Automotive Lidar Technology for Marine Applications – Determining and Increasing the Accuracy of Simultaneous Localisation and Mapping

Berin Dikic
Instituto Superior Técnico
Lisbon, Portugal
berin.ba@gmail.com

***Abstract*—This work aims to develop a way of determining the accuracy of 3D Simultaneous Localisation And Mapping (SLAM) generated maps. The datasets for generating these maps are gathered by a novel mobile lidar sensor package called MObile LIdar SENsor System (MOLISENS), which uses inexpensive automotive lidar sensors. This makes it cheap, mobile and robust, thus it is suitable to be used in a multitude of different environments, including marine ones. The gathered data is then fed to four different SLAM algorithms that are tested, and their performance is evaluated based on a set of different criteria. Out of the tested algorithms, only the LIO-SAM algorithm was able to successfully complete the mapping task on all the datasets, and it also achieved the best map accuracy. However, even LIO-SAM had some deficiencies, mainly an inefficient and pretty basic loop-closure functionality. This deficiency was fixed by combining LIO-SAM with another algorithm called Scan Context, yielding an algorithm called SC-LIO-SAM. Scan Context specializes in loop detection, which makes the loop-closure functionality of the SC-LIO-SAM more accurate and robust. Besides making LIO-SAM more robust, by using SC-LIO-SAM, the accuracy of the generated 3D maps was also improved by incorporating Global Navigation Satellite System (GNSS) data into the SLAM process, as well as by tuning the parameters of the algorithm itself. In conclusion, SC-LIO-SAM emerges as the optimal algorithm for the specific use cases addressed in this work, while the MOLISENS sensor package demonstrates its capability to accurately survey a wide range of environments, including marine settings.**

***Index Terms*—Map accuracy, Marine mapping, SLAM, MOLISENS, lidar, LIO-SAM**

I. Introduction

The increase in popularity of Light Detection and Ranging (lidar) in the last few decades has been closely linked with the rise of autonomous vehicles. Lidar's worth was demonstrated during the Defence Advanced Research Projects Agency (DARPA) challenges in the 2000s, leading to its widespread adoption in the autonomous vehicle industry [1]. However, lidar usage has expanded beyond autonomous vehicles, finding applications in industries such as maritime, consumer electronics, aerospace, archaeology, architecture, and geography [2]. This research paper focuses on lidar applications in the marine industry or closely related fields. The demand for Autonomous Surface Vessels (ASVs) has increased for various purposes, including hydrology surveying, marine resource exploitation, ocean monitoring, and coastal transportation [3]. Lidar plays a crucial role in ASV applications, enabling self-localization, as well as obstacle detection and avoidance. Additionally, shipborne lidar is vital for weather monitoring, detecting obstacles and threats above and below water, environmental monitoring, and more [4]. Lidar's capability for 3D mapping, although not exclusive to the marine industry, is highly relevant, allowing for detailed depiction of coastlines, submerged caves, sea ice fronts, and other features [5].

There are various techniques for producing 3D maps, and lidars have emerged as a popular method. Terrestrial Laser Scanners (TLS) are commonly used for this purpose, despite their drawbacks such as high cost, weight, and lack of robustness. However, automotive lidar technology has rapidly advanced, resulting in cheaper, lighter, more accurate, and robust lidar systems. In collaboration with the University of Graz, Virtual Vehicle Research GmbH developed the MObile LIdar SENsor System (MOLISENS) [5] to complement or replace TLS in specific use cases that require lightweight, mobile, and modular sensor systems. MOLISENS is a standalone modular system that incorporates lidar, and it can be upgraded to include cameras and radars. Equipped with a Global Navigation Satellite System (GNSS) and an Inertial Measurement Unit (IMU), MOLISENS provides georeferenced orientation and positioning. It can operate independently in remote locations, offering high spatial and temporal resolution at a relatively low cost compared to TLS systems. Integrated with small industrial automotive lidars, MOLISENS generates high-resolution point clouds at high acquisition frequencies, making it suitable for dynamic environments. These lidars are more affordable and robust, with prices ranging from €5000 to €10000 and a higher IP68 rating [5].

The main focus of this work is to determine and increase the accuracy of 3D maps produced by the MOLISENS setup, currently utilizing the LIO-SAM Simultaneous Localization And Mapping (SLAM) algorithm [6], and compare it with the ground truth from the Riegl VZ-6000. While MOLISENS has been employed in various measurements such as coastal mapping, cave mapping, and glacier cave mapping [5], an analysis of SLAM-generated 3D maps has not been performed. The objective is to evaluate the accuracy of the current setup and subsequently analyze the results to identify potential improvements. This can involve parameter tuning of the LIO-SAM SLAM algorithm or exploration of alternative algorithms. Multiple SLAM algorithms are implemented and tested, with their results compared.

A. Simultaneous Localisation and Mapping

Simultaneous Localisation And Mapping (SLAM) is a method used by autonomous vehicles and robots to build maps of their environment and determine their location within it [7]. SLAM algorithms emerged in the late 1980s [8] and gained popularity in the 2000s [1]. Initially faced with computational challenges and expensive sensors, SLAM has become widely adopted due to improved processing power and reduced sensor costs. Lidar SLAM, which focuses on lidar point clouds, offers precise measurements, but can struggle with a limited number of distinct features. Maps are created by sequentially matching point clouds, often fused with data from other sensors for optimization [8].

II. Problem Statement

This work addresses the need for an alternative to stationary Terrestrial Laser Scanners (TLSs) in generating comprehensive 3D lidar point cloud maps of larger areas, particularly in marine environments, where TLSs cannot remain fully stationary for extended periods of time. The focus extends beyond marine applications, considering other use cases as well. The approach involves utilizing automotive lidar to capture 3D point clouds while in motion and employing a SLAM algorithm to create a 3D map. The setup for point cloud capturing, known as MOLISENS [5], has already been developed and tested. The main focus lies in the creation and evaluation of the 3D lidar point cloud map.

The process includes acquiring suitable datasets for testing multiple SLAM algorithms under various circumstances, incorporating both marine and land-based scenarios. The selection of appropriate SLAM algorithms is based on their capability to produce 3D point cloud maps and compatibility with the post-processing setup. Multiple SLAM methods are explored to determine the best-suited algorithm for mapping needs. Criteria such as computational requirements, map accuracy, and quality are considered for algorithm selection. Once the maps are generated, the challenge shifts to comparing the results, involving measuring SLAM map accuracy and considering additional criteria such as computational efficiency. Finally, efforts are made to improve the accuracy and computational performance of the selected "winning" SLAM algorithm by analyzing its core concept and suggesting enhancement methods.

III. Data Gathering and Processing Aparatus

A. MOLISENS

The MOLISENS framework, developed by Goelles et al. [5], enables the integration of small industrial sensors into a single sensor unit compatible with the ROS framework. This modular framework supports lidars, radars, and cameras, providing flexibility in sensor selection. The MOLISENS hardware setup follows Ingress Protection (IP) standards for small industrial sensors, ensuring its durability in harsh environments. The sensor unit consists of an Ouster OS1-64 or OS2-64 lidar, an ANN-MB series ublox active multi-band GNSS antenna, and a 9-axis Xsens MTi 630 IMU. With a standard 1/4-inch camera thread, the sensor unit can be mounted on camera handles or tripods, facilitating easy deployment in various scenarios. The MOLISENS setup can be powered by an AC/DC adapter or batteries, depending on the specific application conditions. It scans the environment, collects sensor data, and transmits it to a data logger for storage [5]. The MOLISENS setup is displayed in Figure 1.

Post-processing of the MOLISENS data offers multiple approaches. One option is to utilize SLAM algorithms for generating 3D maps, which can then be further processed using software like Cloud Compare. Alternatively, the raw sequence of point clouds can be replayed using the ROS framework to analyze the data in the recorded order. Another option is to extract and package useful information from the raw lidar data using the pointcloudset Python package. The raw lidar data contains not only the x, y, and z coordinates of each point in the 3D point cloud, but also additional details such as range, intensity, reflectivity, ambient near-infrared, and timestamps. This comprehensive data collection results in dataset sizes in the order of gigabytes, requiring efficient management and processing techniques to extract valuable insights from the recorded 3D point clouds [5].

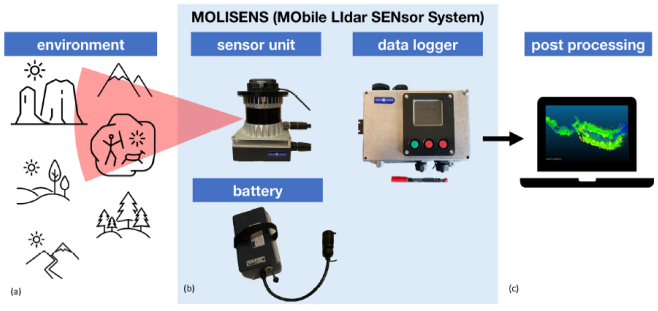

Fig. 1. MOLISENS hardware components and basic operating principle

B. Riegl VZ-6000 TLS

The Riegl VZ-6000 TLS is a powerful 3D scanner based on Riegl's exclusive V-Line technology, offering unmatched long-range reflectorless measurement performance of over 6000 m. It excels in challenging conditions, including reduced visibility caused by haze, rain, snow, or dust, making it ideal for measuring snowy and icy terrains. With a laser wavelength of 1064 nm that is minimally absorbed by snow, it delivers exceptional performance in such environments. The scanner features a broad Field of View (FoV), high-speed data collection, precise ranging capabilities, and advanced features such as target tracking, improved camera options, and onboard sensors. It finds application in various fields such as topography, mining, glacier mapping, snow field monitoring, and civil engineering. Combined with the Riegl software package, RISCAN PRO, it enables the generation of accurate 3D point cloud maps that serve as ground truth data for other measurement techniques, solidifying its position as a reliable tool for diverse applications [9].

C. Post-processing Setup

All data post-processing, as well as the creation of 3D point cloud SLAM maps, is performed on the same Dell Inspiron 14 5000 laptop that was bought in May 2020. Hardware specifications of the laptop are the following:

- **CPU:** Intel Core i7-10510U CPU with the clock speed of 1.80 GHz
- **RAM:** 20 GB DDR4
- **GPU:** NVIDIA GeForce MX230 with 2 GB of dedicated GDDR5 RAM
- **Internal storage:** PC SN740 NVMe WD SSD with 512 GB of storage capacity

This laptop is equipped with Ubuntu 20.04 Long Term Support (LTS) (Focal Fossa) and ROS Noetic Desktop Full version. The choice of this combination is based on Focal Fossa being the last Ubuntu version that supports ROS Noetic, which is the final version of ROS. Additionally, this setup enables a future upgrade from ROS to ROS2. To ensure fair and equal testing conditions for each SLAM algorithm, all data operations including post-processing and SLAM algorithm runs are carried out exclusively, without any other applications or programs running simultaneously in the background.

IV. Datasets

Throughout the course of this research, three different datasets are used to provide variability to the test conditions of each algorithm. Each dataset depicts a different environment, as well as a different mounting vehicle for the MOLISENS system. These individual datasets pose various challenges that the SLAM algorithms need to overcome. More details about each dataset are available in the following paragraphs. Table I offers a brief overview of all the datasets used, highlighting the common settings of a MOLISENS lidar sensor rotation frequency of 10 Hz and a horizontal resolution of 1024 points covering the entire 360° FoV.

One of the datasets used in this study is called *Sonnblick Observatory* and was created on the 21st of March 2023 near the Sonnblick observatory in Rauris Valley, Austria. It consists of two sets of data. The first set, known as the "ground truth" dataset, serves as a reference for generating 3D point cloud maps using different SLAM algorithms. It was collected using the Riegl VZ-6000 TLS positioned at the base of the gondola in the valley station. The second set of data is the "working" dataset, collected by the MOLISENS setup, employing an Ouster OS2-64 lidar. Due to the large size of the dataset, only a portion of the data closest to the valley station was utilized. This portion was selected for its distinct features and high level of "ground truth" detail, thanks to the TLS positioning in the valley station.

Another dataset utilized in this study, called *Virtuelles Fahrzeug Building*, was gathered on the 12th of December 2022 at the site of the Virtual Vehicle Research GmbH office building in Graz, Austria. It comprises the "working" dataset, capturing an urban environment with potential dynamic features like people and cars. Notably, the MOLISENS setup used in this survey was handheld, introducing a constant rocking motion of the sensors. An Ouster OS1-64 lidar was employed for data collection. Importantly, this dataset features a loop, indicating that the survey began and concluded at the same position. This loop structure proves valuable for evaluating the loop-closure capabilities of various SLAM algorithms.

The final dataset used in this study called *Rijeka Harbour*, was gathered on the 15th of October 2021 at Sušak Harbour in Rijeka, Croatia. It focuses on a maritime environment where the sensors experience continuous movement due to the natural motion of the sea. This dataset comprises the "working" dataset, collected using the MOLISENS setup mounted on a boat. The data was captured using an Ouster OS1-64 lidar. Notably, this dataset also features a loop structure, indicating that the survey started and ended at the same position.

V. Methodology

This section covers various aspects related to the accuracy evaluation of a 3D point cloud map. It discusses the impact of vegetation on lidar point clouds, providing insights into an algorithm that eliminates vegetation from a pre-existing 3D SLAM map. Additionally, a metric for comparing different SLAM algorithms is presented. The section also introduces the SLAM algorithms utilized in this work and explores methods to enhance the performance of the most effective algorithm.

TABLE I
OVERVIEW OF THE DATASETS USED IN THIS WORK

Dataset name	Date	Sensor carrier	Ouster sensor	Ground truth	Horizontal resolution	Rotation frequency
ViF building	12/12/2022	Handheld	OS1-64	NO	1024	10 Hz
Rijeka Harbour	15/10/2021	Boat	OS1-64	NO	1024	10 Hz
Sonnblick Observatory	21/03/2023	Gondola	OS2-64	YES	1024	10 Hz

A. Measuring the Accuracy of the SLAM Map

The user manual of Cloud Compare recommends the utilization of the Multiscale Model to Model Cloud Comparison (M3C2) algorithm [10] for robust and signed distance computations between two point clouds [11]. The M3C2 algorithm, introduced by Lague et al. [10], focuses on calculating distances only for specific points known as core points within the point clouds. This approach accelerates the computations by avoiding unnecessary measurements in areas of high point density commonly found in TLS point clouds. Core points are representative of the original reference point cloud and are considered regions of interest during the analysis. The algorithm employs a two-step process, as illustrated in Figure 2, to determine the distances between points of the two point clouds. In the first step, a normal vector $\overrightarrow{N}$ is computed for each point i within the core point cloud. This vector is established by fitting a plane to the neighbouring points of i within a given diameter D. The roughness of the point cloud $\sigma_i(D)$ at scale D is measured using the standard deviation of the neighbouring points from the best-fit plane. In the second step, the defined $\overrightarrow{N}$ is used to project point i onto each point cloud at a projection scale d. This projection is accomplished using a cylinder with a diameter of d, aligned with $\overrightarrow{N}$ and passing through point i. Two distributions of distances are obtained by intersecting the cylinder with the two point clouds. The standard deviations of these distributions provide estimates of the point cloud roughness $\sigma_1(d)$ and $\sigma_2(d)$ along the normal direction, while the means of the distributions i_1 and i_2 represent the average positions of the point clouds, also along the normal direction. Ultimately, the distance between the two point clouds $L_{\text{M3C2}}(i)$ is determined as the distance between i_1 and i_2 [10].

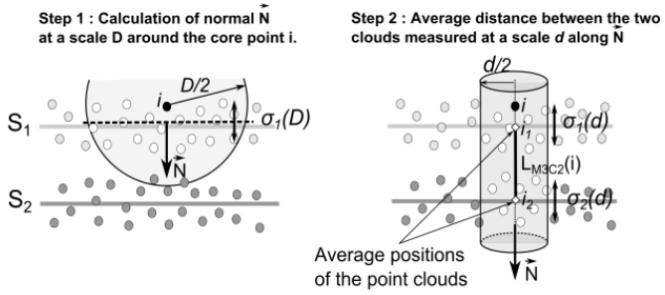

Fig. 2. M3C2 principle

B. Vegetation Removal

Zhang et al. [12] present a method called "Cloth simulation-based construction of pit-free canopy height models (CSF)" to construct accurate and complete Canopy Height Models (CHMs) from airborne lidar data. CHMs play a vital role in forest management, environmental monitoring, and urban planning applications. In this paper, the CSF method is applied to remove vegetation from a pre-existing 3D SLAM map, aiming to create a vegetation-free ground map for precise map comparisons. Traditional CHM construction methods often suffer from data gaps or pits due to occlusion, penetration, or attenuation issues, which significantly affect the accuracy and completeness of the resulting CHMs. To overcome these limitations, the CSF method employs a cloth simulation algorithm that simulates the movement and deformation of a virtual cloth representing the canopy. By simulating the cloth's interaction with lidar data points, the algorithm estimates the canopy surface height at various locations, resulting in improved CHMs [12].

The algorithm's operation can be summarised in the following steps:

1) Data pre-processing: The lidar data is pre-processed to remove noise and outliers and to generate a Digital Terrain Model (DTM),
2) Cloth simulation: The canopy is represented as a virtual cloth made up of a grid of nodes. The cloth simulation algorithm simulates the movement and deformation of the cloth by applying forces to each node based on its position relative to the lidar data points. The forces include gravity, tension, and collision forces. The tension force keeps the cloth taut, while the collision force prevents the cloth from penetrating the lidar data points,
3) Height estimation: Once the cloth simulation is complete, the height of the canopy surface at each location can be estimated by interpolating the heights of the nodes in the virtual cloth,
4) Calibration: The cloth simulation algorithm's parameters are calibrated using a set of ground-truth data to ensure that the simulated canopy surface matches the actual canopy surface,
5) CHM generation: The final CHM is generated by subtracting the DTM from the estimated canopy surface height at each location [12].

C. SLAM Algorithm Performance Metric

To compare different SLAM algorithms, the following criteria can be used:

1) **Visual Inspection**: This criterion involves visually inspecting the output of the SLAM algorithm to assess the quality of the generated maps. It includes evaluating the level of detail, completeness, and overall visual fidelity of the maps.
2) **Map Accuracy**: This criterion measures the accuracy of the generated maps by comparing them to ground truth or reference maps. It assesses how closely the SLAM algorithm is able to estimate the true positions and orientations of the environment.
3) **Map Resolution**: Map resolution refers to the level of detail captured by the SLAM algorithm. It can be determined by the number of points or features in the 3D point cloud map. Higher-resolution maps capture more details and provide a more precise representation of the environment.
4) **Loop-Closure Availability**: Loop closure refers to the ability of a SLAM algorithm to detect and close loops in the trajectory, which occurs when the system revisits a previously visited location. Loop closure is important for accurate mapping and localization. This criterion assesses whether a SLAM algorithm supports loop closure or not.
5) **Playback Factor**: The playback factor refers to the real-time performance of SLAM algorithms. Some algorithms may struggle to run in real-time due to computational demands and hardware limitations. To address this, the lidar data can be played back at a slower rate (e.g., 0.5 or 0.1), simulating a higher-performance setup and enabling better map accuracy.
6) **Implementation Difficulty**: This criterion evaluates the complexity and difficulty of implementing and deploying the SLAM algorithm. It considers factors such as the required hardware, software dependencies, calibration procedures, parameter tuning, and overall ease of integration into a specific application.

The chosen criteria in this work primarily focus on map quality. However, there are additional mainstream criteria mentioned by Valentin [13], such as Absolute Pose Error (APE), Relative Translation Error (RTE), Relative Pose Error (RPE), and Average Distance to the Nearest Neighbor (ADNN). The M3C2 method used for map accuracy is an improvement upon the ADNN method. Valentin [13] has shown that these criteria are heavily correlated, indicating that selecting one criterion is sufficient for evaluating SLAM algorithms.

D. Chosen SLAM Algorithms

Multiple sources were considered to choose suitable SLAM algorithms for testing. The Autoware website [14], along with works by Garigipati et al. [15] and Valentin [13], which specifically focused on ROS implementations, served as the basis for algorithm selection. From these sources, a list of eight candidate algorithms was compiled, and their main features are presented in Table II.

Out of the eight algorithms considered, FD-SLAM was excluded due to its reliance on GNSS data. Among the remaining algorithms, FAST-LIO-LC, ISCLOAM, and A-LOAM were found to be incompatible with the post-processing setup utilized in this thesis, likely because they were designed for older versions of ROS.

Ultimately, four commonly used algorithms were selected: *KISS-ICP*, *LeGO-LOAM*, *HDLGraph-SLAM*, and *LIO-SAM*. The overall objective was to find a balance between 3D map quality and the speed and simplicity of construction. These algorithms employ different technologies and can be categorized as either full SLAM or odometry and mapping. Full SLAM approaches aim to maintain global consistency through loop-closure detection, while odometry-based methods estimate position incrementally frame-by-frame.

E. Performance Optimisation

Based on the comparison results of the chosen SLAM algorithms, it was determined that LIO-SAM is the optimal algorithm for the specific use cases in this thesis. To further enhance LIO-SAM's performance, three improvement approaches were employed, which will be introduced in this section.

1) Loop-closure Improvement: The chosen SLAM algorithm, LIO-SAM, has shown promising results in comparison to other algorithms based on the evaluation criteria. However, there is room for improvement, particularly in the loop-closure method employed by LIO-SAM.

To improve the loop-closure capabilities of LIO-SAM, one of the approaches used is SC-LIO-SAM, which integrates the Scan Context algorithm developed by G. Kim and A. Kim [16]. Scan Context introduces a novel spatial descriptor for global place recognition using 3D lidar scans. By directly capturing the 3D structure of the environment without relying on histograms or prior training, it provides a robust foundation for loop closure. SC-LIO-SAM utilizes grid-based partitioning and height-based representation to summarize the 3D structure of the lidar scans. It incorporates root shifting and considers column shifting to enhance recognition robustness against translation and viewpoint changes. With a two-phase search algorithm for loop closure detection and a rotation-invariant descriptor known as the "ring key," SC-LIO-SAM aims to significantly enhance the loop-closure capabilities of the LIO-SAM algorithm.

2) GNSS Data Inclusion: Integrating GNSS data in SLAM significantly improves the mapping process. By incorporating accurate location information, GNSS enhances the initial pose estimation, establishing a global frame of reference for the map. This enables correct alignment of subsequent measurements and serves as an initialization step. By periodically integrating GNSS data with SLAM estimates, drift is corrected, ensuring long-term localization accuracy.

TABLE II
CONSIDERED SLAM ALGORITHMS

	Required sensors	Optional sensors	ROS version	Newest ROS release	Loop-closure	Tested version	Repository stars	Latest update
FAST-LIO-LC	Lidar & IMU	GNSS	ROS 1	ROS Melodic	NO	d059255	191	11/03/22
FD-SLAM	Lidar & GNSS	IMU	ROS 1	ROS Noetic	YES	-	13	13/04/22
HDL-Graph-SLAM	**Lidar**	**IMU & GNSS**	**ROS 1**	**ROS Noetic**	**YES**	**cb7af42**	**1600**	**28/02/23**
ISCLOAM	Lidar	-	ROS 1	ROS Melodic	YES	d6d7c61	458	13/03/21
LeGO-LOAM	**Lidar**	**IMU**	**ROS 1**	**ROS Melodic**	**YES**	**896a7a9**	**2000**	**02/07/20**
LIO-SAM	**Lidar & IMU**	**GNSS**	**ROS 1 & ROS 2**	**ROS Noetic & ROS Humble**	**YES**	**0be1fbe**	**2500**	**17/04/23**
KISS-ICP	**Lidar**	**-**	**ROS 1 & ROS 2**	**ROS Noetic & ROS Humble**	**NO**	**959e507**	**883**	**06/06/23**
A-LOAM	Lidar	-	ROS 1	ROS Melodic	NO	e51f88c	1700	28/03/19

Combining GNSS measurements with other sensor data reduces uncertainties, improves map quality, and addresses data association challenges. GNSS acts as a constraint to align local maps with the global reference frame, enhancing the overall robustness of the SLAM system [17].

In summary, integrating GNSS data into SLAM offers benefits such as establishing a global frame of reference, improving initialization, ensuring long-term localization accuracy, enhancing mapping precision, and assisting in data association. Both LIO-SAM and SC-LIO-SAM include GNSS data integration capabilities. Enabling this functionality involves activating two specific features and adjusting two parameters within the algorithm's configuration settings.

3) Parameter Optimisation: The last approach to improving LIO-SAM's performance involves tuning the algorithm's parameters. With over 30 user-defined parameters, a selection was made to identify which parameters could be optimized to enhance the accuracy of the generated 3D point cloud map. However, limited documentation and information on the purpose of each parameter posed a challenge in determining which parameters to modify. Ultimately, eight parameters were selected and categorized into four groups: *CPU parameters*, *LOAM feature threshold parameters*, *surrounding map parameters*, and *visualization parameters*. These parameter groups, suggested by Shan et al. [6], directly influence the mapping results. Each group of parameters is individually tested and evaluated to determine their impact on map accuracy. By adjusting and combining the best-performing values from these parameter groups, the goal is to achieve the highest possible accuracy for the final 3D map.

VI. RESULTS

This section presents an overview of the results acquired by generating 3D point cloud maps using the SLAM algorithms introduced earlier. The comparison results of the chosen SLAM algorithms are discussed to analyze their respective performances. Additionally, the section explores the outcomes of performance optimization attempts conducted on the best-performing SLAM algorithm.

A. Comparison Results

Initially, an analysis was conducted on the general characteristics, implementation difficulty, and usage complexity of the chosen SLAM algorithms, which are summarized in Table III. It was observed that KISS-ICP is the only algorithm without loop-closure functionality. LeGO-LOAM emerged as the most challenging algorithm to implement and use, with compatibility limited to ROS Melodic, requiring substantial modifications for ROS Noetic. Despite facing issues with map saving functionality, the comparison quality remained unaffected. Usage of LeGO-LOAM proved complex, as all configurable parameters were confined within a C++ header file, necessitating the inconvenient rebuilding of the entire ROS workspace for any modifications. In contrast, HDL-Graph-SLAM offered a straightforward installation process, albeit with a difficult and unintuitive usage due to over 150 changeable parameters. Both KISS-ICP and LIO-SAM required minor source code modifications for installation, but KISS-ICP presented complexity in saving the map, while LIO-SAM proved more user-friendly despite its 30+ configurable parameters. Notably, the 3D point cloud map could be saved with a single command.

TABLE III
CHOSEN SLAM ALGORITHM CHARACTERISTICS

	Loop-closure	Installation difficulty	Usage difficulty
KISS-ICP	NO	Medium	Medium
HDL-Graph-SLAM	**YES**	Easy	Medium
LeGO-LOAM	**YES**	Hard	Hard
LIO-SAM	**YES**	Medium	Easy

Table IV displays the results of testing various SLAM algorithms on different datasets using a rosbag playback factor of 1 for real-time evaluation. The LIO-SAM algorithm consistently generates 3D SLAM point cloud maps with a resolution that is at least two times better than the other algorithms tested. Furthermore, LIO-SAM is the

TABLE IV
GENERAL SLAM ALGORITHM COMPARISON RESULTS

		Map resolution (points)	Playback factor	Visual inspection
Rijeka Harbour	KISS-ICP	2,190,438	1	Fail
	HDL-Graph-SLAM	1,662,611	1	Fail
	LeGO-LOAM	-	1	Fail
	LIO-SAM	**5,372,780**	1	**Fail/Success**
ViF Building	KISS-ICP	640,608	1	**Success**
	HDL-Graph-SLAM	372,573	1	**Success**
	LeGO-LOAM	-	1	Fail
	LIO-SAM	**1,248,135**	1	**Success**
Sonnblick Observatory	KISS-ICP	1,316,906	1	**Success**
	HDL-Graph-SLAM	1,711,318	1	Fail
	LeGO-LOAM	-	1	Fail
	LIO-SAM	**11,132,938**	1	**Success**

only algorithm that successfully generates maps for all datasets based on the *visual inspection* criterion. The detailed performance information based on this criterion, for each SLAM algorithm on each dataset can be found later in this section, providing further insights into their respective performances.

1) Rijeka Harbour Dataset: During the evaluation, it was observed that the LeGO-LOAM algorithm failed to map the entire survey area, while the KISS-ICP algorithm completed the mapping process without a loop-closure algorithm, resulting in a distorted map with intersecting parts. Similarly, the HDL-Graph-SLAM and LIO-SAM algorithms successfully generated 3D maps, but the loop-closure algorithm failed to properly close the loop in both cases, leading to imperfect merging of certain map sections. Consequently, none of the mentioned algorithms achieved complete success in the mapping process based on the *visual inspection* criterion.

Due to the lack of complete success in the mapping process using the tested algorithms, an alternative experiment was conducted. The dataset was modified by removing the initial noisy lidar and IMU data, which contained significant spikes and noise, until the point when the surveying equipment started moving. By making this modification, the LIO-SAM algorithm successfully generated a 3D map of the dataset, as shown in Figure 3. However, it is worth noting that modifying the data in this way may not be feasible or applicable in all future scenarios. The results of implementing potential approaches to mitigate the impact of noisy initial data are discussed later.

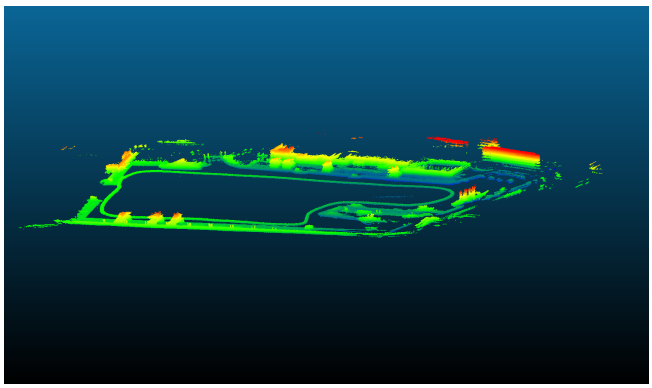

Fig. 3. Successful LIO-SAM generated map - Rijeka Harbour dataset

2) "Virtuelles Fahrzeug" Building Dataset: During the evaluation, it was observed that the LeGO-LOAM algorithm completed the mapping process, but without a functioning loop-closure feature, resulting in a distorted map where a building's outside wall intersected with itself. Similarly, the KISS-ICP algorithm initially appeared to produce a satisfactory map, but closer inspection revealed distortion and duplicated features due to the lack of a loop-closure mechanism. Despite these issues, the KISS-ICP algorithm successfully completed the mapping process according to the *visual inspection* criterion, unlike LeGO-LOAM. On the other hand, both the HDL-Graph-SLAM and LIO-SAM algorithms successfully generated 3D point cloud maps of the ViF building and its surroundings. Additionally, the loop-closure process was reasonably well-executed by both algorithms, indicating the successful completion of the mapping process for both approaches.

TABLE V
MAP ACCURACY - SLAM ALGORITHM COMPARISON RESULTS

	Sonnblick Observatory			Sonnblick Observatory - vegetation removed		
	KISS-ICP	**HDG-Graph-SLAM**	**LIO-SAM**	**KISS-ICP**	**HDG-Graph-SLAM**	**LIO-SAM**
Map accuracy (+-5m)	42.90%	76.26%	**88.07%**	55.41%	31.29%	**95.13%**
Map accuracy (+-3m)	29.38%	56.83%	**81.02%**	19.54%	19.06%	**92.93%**
Map accuracy (+-2m)	20.68%	47.17%	**73.37%**	7.36%	12.93%	**89.57%**
Map accuracy (+-1m)	11.99%	28.32%	**56.48%**	3.23%	7.17%	**73.54%**

3) Sonnblick Observatory Dataset: In the evaluation, the LeGO-LOAM algorithm failed to generate a plausible map of the surveyed area, while the KISS-ICP algorithm successfully produced a plausible 3D point cloud map, which will be further evaluated for accuracy. This outcome led to LeGO-LOAM being considered as a failure, while KISS-ICP met the *visual inspection* criterion. Additionally, the HDL-Graph-SLAM algorithm only mapped a partial lower section of the environment, while the LIO-SAM algorithm successfully generated a 3D point cloud map of the surveyed area. Although the map from HDL-Graph-SLAM is incomplete, both it and the map from LIO-SAM will undergo accuracy testing.

The map accuracy measurement results presented in Table V reveal the performance of the KISS-ICP, HDL-Graph-SLAM, and LIO-SAM algorithms. The LIO-SAM algorithm consistently achieves the highest accuracy, surpassing both the KISS-ICP and HDL-Graph-SLAM algorithms, in both the original map and the map with vegetation removed. However, the KISS-ICP algorithm demonstrates poor accuracy, with a significant deviation of points from their corresponding positions in the reference map. While the HDL-Graph-SLAM algorithm initially appears to generate more accurate maps based on Table V, a closer examination reveals its failure to meet accuracy requirements. The limitation of lidar sensors in capturing vegetation is evident in the comparison between the LIO-SAM algorithm and the reference map, where points with larger distances from their corresponding reference points are predominantly found in areas with dense vegetation. Nevertheless, when the vegetation is removed, the LIO-SAM algorithm accurately represents the ground features. In conclusion, the LIO-SAM algorithm consistently delivers superior map accuracy, outperforming the KISS-ICP and HDL-Graph-SLAM algorithms in both test cases.

B. Optimisation Results

This section presents the results of all the optimization methods tested in the study. The first method focuses on improving loop-closure capabilities using the SC-LIO-SAM algorithm, which is evaluated on all three datasets to compare its performance against the original LIO-SAM algorithm. The second method involves the inclusion of GNSS data and is tested on the Rijeka Harbour and Sonnblick Observatory datasets. Finally, the third method examines the optimization of configurable parameters, specifically on the Sonnblick Observatory dataset, which provides "ground-truth" data for precise analysis of any parameter changes.

1) Loop-closure Improvement: The LIO-SAM algorithm faces challenges in its loop-closure segment, which prompted the development of the SC-LIO-SAM algorithm as a potential solution. The SC-LIO-SAM algorithm successfully generates 3D point cloud maps for the Rijeka Harbour and ViF building datasets, demonstrating its effectiveness in handling loop-closure issues. A comparison of map accuracy between the LIO-SAM and SC-LIO-SAM algorithms reveals a tiny difference. Visual examinations also indicate no noticeable disparities between the two algorithms. Figure 4 shows the map accuracy for the SC-LIO-SAM algorithm on the Sonnblick Observatory dataset. Consequently, the SC-LIO-SAM algorithm will be further tested with the remaining performance improvement methods based on these findings.

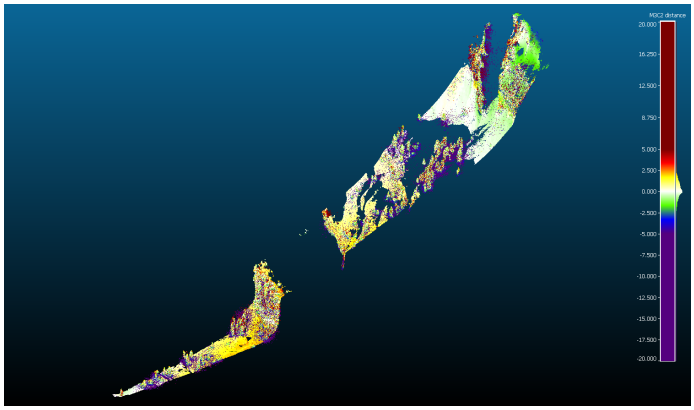

Fig. 4. 3D SLAM generated and "ground-truth" map comparison - SC-LIO-SAM algorithm - Sonnblick Observatory dataset

2) GNSS Data Inclusion: The inclusion of GNSS data in the SLAM process provides multiple benefits. Comparing the map accuracy of LIO-SAM and SC-LIO-SAM reveals that SC-LIO-SAM is used due to its comparable performance. The results demonstrate consistent map accuracy with vegetation present and improved accuracy when vegetation is removed and distance thresholds are reduced. This highlights the potential for increased map accuracy through the integration of

TABLE VI
OPTIMISED PARAMETERS - SLAM GENERATED MAP ACCURACY RESULTS - SONNBLICK OBSERVATORY DATASET

	Sonnblick Observatory			Sonnblick Observatory - vegetation removed		
	LIO-SAM	**SC-LIO-SAM**	**Relative Difference**	**LIO-SAM**	**SC-LIO-SAM**	**Relative Difference**
Map accuracy (+-5m)	88.07%	**88.11%**	**0.05%**	95.13%	**95.31%**	**0.19%**
Map accuracy (+-3m)	81.02%	**81.25%**	**0.28%**	92.93%	**93.11%**	**0.19%**
Map accuracy (+-2m)	73.37%	**73.51%**	**0.19%**	89.57%	**91.01%**	**1.61%**
Map accuracy (+-1m)	56.48%	**56.54%**	**0.11%**	73.54%	**79.81%**	**8.53%**

GNSS data. Additionally, incorporating GNSS data ensures proper compass orientation of the generated 3D maps from the start, eliminating the need for post-processing adjustments. This facilitates the overlay of 3D maps onto 2D maps, as demonstrated by the overlay of a 3D map onto a Google Maps screenshot of the Sušak Harbour in Rijeka, Croatia in Figure 5.

Fig. 5. 3D SLAM generated map overlayed with Google Maps screenshot - Rijeka Harbour dataset

3) Parameter Optimisation: The effects of tuning parameters in four groups are tested individually and then combined to improve map accuracy. The SC-LIO-SAM algorithm is used for testing, comparing the results to the original LIO-SAM algorithm. Tuning the *surrounding map parameter* and *visualisation parameter* groups by decreasing parameter values improves map accuracy. The best parameter values from both groups are combined and fine-tuned, resulting in further accuracy improvements. Table VI shows the comparison of map accuracy between the optimised parameters of SC-LIO-SAM and the original parameters of LIO-SAM. The optimised parameters yield better performance, with a negligible improvement for datasets with vegetation and a noticeable improvement peaking at 8.53% for datasets without vegetation.

VII. DISCUSSION AND CONSLUSIONS

Based on the analysis conducted, it can be concluded that LeGO-LOAM is not suitable for the applications discussed in this research paper due to limitations in map saving and loop closure. Compatibility issues with software versions used in the study may have contributed to its poor performance. The KISS-ICP algorithm also struggled to produce accurate maps in complex environments.

On the other hand, the LIO-SAM algorithm demonstrated success in generating undistorted maps, particularly in non-level terrains. Including GNSS data in the SLAM process showed benefits in terms of map accuracy and compass orientation. Attempts to optimize the LIO-SAM algorithm's parameters resulted in marginal improvements, warranting further investigation.

Future work should involve acquiring diverse datasets with ground-truth information for rigorous evaluation and exploring the impact of individual configuration parameters. Additionally, analyzing external forces acting on the sensor carrier in different environments could enhance the robustness of SLAM algorithms.

In conclusion, this research establishes the viability of using automotive lidars, such as the MOLISENS setup, for accurate environmental mapping. The findings have broad implications across various domains, including marine applications and autonomous transportation. Leveraging the capabilities of automotive lidars opens up opportunities for innovation and safe operations relying on precise environmental maps.

REFERENCES

[1] R. Roriz, J. Cabral, and T. Gomes, "Automotive lidar technology: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6282–6297, 2022.

[2] P. F. McManamon, *Lidar Technologies and Systems*. SPIE Press, 2019.

[3] "Roboat ii: A novel autonomous surface vessel for urban environments." Institute of Electrical and Electronics Engineers Inc., October 2020, pp. 1740–1747.

[4] A. Pantazis, "Lidars usage in maritime operations and eco-autonomous shipping, for protection, safety and navigation for nato allies awareness," 2019.

[5] "Molisens: Mobile lidar sensor system to exploit the potential of small industrial lidar devices for geoscientific applications," *Geoscientific Instrumentation, Methods and Data Systems*, vol. 11, pp. 247–261, August 2022.

[6] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping," July 2020. [Online]. Available: http://arxiv.org/abs/2007.00258

[7] "What is slam (simultaneous localization and mapping)," Avaialble: https://www.mathworks.com/discovery/slam.html (Accessed: 10/10/2022).

[8] C. Debeunne and D. Vivet, "A review of visual-lidar fusion based simultaneous localization and mapping," *Sensors*, vol. 20, no. 7, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/7/2068

[9] "Riegl vz-6000 - 3d ultra long range terrestrial laser scanner with online waveform processing," Avaialble: http://www.riegl.com/uploads/tx_pxpriegldownloads/RIEGL_VZ-6000_Datasheet_2020-09-14.pdf (Accessed: 30/04/2023).

[10] D. Lague, N. Brodu, and J. Leroux, "Accurate 3d comparison of complex topography with terrestrial laser scanner: Application to the rangitikei canyon (n-z)," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 82, pp. 10–26, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0924271613001184

[11] "Distances computation - cloudcomparewiki," Avaialble: https://www.cloudcompare.org/doc/wiki/index.php/Distances_Computation (Accessed: 28/12/2022).

[12] W. Zhang, S. Cai, X. Liang, J. Shao, R. Hu, S. Yu, and G. Yan, "Cloth simulation-based construction of pit-free canopy height models from airborne lidar data," *Forest Ecosystems*, vol. 7, p. 1, December 2020.

[13] B. Valentin, "Evaluation and comparison of 3d lidar based slam algorithms," Master's thesis, Royal Military Academy, January 2021. [Online]. Available: https://www.aia-polytech.be/wp-content/uploads/2021/01/EvaluationAndComparisonOf3DLidarBasedSLAMAlgorithms.pdf

[14] Autoware Documentation, "Available open source slam," Avaialble: https://autowarefoundation.github.io/autoware-documentation/main/how-to-guides/integrating-autoware/creating-maps/open-source-slam/ (Accessed: 09/06/2023).

[15] B. Garigipati, N. Strokina, and R. Ghabcheloo, "Evaluation and comparison of eight popular lidar and visual slam algorithms," 2022.

[16] G. Kim and A. Kim, "Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 4802–4809.

[17] H. Chen, W. Wu, S. Zhang, C. Wu, and R. Zhong, "A gnss/lidar/imu pose estimation system based on collaborative fusion of factor map and filtering," *Remote Sensing*, vol. 15, no. 3, 2023. [Online]. Available: https://www.mdpi.com/2072-4292/15/3/790