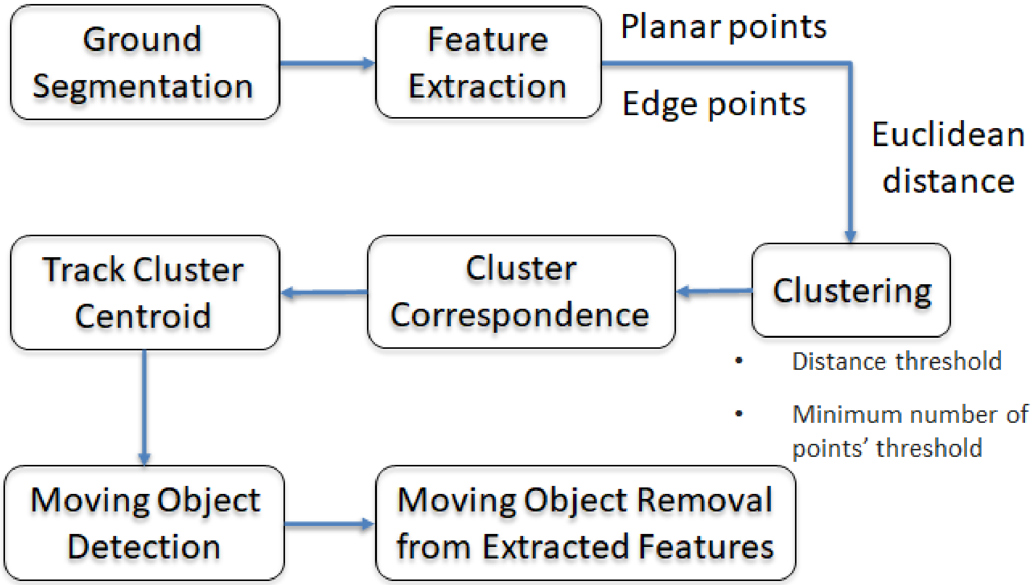


Figure 22 – Block diagram of the moving object removal process

Figure 23 demonstates an example of a point cloud scanned in an indoor environment and the process of moving objects removal. The captured data are shown after the 10×10 m^2 cropping around the sensor. There are several moving objects in the consecutively captured point cloud. As shown in the figure, the points corresponding to the ground become connecting links between each object observed in the point cloud. Due to this reason, clustering the points into individual groups of point clusters becomes difficult. If there exists ground plane points, the Euclidean clustering cannot be used. This is because the multiple independent objects could be grouped as a single cluster due to their connectivity through the ground plane points. The color of a given points in Figure 23 represents the height of laser point cloud.

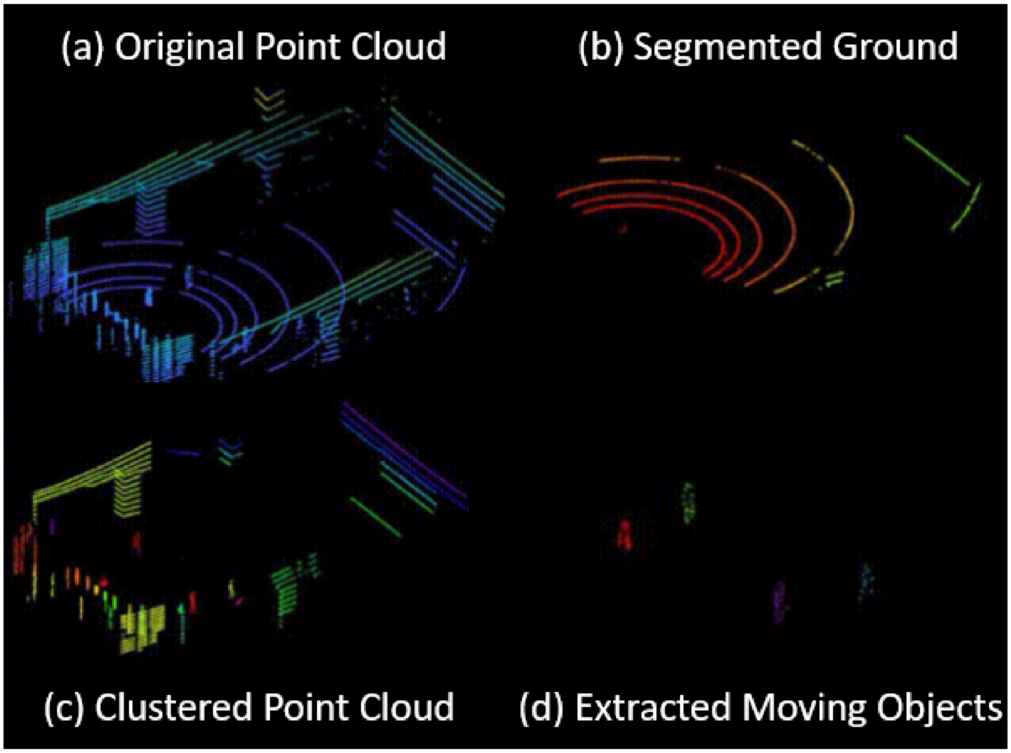


Figure 23 – Illustration of the moving object removal process

The well-known ground plane removal methods is the RANSAC algorithm, which is an iterative plane fitting procedure that randomly selects points from the point cloud to fit a 2D plane according to a provided threshold. However, the conventional RANSAC algorithm frequently fails in extracting the ground plane satisfactorily from cluttered environments. Therefore, the ground segmentation is achieved by the method by using calculating the unevenness of terrain in section 3.6, as shown in Figure 23 (b). Then, the features are extracted by computing curvature using Equation 6, which is an existing method in the LOAM algorithm. If the curvature is larger than threshold, it designates as edge feature. The extracted edge feature points without ground points are clustered with respect to the Euclidean distance between them. This process separates the point cloud into clusters by using the minimum distance threshold between two points to be considered belonging to the same cluster and minimum number of point's threshold within a cluster, as shown in Figure 23 (c).

$$C = \frac{1}{|S|\left\|X_{(k,i)}^{L}\right\|}\left\|\sum_{j\in S, j\neq i}\left(X_{(k,i)}^{L} - X_{(k,j)}^{L}\right)\right\| \qquad \text{Equation 6}$$

Once the point cloud is segmented for several clusters, the cluster correspondence is performed, which is the task of finding relationship between clusters. In other words, every pair of consecutive point cloud frame has an associated map M_t that saves the cluster correspondence relations between the clusters of the two consecutive frames. An example of such cluster correspondence is shown in the Equation 7.

$$M_t = ([2 \rightarrow 5]\ [3 \rightarrow 7] \dots\ [p \rightarrow q])$$

$$1 \leq p \leq k_{t-1}; 1 \leq q \leq k_t$$

Equation 7

$$(k_t\ is\ the\ number\ of\ clusters\ in\ point\ cloud\ at\ time\ t)$$

A volume constraint is implemented with the bounding box (BB) approach to deal with the false positive correspondence. The two matched clusters should have similar volume of its BB. This condition must hold true because the time frame between two consecutive point cloud is around 0.1 sec based on the LiDAR specification and the viewpoint does not change drastically due to the less mobile robot speed. Equation 8 shows the the normalized volume constraint and it can be utilized for removing outliers such as false positive from the map M_t. Let assume that a cluster captured at time t is represented as PC_t^k and the identical cluster k is represented as PC_{t+1}^k which is captured at time t+1.

$$v_1 = Volume\left(BB\left(PC_t^k\right)\right)$$

$$v_2 = Volume\left(BB\left(PC_{t+1}^k\right)\right)$$

Equation 8

$$\frac{|v_1 - v_2|}{|v_1 + v_2|} < \delta$$

A moving object can be described that the pose changes constantly with respect to the inertial frame. The centroid of each cluster is extracted and tracked the location utilizing relative pose of the sensor about the objects. If the location of centroid has certain direction and speed for several consecutive point cloud, it can be detected as a moving object as shown in Figure 23 (d). Finally, the points identified as a moving object are removed from the extracted features which will be used for odometry calculation.

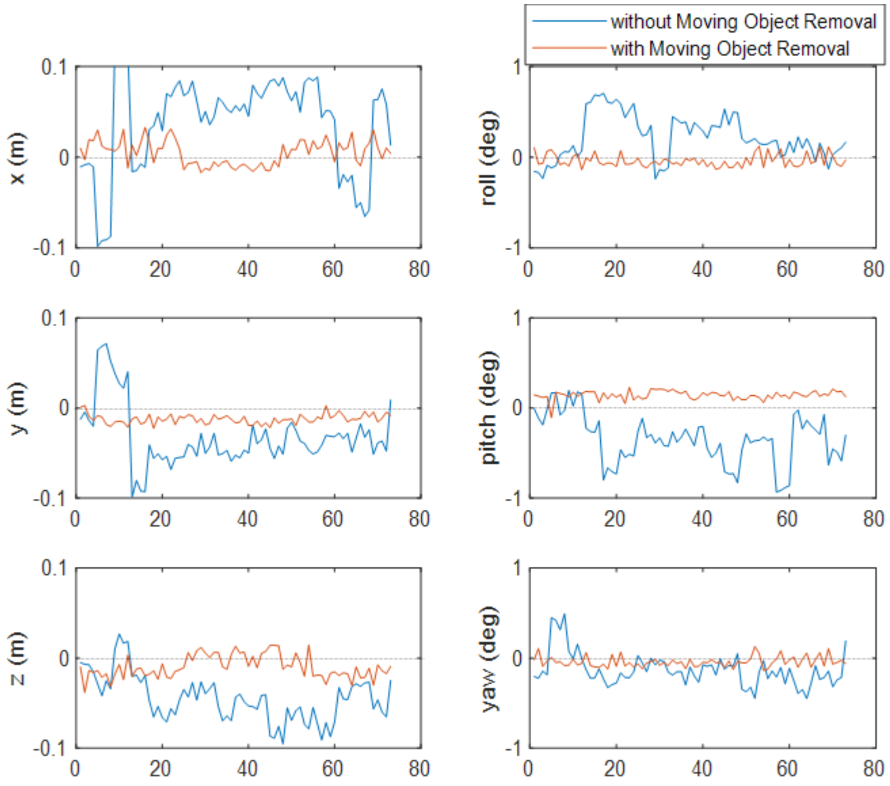


Figure 24 – Effect of moving object removal

Figure 24 and Table 3 provides to verify the effect of moving object removal on the performance of SLAM odometry estimation. As shown in Figure 24, even though the sensor did not move, the estimated odometry can fluctuate because the calculation is performed using extracted feature points and does not consider movement. In other words, if there are many moving objects near the sensor, this will lead to many false feature points, which create inconsistencies in the odometry calculation and can be a source of error. As shown in Figure 24 and Table 3, the odometry estimation varied by 5 cm in position and oscillated by less than 0.5 degrees in orientation using the original LOAM calculation. However, this was reduced to a 1.5 cm variation in position and 0.1 degree variation in orientation when moving object removal was applied. In short, the variation in the odometry estimation when moving object removal was applied was reduced to 28.6% that of the original algorithm. Therefore, removing moving objects from the point cloud yielded odometry results that were 3.5 times more accurate.

Table 3 – Effect of moving object removal

	RMSE	(1) w/o moving object removal	(2) w moving object removal	Variation (%) = (2)/(1)
Position	X (m)	0.0660	0.0144	21.8
	Y (m)	0.0471	0.0134	28.5
	Z (m)	0.0517	0.0159	30.8
	Avg.	0.0549	0.0146	26.6
Orientation	Roll (deg)	0.3285	0.0792	24.1
	Pitch (deg)	0.4420	0.1509	34.1
	Yaw (deg)	0.2265	0.0726	32.1
	Avg.	0.3323	0.1009	30.4

3.8 Path Planning and Tracking Control

Path planning means determining an collision-free path to reach the goal position. There are three prime objectives for inspection using ground robots: (1) minimum time of operation, (2) collision avoidance, and (3) maximum coverage. Shortening the path length plays is the most efficient way of reducing the operation time and can be achieved by the global planner. The global planner is also used to avoid static obstacles, but dynamic obstacles require a local planner. The block diagram of the path planning algorithm including global and local planners is shown in Figure 25 how to relate to each other. Lastly, the maximum coverage is computed by the scan view planning module, which is described in section 3.9.

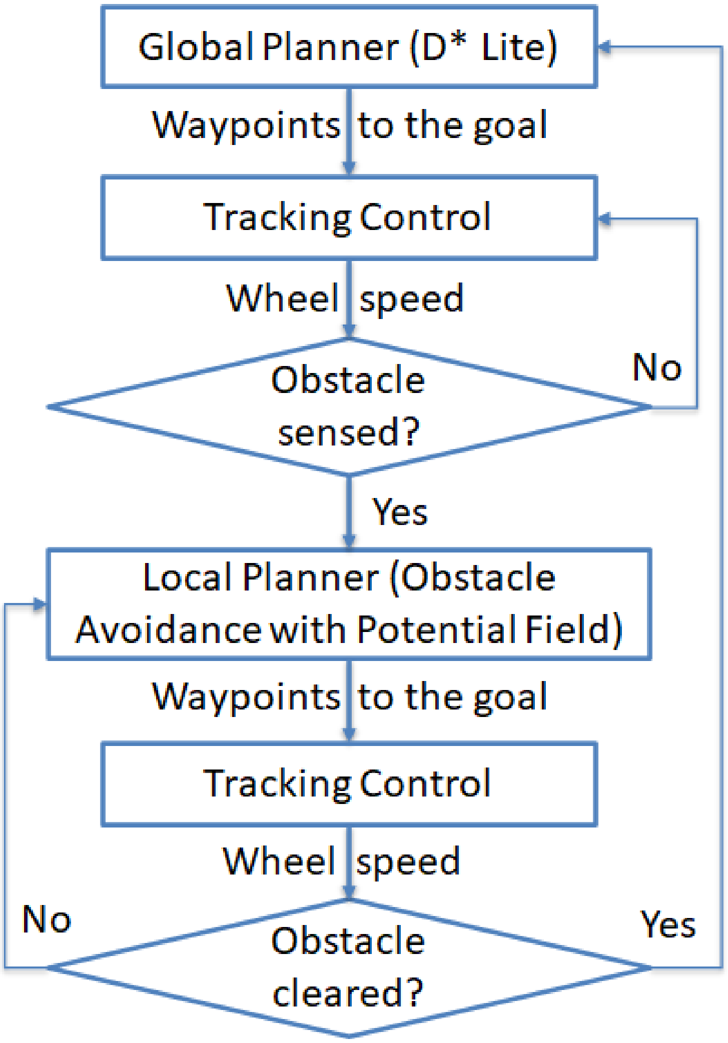


Figure 25 – Block diagram of path planning algorithm

3.8.1 Global Planner

For the global path planner to perform, the goal position must be selected. The proposed system uses the frontier-based approach to determine the exploration goal location. When the GRoMI begins scanning, it first detects objects around it. Then, the local goal position is determined using the frontier algorithm [51] within the range of the currently built map area. Instead of choosing the nearest frontier, the global planner considers both distance and heading angle to select a goal position within the un-scanned area shown in Figure 26 using Equation 9.

$$Goal = \underset{i}{\operatorname{argmin}}(|\theta - \beta_i| \times d_i) \qquad \text{Equation 9}$$

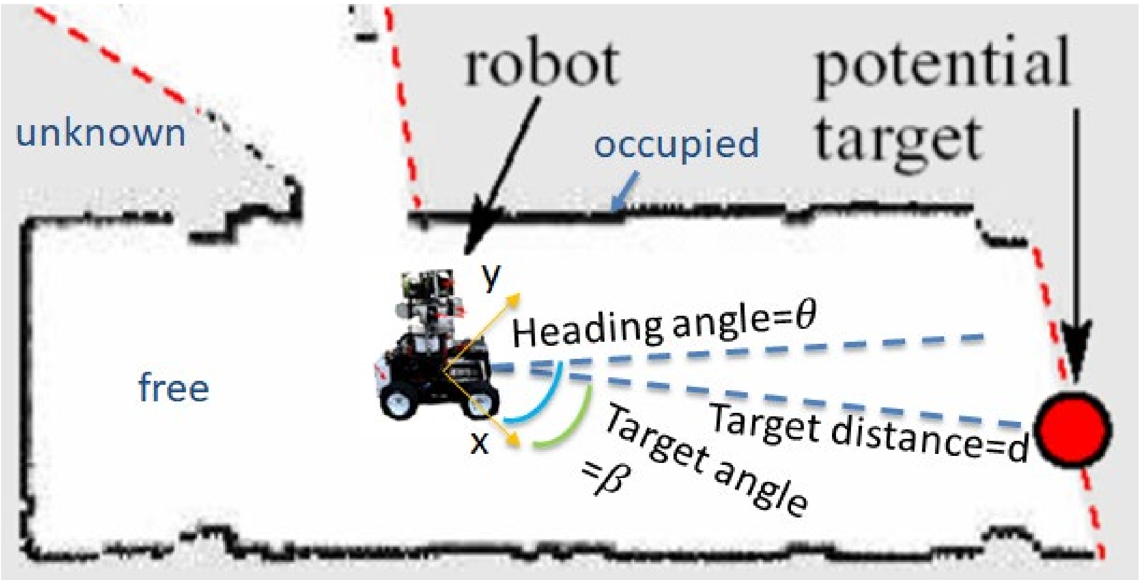


Figure 26 – Example of selecting the global planner goal position

The most advanced existing methods for mobile robot path planning are probabilistic approaches such as RRT, RRT*, or variants thereof [52]. The general approach using

RRT* is to plan in higher-dimensional state space; the planner randomly samples states and either specifically calculates or randomly guesses actions that will take the robot from the current state to the target state. The success of these approaches depends largely on the accuracy of the modeling. If the higher-dimensional state-space model is created, the success rate of planning will be increased. However, the random sampling required to expand the states leads to high computational cost, which makes the RRT-based approach unsuitable for real-time planning in dynamic environments.

Since this study focuses on path planning in an unknown environment, D* Lite is a more appropriate global planner as it has efficient re-planning capabilities that enable it to handle changes in the environment [53]. D* Lite is more computationally efficient than A* searches and easier to implement than the original D* [54]. D*Lite works in the XY plane which is a grid-based planner and the Euclidean distance from the goal position is used as a heuristic. To reduce complexity, it uses a simple model that can move from the current cell to eight adjacent cells as long as there are no obstacles in the adjacent cells. The global planner is initialized with a uniformly discrete 2D occupied grid model of the environment with all unknown edge costs set to a minimum. This enables the planner to estimate an initial guess for the path to the goal without a full map. It is called the robot navigation map (shown in Figure 27) and is based on the unevenness value and slope angle described in section 3.6. First, the currently built point cloud is divided into cells. The cell size must be chosen with care because it is related to computational load and resolution. If the cells are too large, the map may not be encoded properly due to low resolution. The smaller the scale of the cells, the more accurate the voxel map but the larger the memory footprint. For example, in this study, the GRoMI has a footprint of approximately one square meter.

Therefore, it might be assumed that selecting a scale of one meter per cell would be correct. However, while this is a fine starting place, it is less than ideal. If we decide the robot needs a cell plus adjacent cells available to drive through, then we require three traversable cells. Very few doorways are three meters wide, so our robot would be unable to pass, even though in reality it would easily fit. Therefore, a scale of 0.5 meters by 0.5 meters was used as this created a sufficiently detailed map but allowed for the somewhat tight passage of the robot.

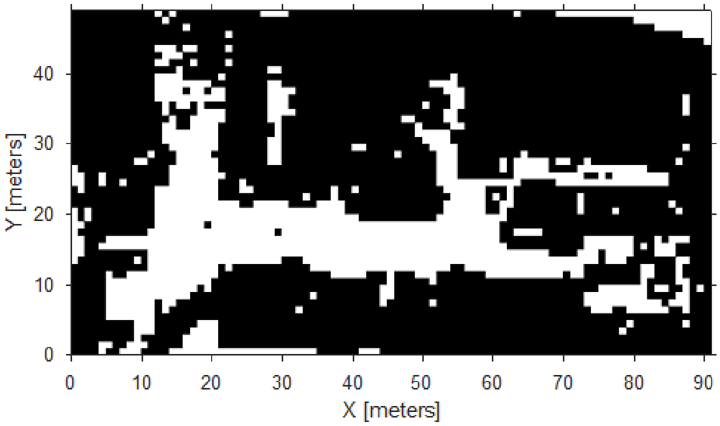


Figure 27 – Robot navigation map (white: traversable, black: non-traversable)

3.8.2 Local Planner

By assuming that all the obstacles are static, the global planner can take into account potential collisions while the robot moves to the goal without considering the actions of the robot in the velocity space or adding the dimension of time to the planning space. However, a cluttered environment may contain unexpected dynamic obstacles. By replacing the global planner with a local planner and estimating the velocity of moving obstacles, the proposed approach can handle moving obstacles efficiently. In this thesis, the object-detection-based potential vector field method [55] is adopted as the local planner. The concept is that obstacles generate an artificial repulsive force and a goal position generates an artificial attractive force. The direction in which the mobile platform moves is determined by the direction of total field intensity, which is the sum of the artificial forces. Dynamic obstacles are detected while the ground robot is moving, and a potential field is constructed to avoid them. The ground robot then will follow the gradient direction of the potential field and requests re-planning from the global planner once it clears the obstacle. Examples of paths estimated using an artificial potential field from a start point (0,0) to a goal point (5,5) are indicated by red lines in Figure 28.

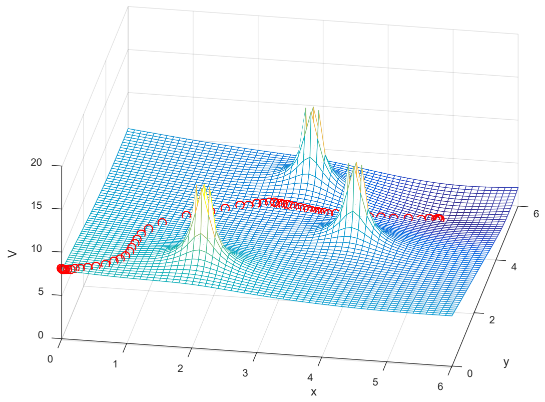


Figure 28 – Example of an artificial potential field and the trajectory

The simulated data were used to evaluate the artificial pre-gastric method. As shown in Figure 29, the laser line was simulated with a yellow line detecting obstacles. In this simulation, moving obstacles were used to demonstrate the robustness of this method. The black circle is the original position of the obstacle, and the red circle is the current position of the obstacle moving over time. The sky blue circle near the red circle, which is the mobile robot's current position, indicates the expected position of the obstacle detected by LiDAR. Finally, the light red circle represents the mobile robot's current position, and the red line represents the mobile robot's optimal trajectory calculated from the pre-artificial camouflage. The contour line shows the potential level with a column barrier obstruction at every point in the field and a local minimum at the target location.

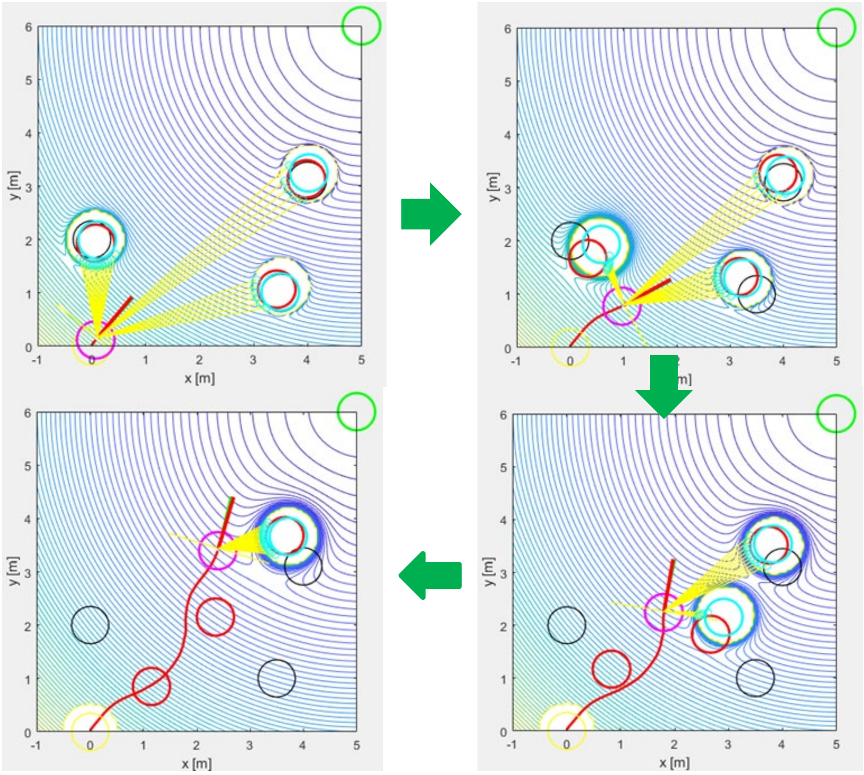


Figure 29 – Simulation of the artificial potential field and odometry model

3.8.3 Path Tracking Control

In order to accurately model the motion of the physical robot, the planner must also take into consideration the low-level controller that drives the physical robot [56]. The tracking controller is essentially a PD controller that drives the robot with generating the motor commands for the left and right wheels of a robot to the nearest waypoint as provided by the global planner. In order to drive the robot from current location (x,y) to a goal location $(\mathrm{x_g}, \mathrm{y_g})$, the robot has to first align towards the goal and then drive forward till it reaches a location within the threshold distance to the goal point. A schematic showing the status of the robot, goal and the control inputs given to the robot is given in Figure 30.

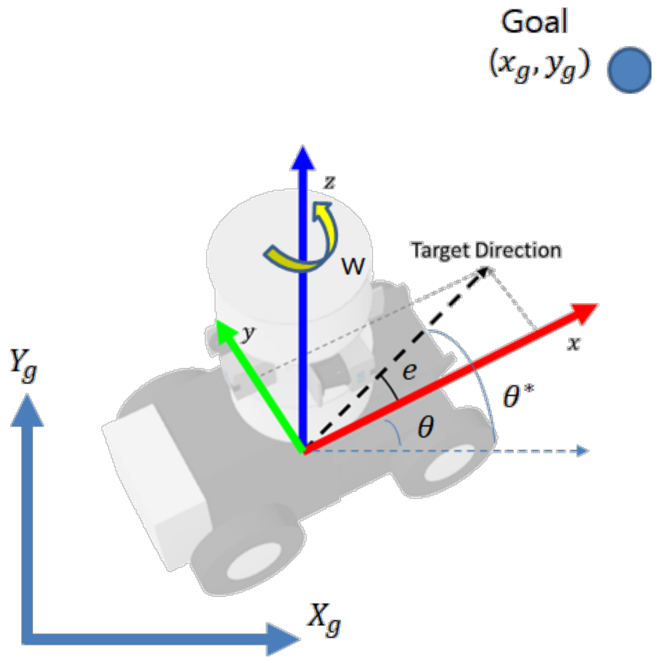


Figure 30 – The input of the path tracking controller

The slope of the line connecting the robot and the goal position, θ*, is the desired orientation of the robot. Based on the heading error the angular velocity control input (ω) to the robot can be determined from the equations given below:

$$\theta^* = \tan^{-1}\left(\frac{y_g - y}{x_g - x}\right)$$

$$e = \theta^* - \theta \qquad \text{Equation 10}$$

$$w = \mathrm{k_p} e + k_d \dot{e}$$

where, $\mathrm{k_p}$, and $\mathrm{k_d}$ are the proportional, and derivative gains respectively, e is the error limited between [–π, π).

The linear velocity of the robot is scaled non-linearly based on the angular velocity as given by:

$$V_{linear} = V_{max}\left(1 - \frac{2\tan^{-1}(|w|)}{\pi}\right) \qquad \text{Equation 11}$$

This allows the robot to slow down before taking turns and the velocities of the left and right wheels are generated as follows:

$$V_{left} = V_{linear} - \beta\theta$$

$$\text{Equation 12}$$

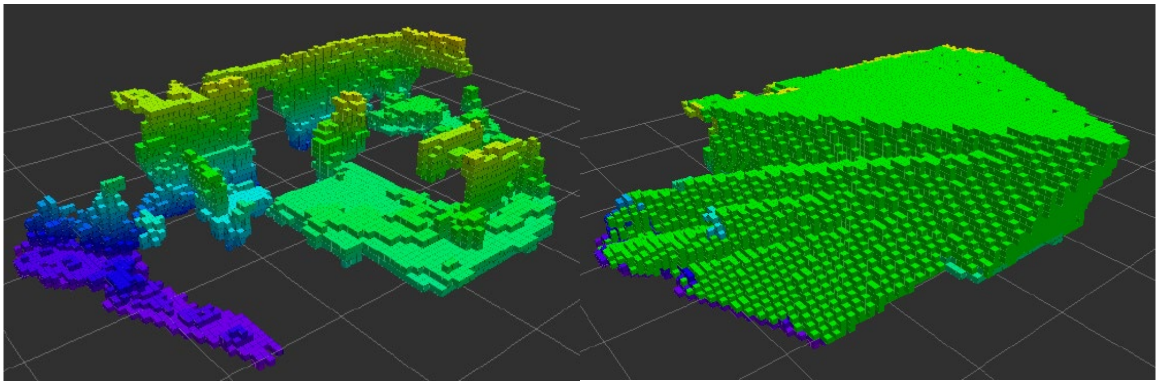
$$V_{right} = V_{linear} + \beta\theta$$

3.9 Scan View Planning

3.9.1 Frontier-Void-based Exploration

One problem related to exploration in 3D is NBV computation, which is the computation of a sequence of viewpoints that continues until an entire scene has been observed by a sensor. This thesis begins with the frontier-void-based exploration approach to exploration in 3D introduced by Dornhege and Kleiner [57]. This approach was chosen as it extends the widely used 2D frontier-based exploration algorithms into three dimensions and can deal with the kinematic constraints of the robot.

The algorithm is built on top of an occupancy grid, which is a map consisting of a regular grid of cube cells (voxels) that can be classified into three disjunct categories: occupied voxels, which contain points from the point cloud; free voxels, which have been covered by the sensor's field of view and do not contain any points (obstacles); and unknown voxels, which have not been covered by the sensor's field of view. A visualization of occupied and free voxels is presented in Figure 31 (a) and (b), respectively. Voxels that are neither occupied nor free are unknown and are not shown.



(a) Occupied voxels

(b) Free voxels

Figure 31 – Visualization of voxels

Free and unknown voxels are further differentiated. In accordance with Yamauchi [41], frontier voxels are defined as free voxels that are neighbors of unknown voxels. Void voxels are unknown voxels that are neighbors of occupied voxels. In this study, two voxels are considered neighbors if and only if they share a common corner, which is following the 26-connected of the definition of pixel connectivity.

The idea behind the frontier-void-based exploration approach is to estimate the exploration potential, that is, the utility, of free voxels in the known map and to use this utility to find the next best scan position. The method proceeds as follows:

1. Void voxels are extracted and clustered,

2. frontier voxels are extracted and clustered,

3. a set of frontier-voids is created that binds together frontier clusters with void clusters, and

4. the set of frontier-voids is used to determine the best location for the next view.

The frontiers are detected during the map update process and stored. Each time the dynamic point cloud is updated and integrated into the map, the voxels are also updated. Voxels are considered frontiers if their occupancy probability is lower than 0.5 and at least one of their 26 direct neighbor voxels has an occupancy probability of exactly 0.5 (which indicates an unknown or void voxel). Put more intuitively, frontier voxels are free voxels located next to unobserved voxels. The sorted frontier list is updated as new frontier voxels emerge and previously unobserved regions are observed. This frontier update is performed continuously at each LiDAR measurement integration.

3.9.2 Determining the Next Best Scan Location

The optimal position for the mobile robot to stop in 3D and acquire a high-resolution stationary scan is determined using the voxel map and visibility analysis. The ray-tracing algorithm [58] is used to calculate the line-of-sight visibility. First, point cloud of the target area is divided into voxels, which are small cubes utilized to represent 3D space, as shown in bottom right of Figure 32. Figure 32 demonstrates the process of generating 3D voxel and 2D robot navigation maps from a dynamic increment point cloud. The occupied voxel is then generating if there are points in a point cloud in the voxel area. Each voxel is allocated one of two statuses, visible or invisible, based on the ray-tracing result.

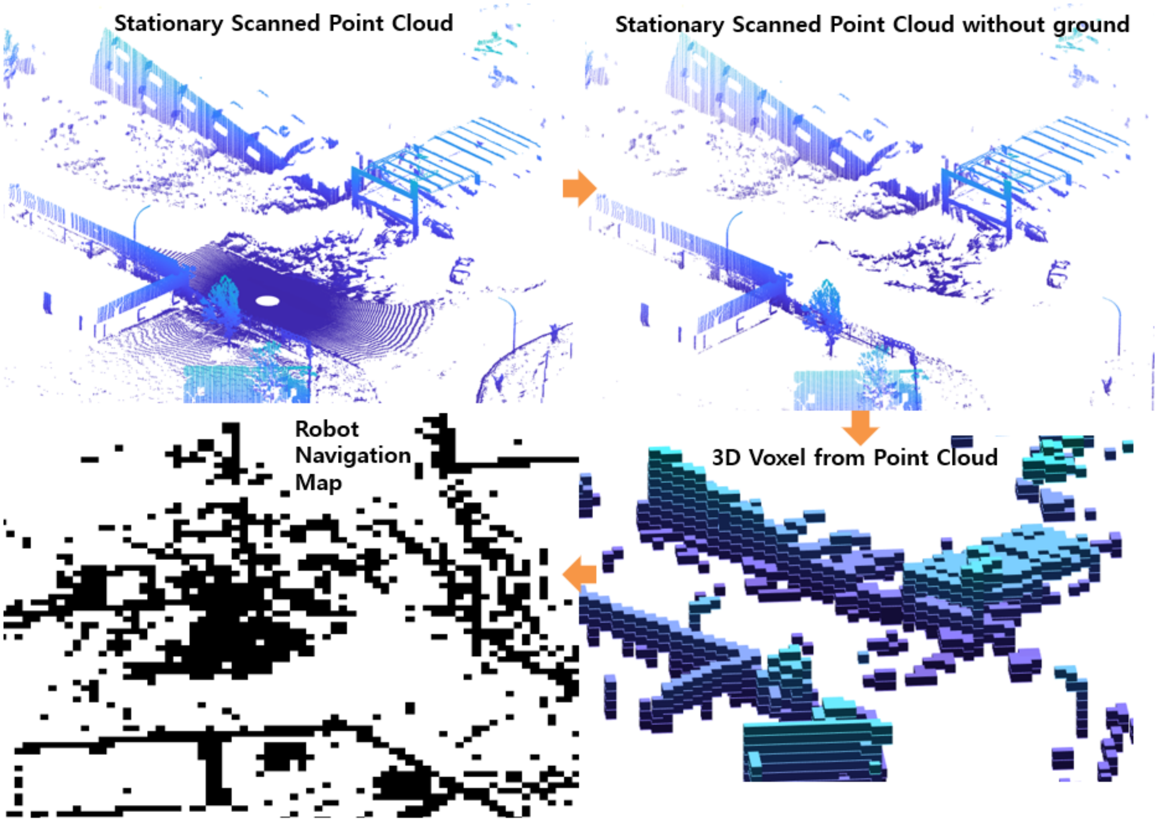

Stationary Scanned Point Cloud
Stationary Scanned Point Cloud without ground
Robot Navigation Map
3D Voxel from Point Cloud

Figure 32 – Process of generating 3D voxel and 2D navigation map from point cloud

To estimate whether a ray intersects with a voxel or not, let assume that the voxels are small spheres with a radius of $\mathrm{r} = \frac{\mathrm{a}}{\sqrt{2}}$, where a is the length of voxel's side [59]. The ray intersects the voxel if its distance from the center of the voxel is less than r. In Figure 33, θ is the angle between the ray touching the edge of the sphere of voxel A and the ray crossing the center of the sphere of voxel A, and φ is the angle between the ray crossing the center of the sphere of voxel A and the ray crossing the center of the sphere of voxel B. If θ > φ, voxel B is occluded by voxel A; otherwise, voxel B is visible.

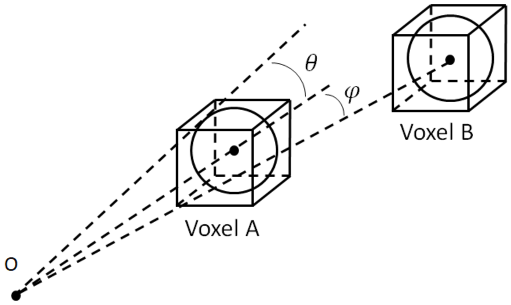
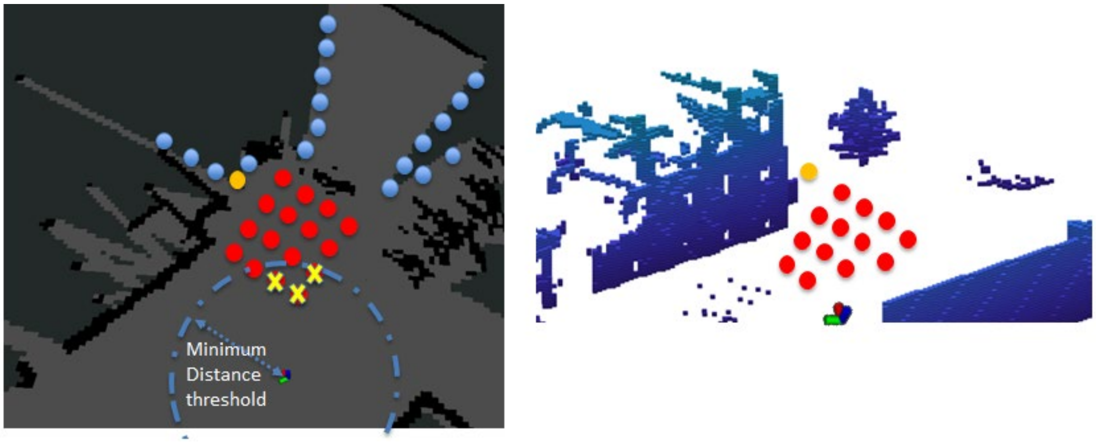


Figure 33 – Example of an occluded voxel



(a) Scan location selection in 2D view (b) Scan location selection in 3D view

Figure 34 – Procedure of scan location selection

The stationary scan locations should be reachable for the ground robot, have a large field-of-view of the surroundings with minimal obstructions, and have minimal overlapping areas between scans to reduce redundancy. To sum up, the scan location selection process should meet the following steps.

1) The traversable area displayed by the gray area in Figure 34 (a) should be identified, which was described in section 3.6 and 3.7.

2) The frontier cell locations which are displayed with the blue dots in Figure 34 (a) should be extracted and determine the temporary navigation goal location which is displayed with the orange color dot by using the frontier-based approach. Both are explained in section 3.8.1.

3) The candidate scan locations are uniformly sampled near the temporary navigation goal location which is displayed in red dots with a predefined number Nc of candidate positions $P_i = [x_i, y_i, z_i]^T$. Because that locations can capture more unknown voxels.

4) The distance between one scan location and others should be greater than the minimum distance threshold between scans. If there is some candidate scan locations in the threshold, it will be removed. This is required to achieve a spatially uniform candidate position which avoids overlapped scan coverage and scan redundancy.

5) The scan view calculation is performed by following equation. It will list in order to maximize the number of visible and void voxels, and to minimize the number of invisible voxels from each candidate scan location.

$$Scan\ Location = \underset{i}{\operatorname{argmax}} \left(\frac{\#voxels_i^{visible} + \#voxels_i^{void}}{\#voxels_i^{invisible}} \right) \qquad \text{Equation 13}$$

6) It should be checked at least one available path is existing from the current location to the selected scan location.

CHAPTER 4. EXPERIMENT RESULTS AND DISCUSSION

4.1 Experiment in Construction Site

4.1.1 Experimental Setup

To validate the proposed framework for autonomous 3D mapping with ground robot in an uneven outdoor environment, an experiment was conducted in a real-world construction site with uneven terrains and obstacles. The location of the experiment was located on the Georgia Institute of Technology campus. Figure 35 shows the test field used in the experiment. The test field contains a building, multiple structures, and obstacles such as construction equipment, materials, fences, rocks, trees, hills, trenches, and piles of dirt. Therefore, the validation for this testbed emphasizes on navigation accuracy compared to simulation and posture stability.

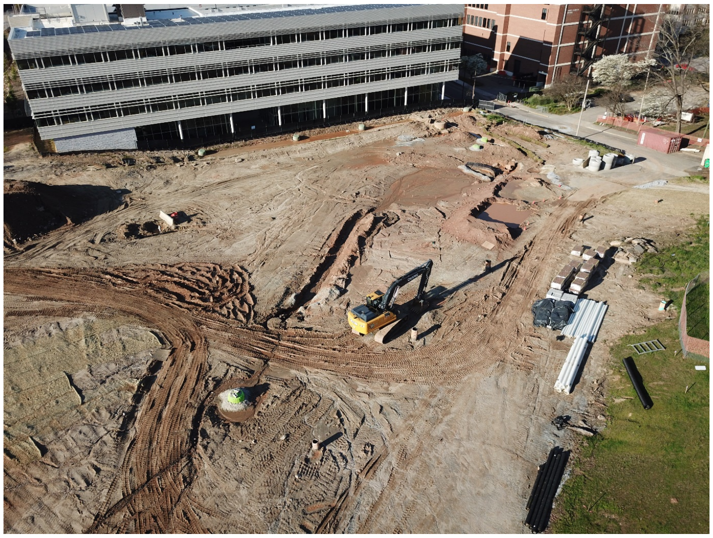


Figure 35 – Field test construction site surrounded by obstacles and hills

4.1.2 Experimental Scenario and Results

Before the experiment, the simulation of scan view calculation is performed to verify the scan completeness in virtual environment based on the approach described in section 3.9.2. The scan view planning proposed in this thesis locates satisfactory stationary scanning locations for ground robots by evaluating the candidate scan locations with a line-of-sight simulation using a 3D laser scanner. To generate this virtual environment, an UAV was deployed the target site at first to build 3D point cloud by using photogrammetry. The UAV-generated 3D point cloud, as shown in the top left side of Figure 36, was used to make the virtual environment which act a basic map to calculate the scan positions and the paths between them. The UAV-generated 3D point cloud converted to 3D voxel for reducing computational load, as shown in bottom middle of Figure 36. All of the 3D voxel size used in this research is 30 cm x 30 cm.

Table 4 demonstrates the results of scan planning simulations with various parameters. The last column represents the lack of completeness (LoC), which measures the completeness of the target site in terms of voxels. This is calculated using Equation 14 It calculates how many number of voxels that cannot be obtained when the stationary scans are performed at the selected scan locations compared to the number of voxels from the UAV-generation point cloud in percentage. In other words, LoC represents to the percentage of unscanned voxels, which are voxels that are not visible from any of the selected scan locations. Therefore, the lower number of LoC would better because the less number of voxels are missed. As shown in Table 4, all 8 simulations were successful in identifying a number of scan locations that cover the complete model with smaller than 4% LoC. Among the eight simulation trials in Table 4, the sixth trial of parameter set was chosen and used to estimate scan locations for the rest of this research. Figure 37 demonstrates the scan locations, planned and actual paths for the testbed which reflects the simulated result of Figure 37 and Table 4.

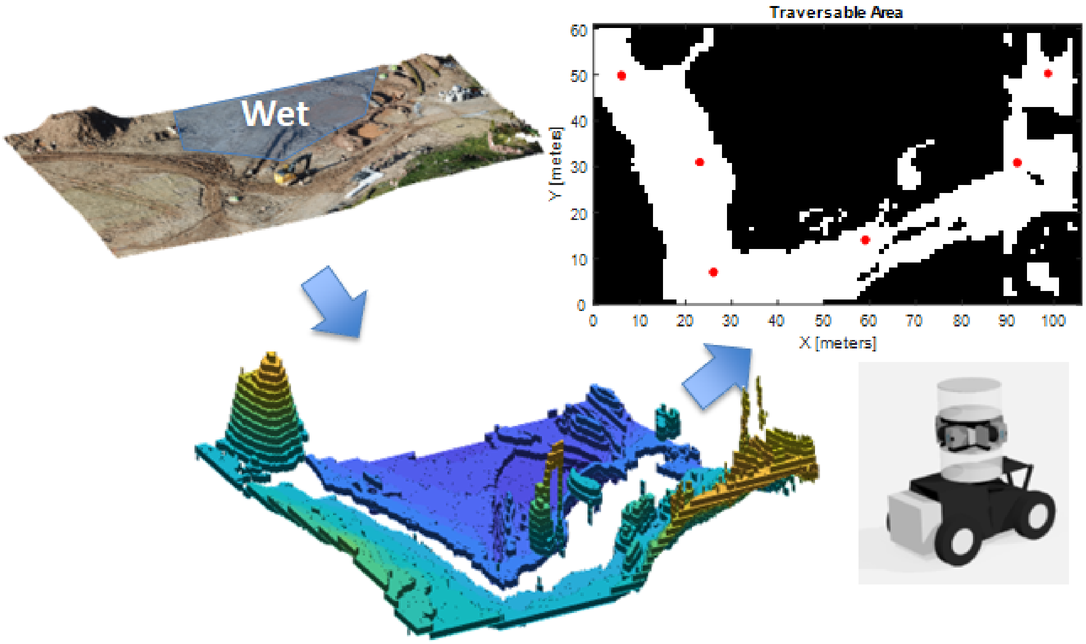


Figure 36 – Procedure of scan view planning with UAV-generated 3D point cloud

$$\mathrm{LoC} = 1 - \frac{\text{number of visible voxels from a scan location}}{\textit{total number of voxels in initial map}} \times 100 \qquad \text{Equation 14}$$

Table 4 – Results of scan view planning simulation

Trials	Min distance between scans (m)	Grid size (m)	Total number of candidate locations	Number of selected locations	LoC (%)
1	15	0.5	1145	8	3.67
2		1	557	7	3.31

3		1.5	278	7	2.78
4		2	131	6	2.42
5	20	0.5	1145	7	3.89
6		1	557	6	3.44
7		1.5	278	5	2.96
8		2	131	5	2.65

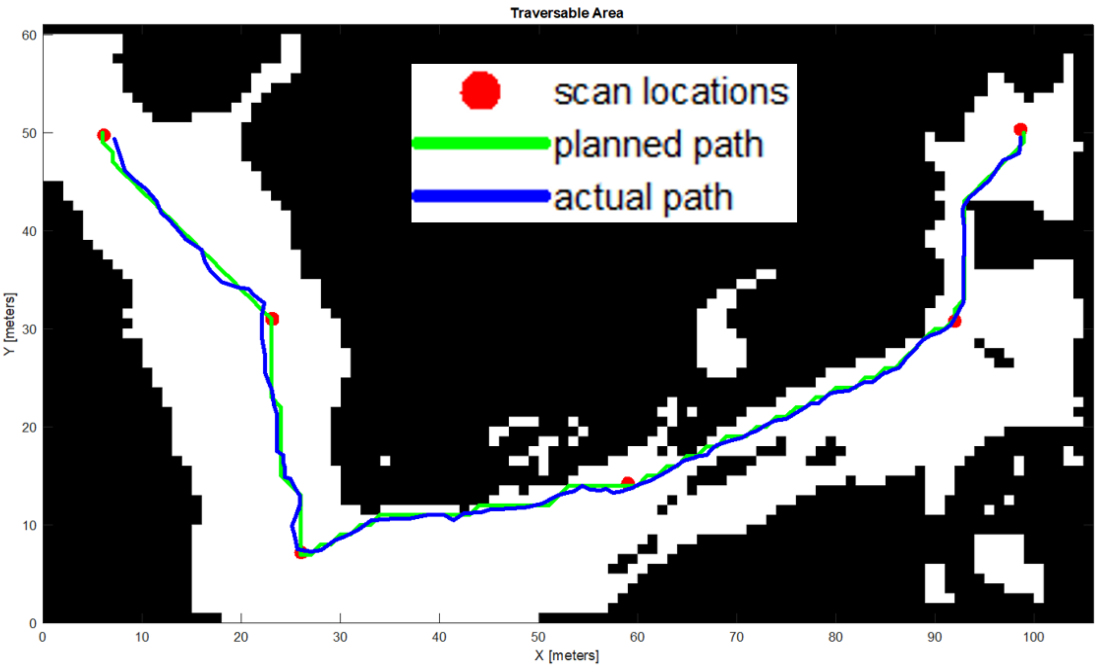


Figure 37 – Determined scan locations and the path from the scan planning simulation

In the experiment, the postural limits for GRoMI were set to both 15° for roll and pitch. The calculation with the shapes and weights of components of GRoMI showed that GRoMI withstands 20° roll angle and 35° pitch angle (See Figure 38). Since soft ground conditions cannot be detected by the LiDAR, however, the postural limits were set marginally to prevent possible tip-over of GRoMI. The linear maximum velocity of the path tracking controller was set to 0.5 m/s, with the proportional control constant 0.4. The global path planner and the path tracking controller stop when GRoMI is within 0.3 m of the goal location until a new goal location is given after scanning at the current location is finished.

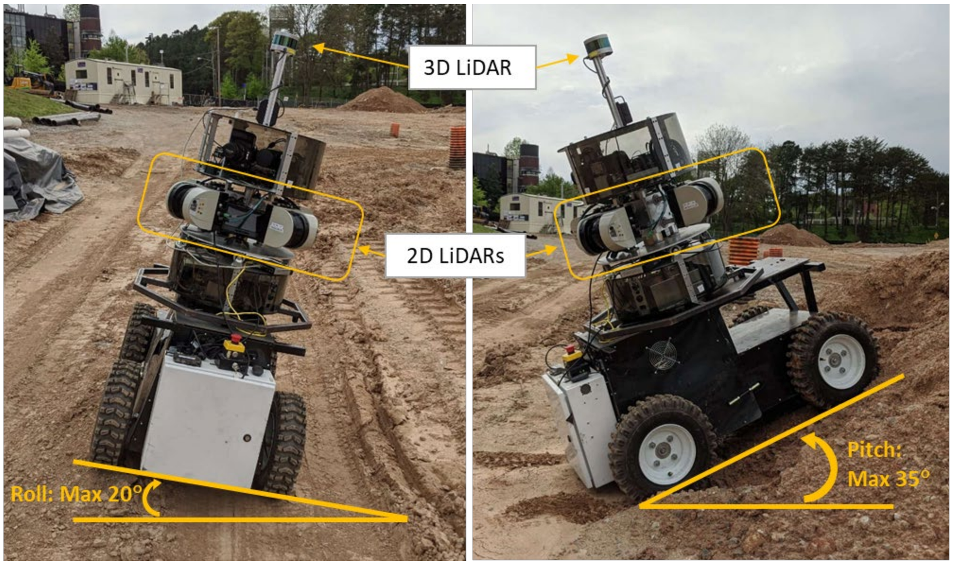
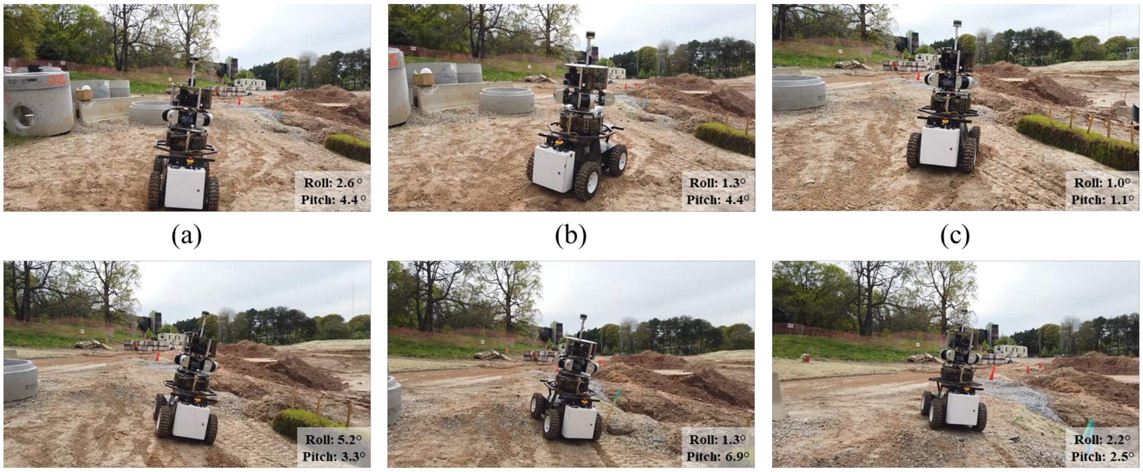


Figure 38 – GRoMI configuration with inclination limits

In the experiment, GRoMI safely navigated to all the scan locations and successfully finished the scanning operation without tipping-over or making a collision with obstacles. Figure 37 illustrates the trajectory of GRoMI following the path generated by the proposed method. Although the testing areas along the navigation trajectory from scan location 1 to 6 were surrounded by various obstacles such as fences, rocks, trees, construction equipment, and materials, the proposed architecture generated safe paths that minimized the distance and postural instability, and GRoMI avoided the obstacles and restricted areas and reached each of the scan locations safely. The average position difference between planned goals and arrived positions was 0.94 m.

While the paths between scan locations 2 and 6 at the test site were mostly planar and surrounded by only a few obstacles, the path from scan location 1 to scan location 2 had many obstacles to avoid with multiple small hills and piles of dirt that easily destabilizes the posture of GRoMI. However, GRoMI succeeded in reaching scan location 2 without tipping-over by minimizing the postural instability. Figure 39 shows that GRoMI passed through a narrow space between obstacles over an unavoidable hill. GRoMI changed the direction to a hill as it was the only possible way to reach the next scan location while avoiding obstacles. Figure 40 illustrates the pitch and roll angles of GRoMI in the experiment. Both pitch and roll angles mostly remained within the allowed ranges. Although the roll angle was close to the limit in the early period due to the localization error, it soon got back to the position within the limit and remained near 0°.

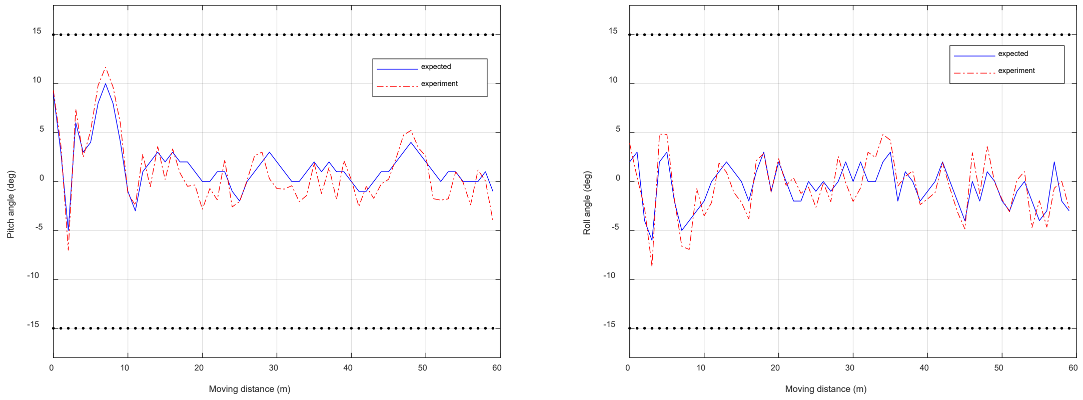


(a) (b) (c)

(d) (e) (f)

Figure 39 – GRoMI navigation which avoids obstacles and hills to minimize the postural instability. (a) t=10s. (b) t=20s. (c) t=30s. (d) t=40s. (e) t=50s. (f) t=60s.

A possible improvement identified in the experiments was to have the algorithm prefer a path on a paved road, if available, to the path with water, mud, or covered with thick grass. In the experiment, GRoMI often took a shortcut on such areas to minimize the path distance. However, the posture of GRoMI was slightly more unstable than expected since GRoMI often moved over small pits or rocks that were not detectible with laser scanners. If GRoMI instead prefers a path on hard flat ground, the postural instability will be further minimized. On wet grounds, GRoMI's wheels were often stuck in the mud, and GRoMI was not able to move out. As further work, the ground condition from camera images will be classified using a convolutional neural network, and the information will be added to the map. Then the path planner can generate path not only for the postural instability but also for the harsh ground conditions such as wet, muddy, or covered with thick grass areas, which will guide a safer and more robust path.



(a) (b)

Figure 40 – Postures of GRoMI. (a) Pitch angles. (b) Roll angles. (Blue solid line: expected angle, red dashed line: experimental result

4.2 Experiment in Disaster Site

4.2.1 Experimental Setup

In environments such as disaster sites after an earthquake or explosion, geometric data is difficult to obtain because these environments are hazardous. Therefore, the autonomous scanning platforms should be a reasonable and ethical solution to provide high-resolution 3D map data that can assist human activity such as exploration, mapping, or search and rescue. To achieve this purpose, the Guardian Center is shown in Figure 41, which has simulated damaged structures from earthquakes, hurricanes, and terrorist attacks, was selected to carry out experiments with the proposed architecture. In this environment, it was possible to investigate the ability of the inspection robot to complete time-critical scanning missions with registration accuracy and scan completeness as validation criteria.

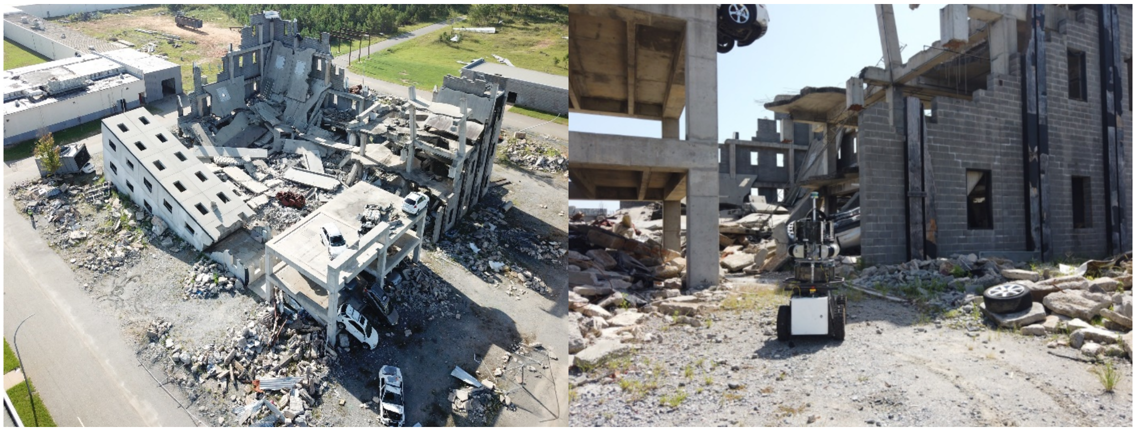


Figure 41 – Field test simulated disaster environment surrounded by debris

4.2.2 Experimental Scenario and Results

A construction site and a simulated disaster site were selected as testbeds for validating the proposed framework and the procedure is presented in the form of a flowchart in Figure 10. The target scanning region should be specified as a form of rectangle with four corner points coordinate which has an origin point at the initial location before it starts. As a first process, a high-resolution stationary scan is performed at the initial location. Once the stationary scan and RGB mapping are completed, the mobile robot starts to move and the dynamic pre-scanning process (i.e., 3D SLAM) is initiated. The horizontal scanner collects horizontal distance information, performs localization, and builds a 2D map using Hector SLAM. At the same time, the 3D LiDAR collects 3D information on the surroundings to avoid obstacles and build a pre-scanning 3D map with 3D SLAM. Figure 42 shows the dynamic pre-scanning and localization process. The coordinate symbol in the circle represents the current position of the mobile robot, the white line means the trajectory of the mobile robot, and the dots show the generated 3D point cloud with height-based color.

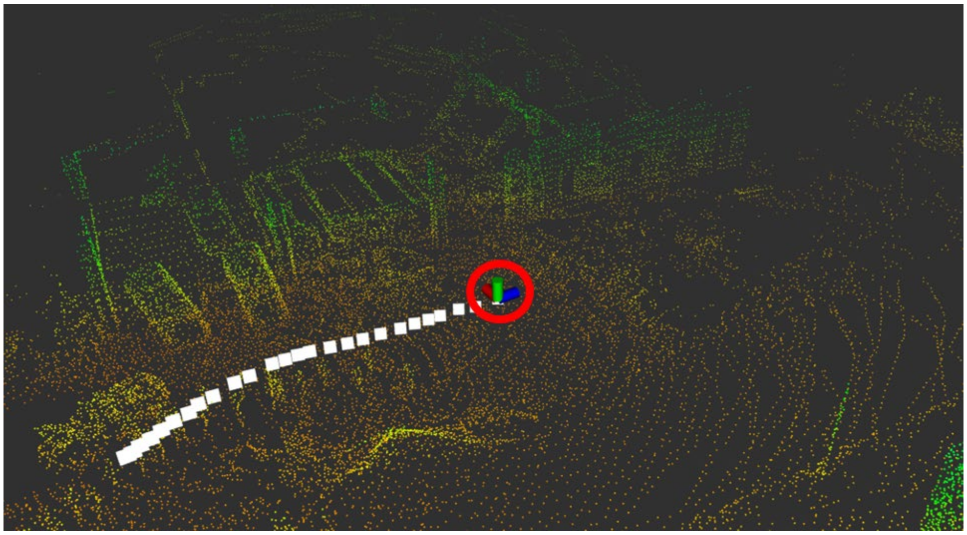
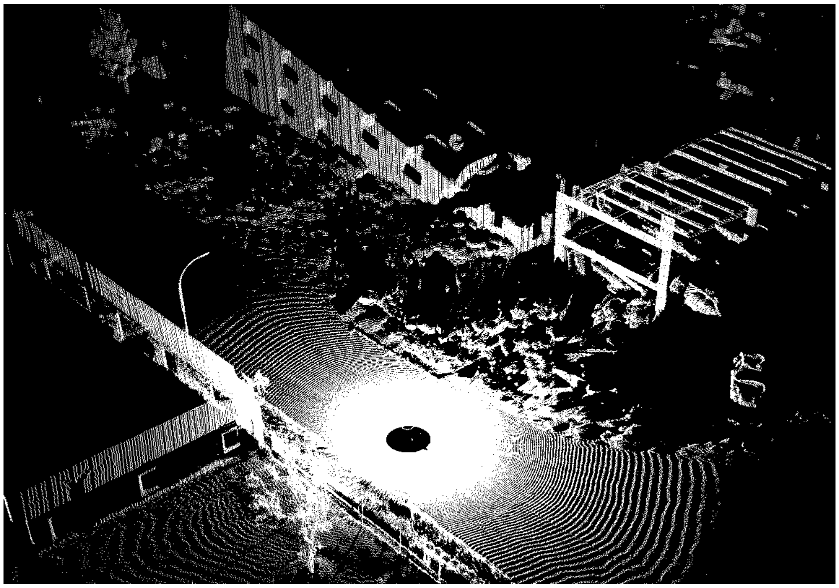


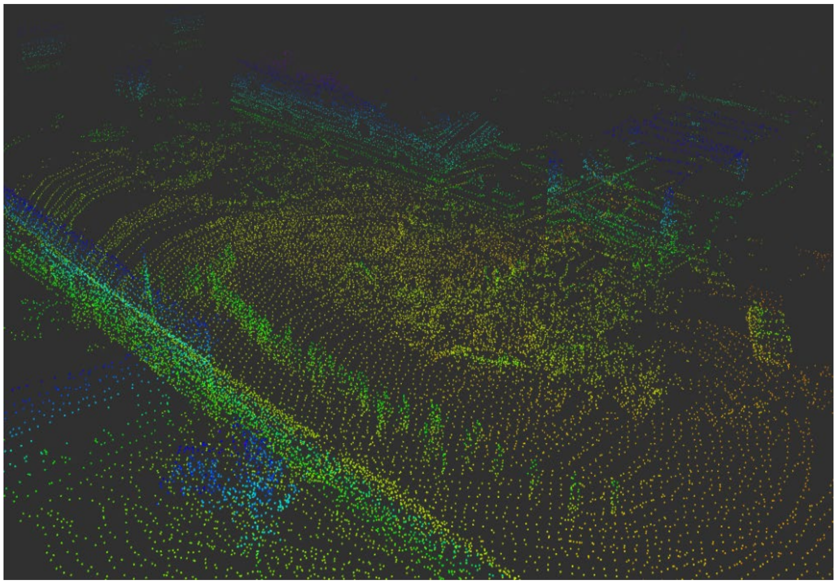
Figure 42 – 3D dynamic pre-scanning, localization, and trajectory

Although the pre-scanning process generates 3D point clouds during the dynamic scan, the proposed framework uses the static scan method to generate the final 3D point cloud. This is because the statically scanned point cloud provides more accurate visual information. Figure 43 compares the results of the statically and dynamically scanned point clouds. The dynamic pre-scan process is more susceptible to noise due to the continuous movement involved, and the points on the pre-scanning 3D map are less dense due to the large size of the data points. The static scan point cloud is not only less noisy but is also captured in higher resolution and includes color information. However, the dynamically scanned 3D information is useful for the robot to better understand the environment it must navigate as it includes features such as downhill slopes, uphill slopes, floor openings, ground obstacles, and entrance height.

While it moves to the goal and performs the dynamic pre-scanning, the scan view planning calculate the next best scan location for stationary scan by an evaluation of fitness technique, as described in section 3.9.2. While the mobile robot moves to the temporary navigation goal, the scan planning algorithm extracts the candidate scan locations near the navigation goal and tries to calculate the scan view based on the previously scanned data. The location estimated by scan planning should be an answer of question where is the best scan location to capture the unknown voxels as many as possible. Once the robot finds the location and arrives, the coordinate transformation is calculated from the previous stationary scan position to the current position by using SLAM localization data. These process cycle keeps going until it covers all the specified region that a ground robot can be reached. After the last stationary scan, the robot returns to the initial position and the final registered point cloud is generated. When all the scans are done, an RGB-texture mapping is performed for better visualization as shown in Figure 44 by using a panoramic view by using Equation 15.



(a) Stationary scan result



(b) Dynamic scan result

Figure 43 – Comparison of the result of stationary and dynamic scanned point cloud in the same region

$$x = \frac{1}{\Delta\theta} \tan^{-1} \frac{v}{w}$$

$$y = \frac{1}{\Delta\theta} \tan^{-1} \frac{w}{\sqrt{u^2 + v^2}} \qquad \text{Equation 15}$$

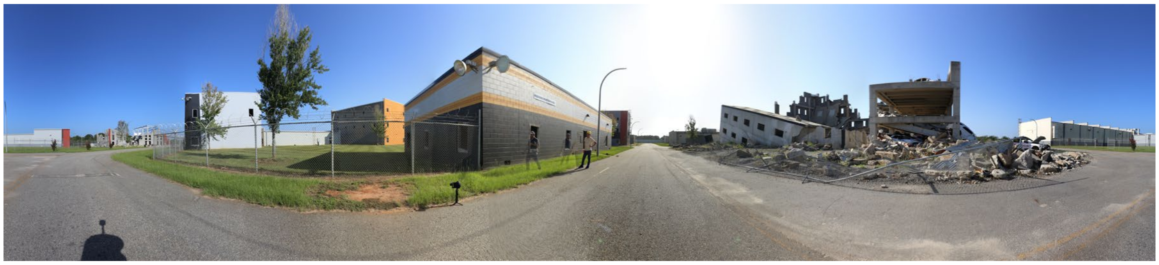


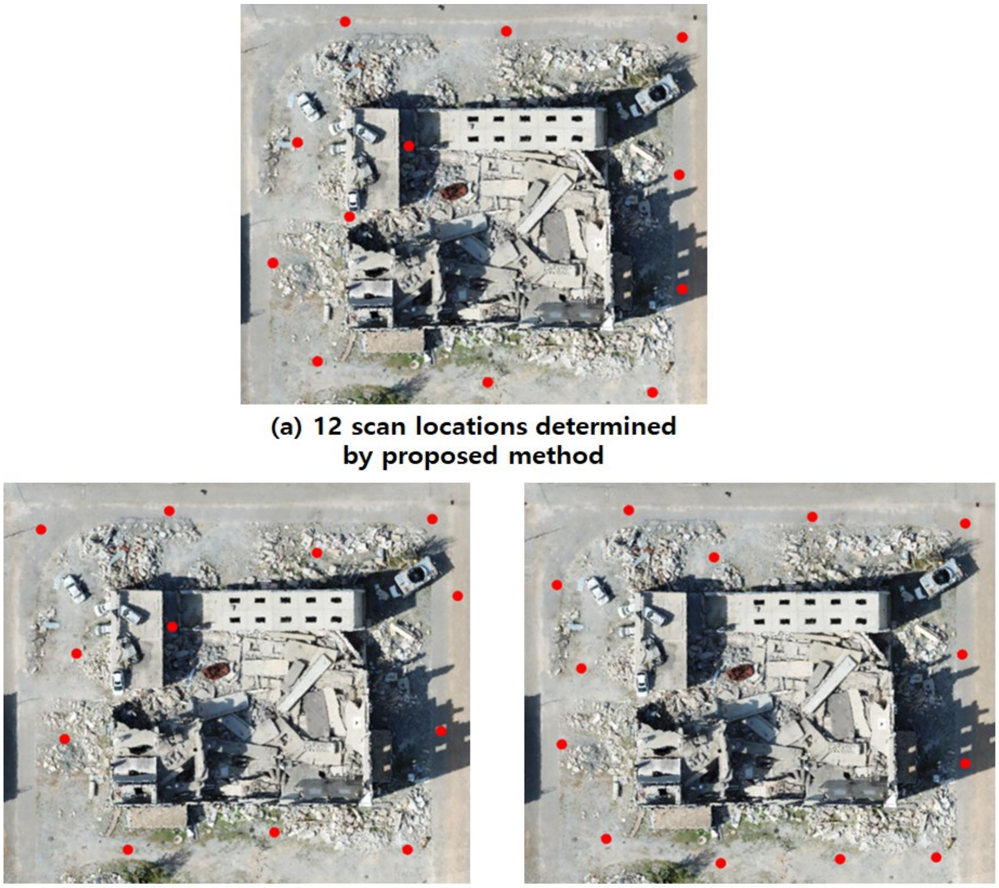
Figure 44 – Panorama image to map RGB data on to point cloud

To verify the efficiency of the proposed scan planning algorithm, it was compared to existing 2D NBV [60] and 3D NBV [61] methods in terms of the number of scans required to reconstruct the target scene and LoC. Most autonomous methods of environmental mapping use the currently built 2D map of the scene and estimate the next scan position based on the future visibility of the scene with low levels of visibility and occlusion. However, 2D information is highly incomplete in terms of occlusion in a 3D world and frequently leads sensors to erroneous or non-optimum positions. By contrast, 3D information-based NBV algorithms are more efficient when exploring volumes and inspecting surfaces of objects. The 3D NBV algorithm is executed in a voxel space with labels: unknown, free, and occupied. The algorithm determines the location with the most unknown voxels as the next scanning position. The approach is similar to the proposed scan planning algorithm. However, the normal 3D NBV does not consider occlusion by objects between the sensor and the target objects or the effects of scan redundancy.

The comparison of the scan location selection process was conducted in a simulated environment built from the UAV-generated point cloud. The scan location results for each case are presented in Figure 45. As Table 5 shows, the proposed method of scan planning generated twelve scan locations with less than 5% LoC, which was the lowest LoC, while the 2D and 3D NBV methods generated a similar or greater number of scans with 6~7% LoC. Therefore, the proposed scan planning approach was superior because it reduced the number of scans or kept the same number of scans with reduced LoC. The proposed method reduced the unscanned area by 2.498% compared to the 2D NBV algorithm with the same number of scans and reduced the unscanned area by 1.29% compared to the 3D NBV algorithm with one fewer scan.

Table 5 – Results of scan view planning

	Methods	Number of scans	Time (min)	LoC (%)
Simulation	Proposed method	12	0.924	4.925
	2D NBV	12	0.259	7.423
	3D NBV	13	0.895	6.215
Real	GRoMI (Automated data collection)	11	70	5.573
	Faro Scanner (manual data collection)	16	160	5.292



(a) 12 scan locations determined by proposed method

(b) 12 Scan locations determined by 2D NBV

(c) 13 Scan locations determined by 3D NBV

Figure 45 – Simulated scan locations for each scan planning algorithm from the map created by a drone

For the real-world experiment, the proposed scan planning method was applied in the GRoMI and generated eleven scan locations with 5.573% LoC. One difference between the simulation and the real-world experiment concerns whether information on the target site is known or not. For the simulation, the goal was to find all the scan locations with a known map; however, the goal of the real-world experiment with GRoMI was to find the next scan location using the currently built map, which contained partial information on the target site. Therefore, the latter cannot guarantee optimal locations. Nevertheless, the real-world experiment results of 5.573% LoC and eleven scan locations were relatively good compared with conventional approaches in terms of both the number of scans and LoC. By contrast, the results using human intuition and a Faro Scanner were sixteen scan locations with 5.292% LoC as shown in Figure 46. The number of scans was reduced remarkably (by five scans) and there was little difference (0.281%) between the LoC of the Faro Scanner and that of the GRoMI data as shown in Table 5. However, comparing human intuition to the algorithm is not rational, and one of the scan locations of the Faro Scanner was located in a region that was inaccessible for GRoMI, which was scan #16 in Figure 46. This is why the LoC of the Faro Scanner was better than that of GRoMI. Even though the comparison with the Faro Scanner was only for reference purposes, the proposed scan planning method reduced the number of scans required.

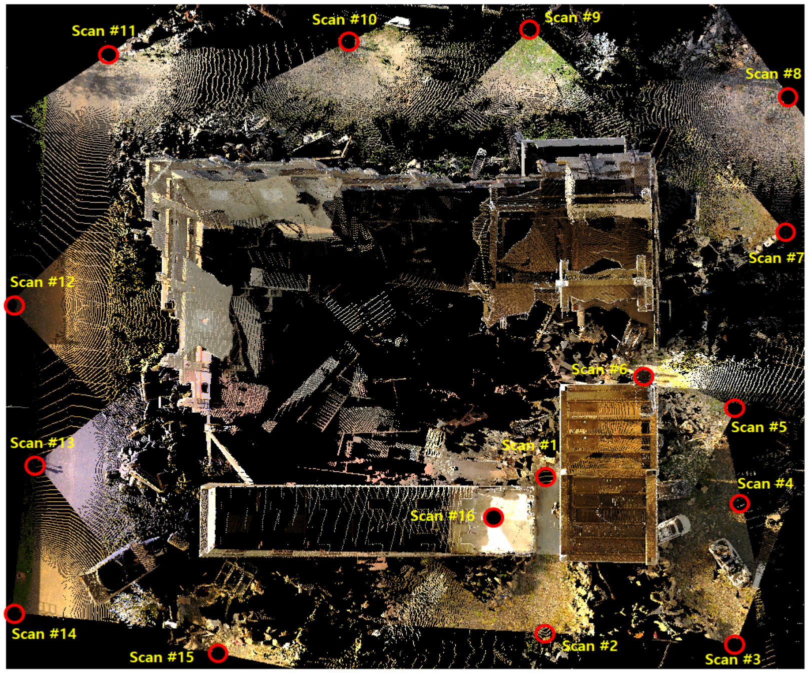


Figure 46 – Scan locations of Faro Scanner in a disaster site

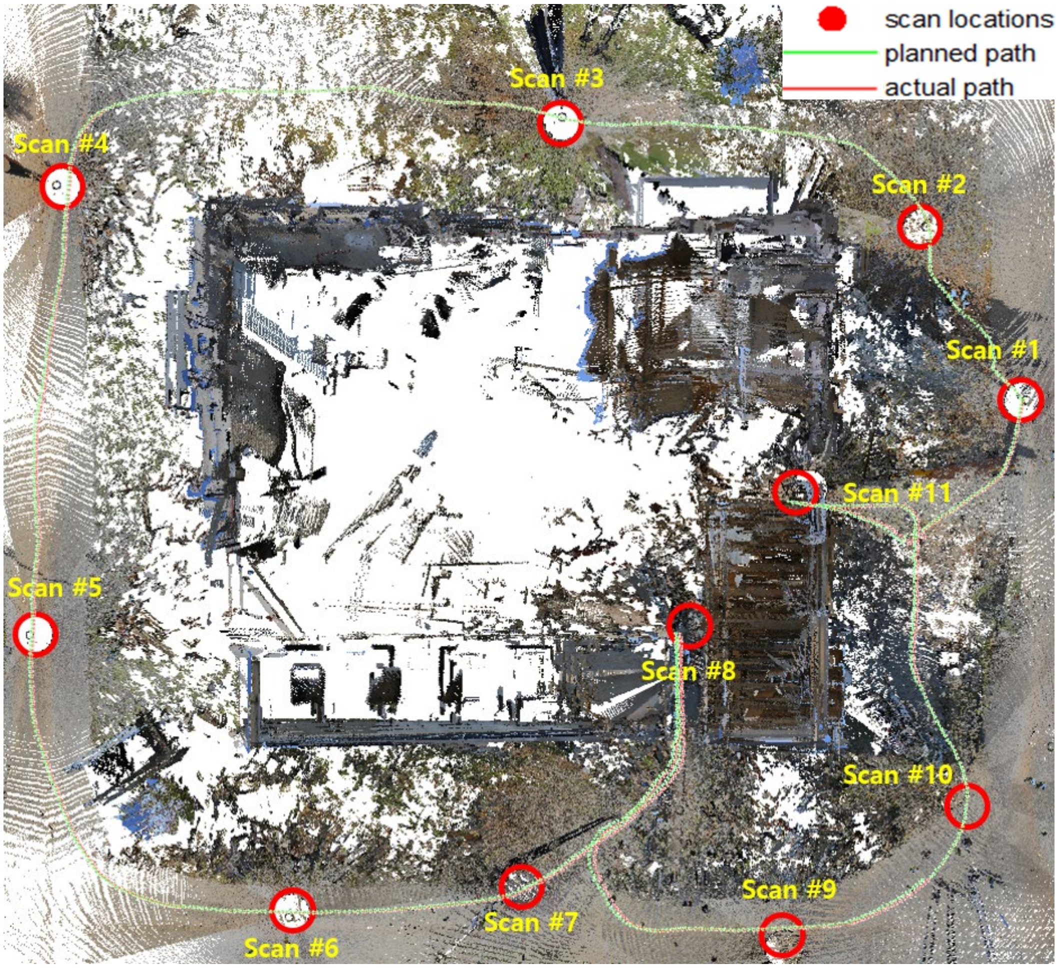


Figure 47 – Scan locations and path of mobile robot in a disaster site scanned by GRoMI

Figure 47 shows the selected eleven stationary scan locations of the target site and path between scan locations. These scan locations and path are determined while the ground robot is moving in real-time. Figure 48 demonstrates the result 3D point cloud by using Faro Scanner and GRoMI.

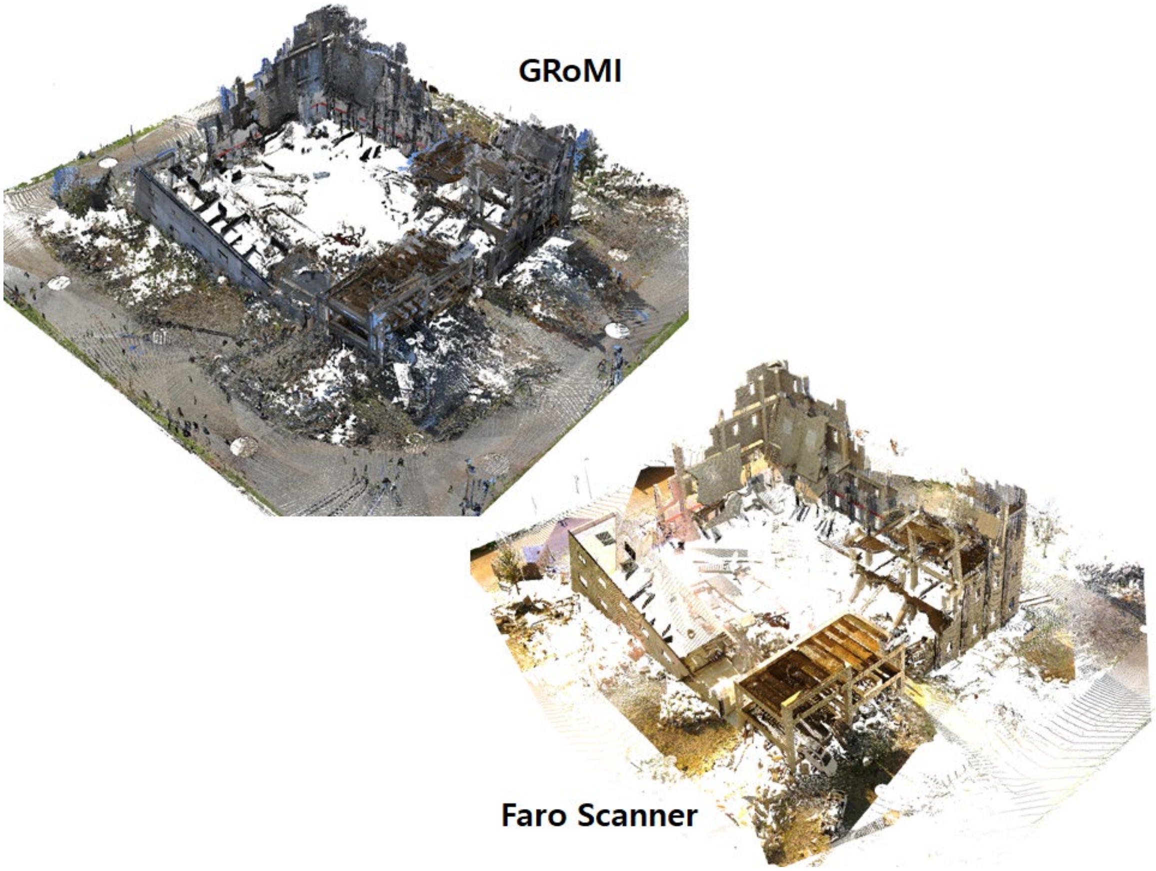


Figure 48 – Registered RGB point cloud both GRoMI and Faro generated

4.3 Discussion

These experimental results show that the proposed autonomous laser scanning framework can effectively collect and model large-scale sites or surrounding areas using a hybrid robotic data collection system without a priori knowledge of the target site. The results were favorable for both a construction site and a disaster site and for both simulated and real-world environments. The proposed method of scan planning generated twelve scan locations with less than 5% LoC, while the conventional methods generated a similar or greater number of scans with 6~7% LoC. This means that 3D-based view planning, which considers occlusions and scan redundancy, can improve the process of identifying scan locations.

Table 6 – Total length of path and RMSE deviation from the experiment

		Total path length (m)	RMSE deviation (m) (Plan-Actual)	$Ratio$ (%) $= \frac{RMSE\ deviation}{Path\ length}$
Construction site	Planned	146.17	-	-
	Actual	149.25	0.5426	0.36%
Disaster site	Planned	295.12	-	-
	Actual	297.78	0. 4195	0.14%

Table 7 – Total length of path and RMSE deviation from literature

Scenario	#1 [93]	#2 [93]	#3 [93]	#4 [93]	#5 [94]	#6 [95]	#7 [96]	#8 [97]	#9 [97]
Total path length (m)	97.12	97.12	196.35	196.35	10.68	21.28	40	70	80
RMSE deviation (m)	0.19	0.27	0.27	0.31	0.03	0.039	0.162	0.255	0.321
Ratio (%)	0.20	0.28	0.14	0.16	0.28	0.18	0.40	0.36	0.40

Table 6 demonstrates the navigation accuracy of the proposed framework by comparing the planned path to the actual path. The path planner generates the waypoints to the next goal. The total length of the planned path is computed from the distance from the first planned waypoint to the last, which is determined by drawing a straight line between each waypoint. The actual path joining the waypoints is calculated by tracking control. Therefore, the actual path is always longer than the planned path. The actual path was around 3.08 meters longer than the planned path for the construction test site and around 2.66 meters longer for the disaster test site. The RMSE deviation is also computed to show the extent to which the actual path was off track by taking the planned path as a ground truth. The RMSE deviation was 0.5426 meters for the construction site and 0.4195 meters for the disaster site. Navigation accuracy was reported in the literature [62–66] by measuring the RMSE of the degree of deviation from the planned path. However, this can vary depending on the shape or length of the path. Therefore, the ratios between the degree of deviation from the planned path and the total length of path are compared and evaluated, as shown in the last column of Table 6. The ratios reported in the literature vary from 0.16% to 0.4% in nine scenarios as shown in Table 7. The experiment results for this thesis show ratios of between 0.14% and 0.36%, which are comparable results reported in the literature. Even though the total length of the path through the disaster site is longer than the length of the path through the construction site, the overall navigation accuracy is better for the disaster site than for the construction site. This is because SLAM errors accumulate, so the longer the navigation length, the greater the error. Since the path through the construction site was not closed but open, the SLAM error kept accumulating. However, the mobile robot returned to the initial location in the disaster site, so a SLAM loop closure was performed. This is why the overall navigation accuracy is better for the disaster site. Therefore, when mission planning for the site mapping process with the proposed framework, a return to the initial location should be considered to improve navigation accuracy.

CHAPTER 5. CONCLUSION

The 3D laser scanning with mobile platform has become increasingly important in the civil engineering and construction fields. The operators are usually determinie the number and location of the scans in the data acquisition stage, however, this approach does not efficient because it is labor-intensive and time-consuming. The human intuitive decision making could include human error. To solve this problem, many researchers developed laser scanning platforms and had contributed to the remote sensing for civil engineering applications. However, the self-determined robotic laser scan, path planning, and SLAM for cluttered environments are still challenging if there is no a priori map.

To address these issues, the three sub-objectives were presented and validated using simulations and real outdoor experiments. First, the ground and moving objects in the point cloud were identified and removed for better SLAM calculation. The variation in the odometry estimation when moving object removal was applied was reduced to 28.6% that of the original algorithm, which means the odometry results were 3.5 times more accurate. Second, a navigation goal location and the path to that goal were estimated and continuously updating. Even though the navigation accuracy relies on the ground condition or shape of the path, the overall path accuracy with the RMSE metric was around 0.4~0.5 meters for 150~300 meters total path length, and the ratio of deviation compared to total path length was between 0.14% and 0.36%, which turned out acceptable in literatures' result. Lastly, the steps for determining effective scan view locations to capture the point cloud were presented. The proposed method reduced the unscanned region by 2.498%

compared to the 2D NBV algorithm with the same number of scans and by 1.29% compared to the 3D NBV algorithm with one fewer scan.

The advantages of the proposed framework over those presented in prior studies are as follows: (1) the framework automates the point cloud acquisition process by self-determined the robot's preferred scan locations and planning navigation paths, (2) the ground and moving objects are filtered out for better odometry calculation, and (3) it requires fewer scans to reconstruct large sites by reducing redundant scans and missing area using the scan view planning, thus reducing the time required for and cost of data collection and processing.

Even though all of the proposed methods in this dissertation have been validated through simulation and experiments, they can be further improved in future studies by considering aspects such as terrain materials conditions or roughness to provide more detailed estimates of traversability and level of slip. Moreover, there are still some regions that cannot be modeled due to their inaccessibility for a ground robot. If drones can be used with ground robots, the scanning performance can be improved. The movement of drones is more flexible than that of ground robots as the former are not affected by terrain conditions or ground obstacles (e.g., debris, holes) and, therefore, have better access to cluttered environments. However, drone-based data collection should be limited to zones that cannot be accessed by ground robots because the data quality (i.e., accuracy, density, noise level) from drones is not as good as that from ground robots.

REFERENCES

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, I. Reid, J.J. Leonard, Past , Present , and Future of Simultaneous Localization and Mapping : Toward the Robust-Perception Age, 32 (2016) 1309–1332.

[2] S. Koenig, C. Tovey, Y. Smirnov, Performance bounds for planning in unknown terrain, Artificial Intelligence. 147 (2003) 253–279. doi:10.1016/S0004-3702(03)00062-6.

[3] J. Ng, A Practical Comparison of Robot Path Planning Algorithms given only Local Information, n.d.

[4] M. Buehler, K. Iagnemma, S. Singh, The DARPA Urban Challenge: Autonomous Vehicles in City Traffic, 1st ed., Springer Publishing Company, Incorporated, 2009.

[5] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke, D. Johnston, S. Klumpp, D. Langer, A. Levandowski, J. Levinson, J. Marcil, D. Orenstein, J. Paefgen, I. Penny, A. Petrovskaya, M. Pflueger, G. Stanek, D. Stavens, A. Vogt, S. Thrun, Junior: The Stanford entry in the Urban Challenge, Journal of Field Robotics. 25 (2008) 569–597. doi:10.1002/rob.20258.

[6] P. Pfaff, R. Triebel, W. Burgard, An Efficient Extension to Elevation Maps for Outdoor Terrain Mapping and Loop Closing, The International Journal of Robotics Research. 26 (2007) 217–230. doi:10.1177/0278364906075165.

[7] T. Chang, S. Legowik, M.N. Abrams, Concealment and Obstacle Detection for Autonomous Driving, (1999) 1–15.

[8] R.D. Morton, E. Olson, Positive and negative obstacle detection using the HLD classifier, in: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2011: pp. 1579–1584. doi:10.1109/IROS.2011.6095142.

[9] S. Kuthirummal, A. Das, S. Samarasekera, A graph traversal based algorithm for obstacle detection using lidar or stereo, in: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2011: pp. 3874–3880. doi:10.1109/IROS.2011.6094685.

[10] C. Guo, W. Sato, L. Han, S. Mita, D. McAllester, Graph-based 2D road representation of 3D point clouds for intelligent vehicles, in: 2011 IEEE Intelligent Vehicles Symposium (IV), 2011: pp. 715–721. doi:10.1109/IVS.2011.5940502.

[11] J.-F. Lalonde, N. Vandapel, D.F. Huber, M. Hebert, Natural terrain classification using three-dimensional ladar data for ground robot mobility, Journal of Field Robotics. 23 (2006) 839–861. doi:10.1002/rob.20134.

[12] J. Pellenz, F. Neuhaus, D. Dillenberger, D. Gossow, D. Paulus, Mixed 2D/3D Perception for Autonomous Robots in Unstructured Environments BT - RoboCup 2010: Robot Soccer World Cup XIV, in: J. Ruiz-del-Solar, E. Chown, P.G. Plöger (Eds.), Springer Berlin Heidelberg, Berlin, Heidelberg, 2011: pp. 303–313.

[13] A. Sinha, P. Papadakis, Mind the gap: detection and traversability analysis of terrain gaps using LIDAR for safe robot navigation, Robotica. 31 (2013) 1085–1101. doi:DOI: 10.1017/S0263574713000349.

[14] D. Anguelov, B. Taskarf, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, A. Ng, Discriminative learning of Markov random fields for segmentation of 3D scan data, in: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), 2005: pp. 169–176 vol. 2. doi:10.1109/CVPR.2005.133.

[15] V.J. Lumelsky, A.A. Stepanov, Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape, Algorithmica. 2 (1987) 403–430. doi:10.1007/BF01840369.

[16] P.E. Hart, N.J. Nilsson, B. Raphael, A formal basis for the heuristic determination of minimum cost paths, IEEE Transactions on Systems Science and Cybernetics. 4 (1968) 100–107.

[17] S.M. LaValle, J. James J. Kuffner, Randomized Kinodynamic Planning, The International Journal of Robotics Research. 20 (2001) 378–400. doi:10.1177/02783640122067453.

[18] J. Nasir, F. Islam, U. Malik, Y. Ayaz, O. Hasan, M. Khan, M.S. Muhammad, RRT*-SMART: A Rapid Convergence Implementation of RRT*, International Journal of Advanced Robotic Systems. 10 (2013) 299. doi:10.5772/56718.

[19] I. Noreen, A. Khan, Z. Habib, A Comparison of RRT, RRT* and RRT*-Smart Path Planning Algorithms, in: 2016.

[20] A. Homam, H. Amin, Near Real-Time Motion Planning and Simulation of Cranes in Construction: Framework and System Architecture, Journal of Computing in Civil Engineering. 26 (2012) 54–63. doi:10.1061/(ASCE)CP.1943-5487.0000123.

[21] V.J. Lumelsky, a. a Stepanov, Dynamic path planning for a mobile automaton with limited information on the environment, IEEE Transactions on Automatic Control. 31 (1986) 1058–1063. doi:10.1109/TAC.1986.1104175.

[22] J. Borenstein, Y. Koren, Real-time obstacle avoidance for fast mobile robots in cluttered environments, IEEE International Conference on Robotics and Automation. (1990) 572–577. doi:10.1109/robot.1990.126042.

[23] T. Sobh, X. Xiong, Prototyping of robotic systems: applications of design and implementation, Information Science Reference. (2012) 498.

[24] D. Fox, B. Wolfram, S. Thrun, The Dynamic Window Approach to Collision Avoidance, Robotics & Automation Magazine, IEEE. (1997) 1–23. doi:10.1109/100.580977.

[25] V. Sequeira, J.G.M. Goncalves, M.I. Ribeiro, 3D reconstruction of indoor environments, in: Proceedings of 3rd IEEE International Conference on Image Processing, 1996: pp. 405–408 vol.2. doi:10.1109/ICIP.1996.560851.

[26] H. Surmann, A. Nüchter, J. Hertzberg, An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments, Robotics and Autonomous Systems. 45 (2003) 181–198. doi:10.1016/j.robot.2003.09.004.

[27] M. Strand, R. Dillmann, Using an attributed 2D-grid for next-best-view planning on 3D environment data for an autonomous robot, in: 2008 International Conference on Information and Automation, 2008: pp. 314–319. doi:10.1109/ICINFA.2008.4608017.

[28] N. Blodow, L.C. Goron, Z. Marton, D. Pangercic, T. Ruhr, M. Tenorth, M. Beetz, Autonomous semantic mapping for robots performing everyday manipulation tasks in kitchen environments, in: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2011: pp. 4263–4270. doi:10.1109/IROS.2011.6094665.

[29] C. Potthast, G.S. Sukhatme, A probabilistic framework for next best view estimation in a cluttered environment, Journal of Visual Communication and Image Representation. 25 (2014) 148–164. doi:10.1016/j.jvcir.2013.07.006.

[30] B. Charrow, G. Kahn, S. Patil, S. Liu, K. Goldberg, P. Abbeel, N. Michael, V. Kumar, Information-Theoretic Planning with Trajectory Optimization for Dense 3D Mapping, (n.d.).

[31] L. Iocchi, S. Pellegrini, BUILDING 3D MAPS WITH SEMANTIC ELEMENTS INTEGRATING 2D LASER , STEREO VISION AND IMU ON A MOBILE ROBOT, (2004).

[32] D. Borrmann, A. Nüchter, M. Đakulović, I. Maurović, I. Petrović, D. Osmanković, J. Velagić, A mobile robot based system for fully automated thermal 3D mapping, Advanced Engineering Informatics. 28 (2014) 425–440. doi:10.1016/j.aei.2014.06.002.

[33] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, R. Siegwart, Receding horizon path planning for 3D exploration and surface inspection, Autonomous Robots. 42 (2018) 291–306. doi:10.1007/s10514-016-9610-0.

[34] R.B. Rusu, Z.C. Marton, N. Blodow, M. Dolha, M. Beetz, Towards 3D Point cloud based object maps for household environments, Robotics and Autonomous Systems. 56 (2008) 927–941. doi:https://doi.org/10.1016/j.robot.2008.08.005.

[35] R. Kurazume, S. Oshima, S. Nagakura, Y. Jeong, Y. Iwashita, Automatic large-scale three dimensional modeling using cooperative multiple robots, Computer Vision and Image Understanding. 157 (2017) 25–42. doi:10.1016/j.cviu.2016.05.008.

[36] S.A. Prieto, B. Quintana, A. Adán, A.S. Vázquez, As-is building-structure reconstruction from a probabilistic next best scan approach, Robotics and Autonomous Systems. 94 (2017) 186–207. doi:https://doi.org/10.1016/j.robot.2017.04.016.

[37] P.S. Blaer, P.K. Allen, Data acquisition and view planning for 3-D modeling tasks, in: 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007: pp. 417–422. doi:10.1109/IROS.2007.4399581.

[38] Z. Meng, H. Qin, Z. Chen, X. Chen, H. Sun, F. Lin, M.H. Ang, A Two-Stage Optimized Next-View Planning Framework for 3-D Unknown Environment Exploration, and Structural Reconstruction, IEEE Robotics and Automation Letters. 2 (2017) 1680–1687. doi:10.1109/LRA.2017.2655144.

[39] C. Connolly, The determination of next best views, in: Proceedings. 1985 IEEE

International Conference on Robotics and Automation, 1985: pp. 432–435. doi:10.1109/ROBOT.1985.1087372.

[40] R. Kurazume, S. Oshima, S. Nagakura, Y. Jeong, Y. Iwashita, Automatic large-scale three dimensional modeling using cooperative multiple robots, Computer Vision and Image Understanding. 157 (2017) 25–42. doi:https://doi.org/10.1016/j.cviu.2016.05.008.

[41] B. Yamauchi, A frontier-based approach for autonomous exploration, in: Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. "Towards New Computational Principles for Robotics and Automation," 1997: pp. 146–151. doi:10.1109/CIRA.1997.613851.

[42] H. Moravec, A. Elfes, High resolution maps from wide angle sonar, in: Proceedings. 1985 IEEE International Conference on Robotics and Automation, 1985: pp. 116–121. doi:10.1109/ROBOT.1985.1087316.

[43] S. Kohlbrecher, O. Von Stryk, J. Meyer, U. Klingauf, A Flexible and Scalable SLAM System with Full 3D Motion Estimation, Proceedings of the 2011 IEEE International Symposium on Safety,Security and Rescue Robotics. Kyoto,Japa (2011) 155–160. doi:10.1109/SSRR.2011.6106777.

[44] D. Holz, S. Behnke, Sancta simplicitas - on the efficiency and achievable results of SLAM using ICP-based incremental registration, in: 2010 IEEE International Conference on Robotics and Automation, 2010: pp. 1380–1387. doi:10.1109/ROBOT.2010.5509918.

[45] G. Grisettiyz, C. Stachniss, W. Burgard, Improving Grid-based SLAM with Rao-Blackwellized Particle Filters by Adaptive Proposals and Selective Resampling, in: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, 2005: pp. 2432–2437. doi:10.1109/ROBOT.2005.1570477.

[46] A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss, W. Burgard, OctoMap: an efficient probabilistic 3D mapping framework based on octrees, Autonomous Robots. 34 (2013) 189–206. doi:10.1007/s10514-012-9321-0.

[47] D. Meagher, Geometric modeling using octree encoding, Computer Graphics and Image Processing. 19 (1982) 129–147. doi:https://doi.org/10.1016/0146-664X(82)90104-6.

[48] K.M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, W. Burgard, OctoMap : A Probabilistic , Flexible , and Compact 3 D Map Representation for Robotic Systems, in: 2010.

[49] J. Zhang, S. Singh, LOAM : Lidar Odometry and Mapping in Real-time, (n.d.).

[50] M. Shneier, T. Chang, T. Hong, W. Shackleford, R. Bostelman, J.S. Albus, Learning traversability models for autonomous mobile vehicles, (2008) 69–86. doi:10.1007/s10514-007-9063-6.

[51] L. Freda, G. Oriolo, Frontier-Based Probabilistic Strategies for Sensor-Based Exploration, in: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, 2005: pp. 3881–3887. doi:10.1109/ROBOT.2005.1570713.

[52] D.J. Webb, J.V.D. Berg, Kinodynamic RRT*: Optimal Motion Planning for Systems with Linear Differential Constraints, ArXiv. abs/1205.5 (2012).

[53] S. Koenig, M. Likhachev, D*lite, in: Eighteenth National Conference on Artificial Intelligence, American Association for Artificial Intelligence, USA, 2002: pp. 476–483.

[54] A. Stentz, The Focussed D* Algorithm for Real-Time Replanning, in: Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1995: pp. 1652–1659.

[55] J. Chen, P. Kim, Y.K. Cho, J. Ueda, Object-sensitive potential fields for mobile robot navigation and mapping in indoor environments, 2018 15th International Conference on Ubiquitous Robots, UR 2018. (2018) 328–333. doi:10.1109/URAI.2018.8441896.

[56] Y. Kuwata, G.A. Fiore, J. Teo, E. Frazzoli, J.P. How, Motion planning for urban driving using RRT, in: 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008: pp. 1681–1686. doi:10.1109/IROS.2008.4651075.

[57] C. Dornhege, A. Kleiner, A frontier-void-based approach for autonomous exploration in 3d, in: 2011 IEEE International Symposium on Safety, Security, and Rescue Robotics, 2011: pp. 351–356. doi:10.1109/SSRR.2011.6106778.

[58] D.D. Lichti, Ray-Tracing Method for Deriving Terrestrial Laser Scanner Systematic

Errors, 143 (2017) 1–5. doi:10.1061/(ASCE)SU.1943-5428.0000213.

[59] A. Adán, B. Quintana, A.S. Vázquez, A. Olivares, E. Parra, S. Prieto, Towards the automatic scanning of indoors with robots, Sensors (Switzerland). 15 (2015) 11551–11574. doi:10.3390/s150511551.

[60] M. Heidari Mozaffar, M. Varshosaz, Optimal placement of a terrestrial laser scanner with an emphasis on reducing occlusions, Photogrammetric Record. 31 (2016) 374–393. doi:10.1111/phor.12162.

[61] J. Chen, Y.K. Cho, Z. Kira, Multi-view Incremental Segmentation of 3D Point Clouds for Mobile Robots, IEEE Robotics and Automation Letters. (2019) 1. doi:10.1109/LRA.2019.2894915.

[62] S. Yan, Z. Yangyang, L. Lu, Path Following Control of Tracked Mobile Robot Based on Dual Heuristic Programming, in: 2019 5th International Conference on Control, Automation and Robotics (ICCAR), 2019: pp. 79–84. doi:10.1109/ICCAR.2019.8813386.

[63] X.V. Ha, C. Ha, H.S. Choi, Path Tracking Control of a Mobile Robot by Using Dual Estimation Algorithm, Advances in Mechanical Engineering. 5 (2013) 367127. doi:10.1155/2013/367127.

[64] T. Hiramatsu, Path Tracking Controller for Agricultural Vehicle, 2020. https://www.yanmar.com/kr/about/technology/technical_review/2020/0326_5.html .

[65] S.A.R. and P.H. Batavia, Evaluating Path Tracker Performance for Outdoor Mobile Robots, in: Pp. 388-397 in Automation Technology for Off-Road Equipment, Proceedings of the July 26-27, 2002 Conference (Chicago, Illinois, USA), ASABE, St. Joseph, MI, n.d. doi:https://doi.org/10.13031/2013.10028.

[66] O. Ringdahl, O. Lindroos, T. Hellström, D. Bergström, D. Athanassiadis, T. Nordfjell, Path tracking in forest terrain by an autonomous forwarder, Scandinavian Journal of Forest Research. 26 (2011) 350–359. doi:10.1080/02827581.2011.566889.